Interpretable Machine Learning in Financial Statement Fraud Detection: An Analysis of the Need for Explanations, their Potential, and their Limitations

Inaugural-Dissertation

to obtain the degree of Doktor der Wirtschaftswissenschaften
submitted to the Faculty of Business Administration and Economics
at the Heinrich Heine University Düsseldorf

Presented by
Leonhard J. Lösse

Supervisor:
Univ.-Prof. Dr. Barbara E. Weißenberger
Chair of Management Control and Accounting
Heinrich Heine University Düsseldorf

Düsseldorf, March 2025

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AAER ..... *Accounting and Auditing Enforcement Releases*

ACFE ..... *Association of Certified Fraud Examiners*

AI ..... *Artificial Intelligence*

AICPA..... *American Institute of Certified Public Accountants*

APAS..... *Abschlussprüferaufsichtsstelle (German Auditor Oversight Office)*

AS ..... *Auditing Standard*

AUC ..... *Area Under the (ROC) Curve*

BaFin ..... *Bundesanstalt für Finanzdienstleistungsaufsicht (German Federal Financial Supervisory Authority)*

CFRM ..... *Center for Financial Reporting and Management*

CRISP-DM ..... *Cross Industry Standard Process for Data Mining*

e.g. ..... *exempli gratia*

ESMA ..... *European Securities and Market Authority*

et al. ..... *et alii*

EU ..... *European Union*

FINMA ..... *Swiss Financial Market Supervisory Authority*

FN ..... *False Negative*

FNR ..... *False Negative Rate*

FP ..... *False Positive*

FPR ..... *False Positive Rate*

FREP ..... *Financial Reporting Enforcement Panel (Deutsche Prüfstelle für Rechnungslegung)*

GAAP ..... *Generally Accepted Accounting Principles*

GVKEY ..... *Global Company Key (Compustat)*

IAASB ..... *International Auditing and Assurance Standards Board*

ICE ..... *Individual Conditional Expectation curves*

IDW ..... *Institut der Wirtschaftsprüfer in Deutschland e. V. (Institute of Public Auditors in Germany)*

i.e. ..... *id est*

IPO ..... *Initial Public Offering*

ISA ..... *International Standard on Auditing*

ISQC ..... *International Standards on Quality Control*

LIME ..... *Local Interpretable Model-agnostic Explanations*

MD&A ..... *Management Discussion and Analysis*

NDCG@k ..... *Normalized Discounted Cumulative Gain at the position k*

NLP ..... *Natural Language Processing*

p. ..... *page*

PCAOB ..... *Public Company Accounting Oversight Board*

pp. ..... *pages*

QS ..... *Qualitätssicherungsstandard (German Standard on Quality Control)*

ROC ..... *Receiver Operating Characteristic*

RQ ..... *Research Question*

RUS ..... *Random Under-Sampling*

SEC ..... *Securities and Exchange Commission*

SHAP ..... *Shapley Additive Explanations*

SIC ..... *Standard Industrial Classification*

SOX ..... *Sarbanes-Oxley Act of 2002*

TN ..... *True Negative*

TP ..... *True Positive*

TPR ..... *True Positive Rate*

US ..... *United States (of America)*

USD ..... *United States Dollar*

WPK ..... *Wirtschaftsprüferkammer (German Chamber of Auditors)*

WRDS ..... *Wharton Research Data Services*

XAI ..... *Explainable Artificial Intelligence*

A. Introduction

1. Motivation and Theoretical Background

In the 1990s, *Beneish* developed what later became known as the M-Score, one of the early models proposed to identify earnings manipulations, which was ultimately published in 1999 in the Financial Analysts Journal (*Beneish*, 1999a). A group of students in the United States applied the M-Score as part of a university project and indicated that manipulation might be present in the case of Enron – this was in 1998 (*Ghosh et al.*, 1998). The major collapse of Enron followed in 2001. Although the group of students issued a sell recommendation primarily based on valuation concerns (*Morris*, 2009), this case serves as a striking and illustrative example that statistical models have, at least in some cases, been capable of identifying real cases of manipulations in financial statements, even if such predictions did not necessarily lead to earlier detection by statutory auditors and enforcement authorities.

In the time leading up to Enron, the former Chairman of the Securities and Exchange Commission (SEC), Arthur Levitt, highlighted in a speech – later titled "The 'Numbers Game'" – the pressure exerted by capital markets to meet expectations, and not merely fall just short of them as, failing to do so could lead to exaggerated reactions from the capital markets (*Levitt*, 1998). Conversely, meeting analyst earnings forecasts is rewarded with abnormally positive returns (*Bird et al.*, 2019). In this context *Griffiths* vividly described in his book ‘Creative Accounting’ that “every set of published accounts is based on books which have been gently cooked or completely roasted” (*Griffiths*, 1986, p. 1). Similarly metaphorically, *Giroux* describes that even after the Sarbanes-Oxley Act of 2002 (SOX), which was enacted in response to scandals like Enron, that the "earnings magic continues" (*Giroux*, 2006, p. 5). Thus, after

having peaked around the turn of the millennium for the last time, including the case of Enron, financial statement fraud has once again become paramount with incidents of international interest within the recent years such as Wirecard (*McCrum*, 2020). Against this background the recent case of the German Adler Group showed, that market failure, as theoretically described by *Akerlof* in his paper on market for lemons (*Akerlof*, 1970), can – at least in part – be transferred to the audit market, as Adler Group was seeking to find an auditor for its financial statements (*Bender et al.*, 2022). While regulation and internal auditing can play a preventive role to some extent in mitigating the occurrence of fraud (*Bonrath & Eulerich*, 2024), the continued occurrence of such cases shows: Financial statement fraud, especially if active and meticulously planned, can never be prevented entirely, despite the existence of audits and enforcement structures. Conversely, international auditing firms assess the current environment and its conditions as being more susceptible to institutional crime than ever before (*PwC*, 2022, 2024). Still, according to the Association of Certified Fraud Examiners (ACFE), financial statement fraud remains the rarest form of white-collar crime. However, it results in the largest median loss and has increasingly severe consequences the longer these cases remain undetected (*ACFE*, 2024).

To address this issue and facilitated by an increasing amount of readily available data science applications, current research at the interface of accounting and information systems has been exploring the potential of using machine learning to detect signs of financial statement fraud in firms' financial reports as early as possible. The importance of technology for future approaches to detecting financial statement fraud is by no means solely a subject of research; the auditing profession is also actively working to develop the necessary frameworks. This is

also reflected in broader discussions between professional practice and academia, where digitalization initiatives in accounting and auditing are highlighted as a key challenge for audit firms (*Weißenberger et al.*, 2019). At the same time, universities aim to educate future professionals with the necessary expertise at the intersection of accounting and IT, or respectively data science, to meet these evolving demands (*Bravidor et al.*, 2020). In the specific context of technology-driven fraud detection within the audit profession for example, the discussion paper and exposure draft of the International Auditing and Assurance Standards Board (IAASB) regarding the International Standard on Auditing (ISA) 240 (revised), particular emphasizes the use of technology and how conditions can be established to promote its meaningful and comprehensive use (*IAASB*, 2020, 2024).

In this vein, during the last decade research has focused on developing machine learning models that identify financial statement fraud as exactly as possible. When referring to machine learning models in this setting, it typically involves models categorized under supervised learning. This means a model is trained, applying a selected algorithm, to learn the relationship between various variables based on examples, which is why it is also referred to as 'learning by example' (*Hastie et al.*, 2017). Within this context, this means that the model equations typically use the occurrence of financial statement fraud as a dichotomous dependent variable, also called target, while various independent variables, also called features, aim to explain the occurrence of fraud, for example, in the form of financial metrics derived from financial statements. Starting with simpler, inherently interpretable models such as the M-Score by *Beneish* (1999a), which relied on logistic regressions using only a few financial ratios. This early model was proposed to detect earnings manipulation using financial ratios calculated

based on accounting data. Thereby, the regression model included the dichotomous variable M as the dependent variable, where 1 indicated cases of manipulation and 0 represented non-manipulated cases. As independent variables the model utilized eight financial ratios calculated from disclosed accounting data, including among other the days' sales in receivables index and the sales growth index. Since then, increasingly more complex models have been developed, trained on growing volumes and types of data. This includes approaches using, e.g., support vector machines and neural networks (*Cecchini et al.*, 2010a) and expands both the amount and variety of data incorporated, for instance by including additional narrative textual data from financial statements (*Glancy & Yadav*, 2011; *Purda & Skillicorn*, 2015). In contrast to previous research *Bao et al.* (2020) found, as a decisive step for further research, by using more current algorithms as RUSBoost that using raw financial data items instead of precalculated financial ratios can further improve classification performance.

This development, as described in detail within the review of literature in section B, has resulted in a most relevant drawback: The focus lied almost exclusively on the mere detection of financial statement fraud cases. Despite this, still numerous fraud cases remain undetected. But in addition to these errors of undetected fraud cases, a high number of false positive predictions, i.e., the erroneous classification of legally compliant cases as fraudulent, was a negative side effect that was given little consideration (*Beneish & Vorst*, 2022). As machine learning algorithms typically create black box models that do not allow the potential user to identify the reason exactly why a financial report is classified as (non-)fraudulent, each false positive classification results in high costs. These costs can differ among primary potential user groups of such models as auditors, enforcement authorities and investors, and thus, result from,

e.g., manual audit efforts and/or losses in reputation or foregone investment opportunities (*Beneish & Vorst*, 2022). Therefore, as stated by *Beneish and Vorst* (2022) cost-efficient implementations for potential user groups remain pending. In particular with regard to auditors, a lagging adoption of advanced predictive analytics in auditing is discussed in literature, especially highlighting the use of such approaches remains uncommon for fraud detection (*Vitali & Giuliani*, 2024).

2. Research Gaps and Objectives

With regard to the lack of actual implementation of machine learning approaches for financial statement fraud detection by potential user groups, it is essential to examine the hindering factors and explore the general possibilities for addressing these obstacles in a solution-oriented manner. *Beneish and Vorst* (2022) consider two superordinate options for future research to address the lack of implementation. First, they suggest pursuing approaches that can reduce the number of false positives. Whether further significant improvements in reducing the number of false positive predictions is achievable while maintaining a sufficiently high level in sensitivity, i.e., the ability of a model to detect fraudulent cases, remains questionable as especially as the amount of training data is restricted due to the limited occurrence of financial statement fraud events. Second, to lever the potential of machine learning algorithms in this field, another approach is considered to be promising. It consists in trying to make the fraud detection models trained by machine learning algorithms more transparent by implementing additional analyses to identify the main causes for a classification as being fraudulent.

These options should be considered in the context of literature increasingly calling for addressing the issue of potential biases in machine learning models within accounting and

auditing (*Cho et al.*, 2020). Against this background, *Aboud and Robinson* find that, specifically in the context of data analytics applications for detecting fraudulent financial reporting, the aspect of unproven or insufficiently demonstrated capability of the models represents one of the key barriers to implementation (*Aboud & Robinson*, 2022). Only if these explanation approaches can, for example, target specific manipulations can they create value for different user groups. Explanations could enable individuals with domain expertise to validate the classifications by assessing the driving factors behind them. This would allow explanations to either provide starting points for in-depth audits or offer more transparent risk assessments, enabling investors, for instance, to make informed decisions about excluding certain companies from their portfolios. If these expectations could be met, local explanations might provide a way to promote the implementation of models for financial statement fraud detection, despite the now stagnating classification performance.

To explicitly achieve this, initial exemplary approaches in the field of financial statement fraud detection have been developed to generate local explanations for individual predictions (*Craja et al.*, 2020; *Zhang*, C. *et al.*, 2022). While *Craja et al.* (2020) propose an textual approach to highlight those narratives within financial statements which lead to a classification as being potentially fraudulent, *Zhang*, C. *et al.* (2022) apply various interpretable machine learning analyses to exemplary show the potential of additional explanations by using only financial data. These approaches have in common aiming to enhance transparency by identifying which variables, or respectively which parts of financial statements, have driven a particular classification. Unlike general weightings, as those in regression models, which

provide insights into the overall contribution of variables at the model level, these explanation techniques focus on variables' contribution for individual observations (*Molnar*, 2022).

Building on the persistent lack of implementation of financial statement fraud detection models and first insights into the potential of an increase in interpretability, the overarching question, therefore, is whether more interpretable machine learning approaches can contribute to making predictions more manageable and thereby generate greater utility. Here, more interpretable refers especially to approaches deriving local explanations for individual observations, i.e., identifying the key drivers in the form of specific balance sheet or income statement positions that lead to a particular classification decision for individual observations. This, in turn, could provide valuable insights and serve as a basis for further and more targeted investigations. To this end, the overarching question needs to be broken down. While the previous research has highlighted the general potential of more interpretable models as part of the solution to encourage actual implementation, first, there remains a lack of detailed insights into the needs of potential user groups regarding interpretable machine learning approaches, particularly the promising local explanations. Thus, it is necessary to analyze what kind of explanations could be principally useful. Second, little is known about whether local explanations are truly capable of providing useful insights into the mechanisms of the models, i.e., whether the required explanations are actually able to point out manipulated areas in the financial statement (*Zhang*, C. *et al.*, 2022). These two aspects are therefore discussed briefly below, together with the specific research questions arising from the analyses of the literature, which are also summarized again in section B.4 at the end of the literature review.

Therefore, first with regard to the need for more interpretable detection models, I argue that following this avenue of research more closely makes it necessary to address the diverging perspectives of different user groups of financial statements as well as their legal and operating environments. In a highly regulated sector with outstanding professional requirements for expertise, concerns regarding trust, accountability, and efficient implementation of human-machine-interaction-based applications must be considered as these factors might give rise to different requirements with regard to the types of additional explanations. Therefore, in section C I conceptually address the following two research questions taking into consideration the demand and usability of interpretable machine learning approaches with regard to important potential user groups in their professional environment:

RQ1: What legal and organizational conditions drive the need for financial statement fraud predictions' interpretability?

RQ2: What behavioral interactions must be considered for effective and efficient implementation in a highly regulated setting with high professional requirements?

In a nutshell, the analysis suggests that even though different settings apply for diverging user groups of financial statements, the application of machine learning models for detecting financial statement fraud without additional transparency is reasonable only under very narrow assumptions. Enforcement authorities can assess abstract risk for a risk-oriented selection of firms to be audited in the context of sampling examinations and non-professional investors might reduce financial losses through avoiding investments in potentially fraudulent firms. In contrast, user groups in other potential use cases are regularly prevented from applying opaque models by legal and organizational restrictions and further impairing behavioral factors.

The questions addressed in section C therefore contribute to the literature on machine learning-based financial statement fraud detection first by offering a conceptual framework on major users' demands for more detailed explanations on a local level. Requirements are derived from and discussed against the backdrop of legal and organizational environments.

The analyses indicate a high demand for reliable systems capable of providing local explanations for individual classification decisions. While this demand appears to be lowest for investors, it could be particularly valuable for enforcement authorities and auditors. For these groups, such approaches and tools could facilitate more efficient resource allocation and thereby enable the earlier and more effective detection of fraud cases. In the context of black-box models, legal certainty and the necessary transparency for auditors represent a significant barrier. This challenge could potentially be addressed through reliable local explanations.

Second with regard to the explanations' actual ability, this research includes the analyses of model-agnostic approaches to explain classification results individually on a local level according to the following research questions 3 to 5 (section D). To this end, I train financial statement fraud detection models using the RUSBoost algorithm. As proposed by *Bao et al.* (2020), I use raw financial data items instead of financial ratios as training data. This approach enables the following analysis: The Accounting and Auditing Enforcements Releases (AAER) dataset by *Dechow et al.* (2011), employed as proxies for financial statement fraud and covering all AAER published by the SEC up to 2019, provides a categorization of misstatement types, such as misstated revenues or misstated inventories. These misstatement types can be thematically matched with the financial data items used in model training. If, during the application of local explanations, financial data items stand out that are specifically related to

the actual type of manipulation, these could be considered good explanations, as they would provide concrete indicators of manipulated areas and serve as starting points for targeted, in-depth investigations. I analyze the potential of local explanations using Local Interpretable Model-agnostic Explanations (LIME) and Shapley Values; two model-agnostic approaches which provide local explanations based on a trained classification model, i.e. individually for each analyzed observation rather than on an aggregated level of the whole model. LIME derives local surrogate models, which are inherent interpretable, for an individual observations to be explained by slightly perturbing the observation's feature values and observing the effects on the classification results (*Ribeiro et al.*, 2016a). In contrast, Shapley Values are originally based on a game theoretic approach proposed by *Shapley* (1953) considering features as players in a game and calculating the contribution to each feature for a certain prediction result (*Molnar*, 2022). The features, based on their local explanations provided by LIME and Shapley Values (i.e., their effects on specific classification decisions), are then transformed into a ranking, which enables comparability between LIME and Shapley Values. On this basis, the following key questions, aligned with the respective classification results, are addressed.

Research Question 3 addresses prediction results, in which actual misstatements are correctly classified as misstatements. The focus lies on whether, beyond the actual classification result, explanations can be derived that allow conclusions to be drawn about the manipulated area of the financial statements. To enable such insights, the explanations would need to show that features associated with the manipulated area significantly contribute to the classification outcome. Thus, for local explanations to be considered useful, they should, e.g., in the case of a misstatement of receivables, assign a high explanatory contribution to a feature that represents

receivables-related balance sheet items. Accordingly, for these cases, the research question is stated as follows:

RQ3: With regard to true positive predictions, i.e., detected misstatements: Do features which are related to a certain type of misstatement contribute to the classification as being misstated?

Research Question 4 addresses prediction results in which actual misstatements were incorrectly classified as non-misstatements. The underlying question is conceptually analogous to Research Question 3 and seeks to determine whether, despite the incorrect prediction, the features associated with the manipulated area were nonetheless key drivers of the classification toward a misstatement. This could serve as an additional indication that the inherent mechanisms of the models would be indeed driven by features affected by manipulations. To return to the previous example, a high explanatory contribution from a receivables-related feature – despite an incorrect classification result – could indicate that the models are indeed driven by the manipulated areas of the financial statements. Accordingly, Research Question 4 is formulated as follows:

RQ4: With regard to false negative predictions, i.e., undetected misstatements: Despite their incorrect classification, do features which are related to a certain type of misstatement contribute to the classification as being misstated?

As highlighted by *Beneish & Vorst* (2022) the large amount of false positive predictions represents a major obstacle of the implementation of machine learning models in this context. Therefore, independent of the actual cases of misstatements, it is of particular interest to investigate whether patterns or systematic tendencies can be observed in cases where no

misstatements are present. Such insights could reveal important information about potential biases inherent in the models or explanation approaches. Accordingly, Research Question 5 is formulated as follows:

RQ5: With regard to false positive predictions, i.e., false alarms in the absence of an actual misstatement: Does the distribution of the explanations provide indications that potential biases influence the predictions in a way that does not align with the original training objective?

In summary, based on the analyses it can be concluded that the analyzed explanation approaches, LIME and Shapley Values, are not consistently able to generate sufficiently reliable explanations for a large number of classification decisions. While there are isolated examples of seemingly good explanations that appear to align well with specific misstatement types, negative examples are also evident. When viewed as a whole, initial promising patterns within the distributions for certain misstatement types can be observed. However, these are significantly limited in their explanatory power and reliability due to potential biases and the generally high variability of the explanations. Therefore, the findings of this research contribute particularly by demonstrating that, while the literature presents initial exemplary approaches to local explanations that appear promising, these must be treated with great caution. If individual local explanations are highlighted without verifying them within the overall context of a model, there is a material risk of drawing misleading conclusions, potentially fostering a false sense of security.

3. Design Science Approach

The previously outlined research questions are methodologically examined using a Design Science Approach. Due to the increasing availability of data, powerful software and hardware, data science approaches have become an integral part of any research discipline. *Hey et al.* (2009) even stated data-intensive computing as the "fourth science paradigm" following on empirical, theoretical and computational scientific approaches (*Hey et al.*, 2009). In times of an increase in data types and volumes, it is obvious that this data will also be incorporated into models as potential variables. In this context, *Anderson* put forward the provocative thesis "the data deluge makes the scientific method obsolete" in his article 'The end of theory', stating that the amount of data would lead to circumstances in which "correlation is enough" (*Anderson*, 2008). In contrast, *Box*, as early as 1976, has coined his well-known statement "all models are wrong" because people regularly "make tentative assumptions about the real world which we know are false but which we believe may be useful" (*Box*, 1976, p. 792) that remains highly relevant in this context even today. Subsequently, this statement was also often adapted as "All models are wrong, but some are useful." (*Anderson*, 2008). Though, not only in the age of machine learning but much earlier, great importance was attached to the simplicity of relationships in order to be useful or more useful than more complex models. Following Occam's razor, *Box* describes overparameterization as a sign of mediocrity back in 1976 (*Box*, 1976). Its unaltered relevance is highlighted by prominent examples even in the era of Big Data. The case of Google Flu Trends aimed to enable the early detection and prediction of flu outbreaks based on search queries. However, Flu Trends suffered from overfitting, where the model overly learned historical patterns that were, in part, susceptible to external influences.

This made forward-looking predictions based on new search queries significantly more challenging (*Lazer et al.*, 2014). Detached from the mere availability of more and more data and potential variables and the technical ability to process them, the relevance of a well-considered feature selection is highlighted, e.g., by the findings from *Silberzahn and Uhlmann* (2015, p. 190), as they initiated 29 teams to conduct a research project based on the same dataset to answer the question if "football (soccer) referees are more likely to give red cards to players with dark skin" resulting in different and partially contradictory findings (*Silberzahn & Uhlmann*, 2015). Thus, taking these challenges into consideration, the availability of a large number of potential features itself does not justify neglecting theory and simply incorporating all data available as overfitting might result in the final models. This applies to two key aspects: First, the fundamental methodological approach within machine learning-based research projects, and second, the training of the models themselves.

While the theoretical foundations of financial statement fraud detection as well as its corresponding drivers and characteristics are discussed in sections A.1.3 and A.1.4 as a basis for the reasoning of the feature selection in section D.3.1 the methodological approach is formative for the entire thesis and its structure. Against the backdrop of increasing data-driven approaches, the methodological framework must be contextualized accordingly. In particular, the present application case of financial statement fraud detection represents a practically relevant problem that is to be addressed using a contemporary interpretable machine learning approach. This approach operates within a tension field between classification performance, the practical utility of the model in the form of explanations, and the risk of potentially misleading results. This is accompanied by the demand for more practice-relevant research. The aim should

increasingly be to maintain the necessary scientific rigor while at the same time conducting pragmatic and problem-oriented research (*Drnevich et al.*, 2020). As a result, solutions for accounting research are also seen in the use of more interdisciplinary approaches involving machine learning (*Ke*, 2024). Particularly in the context of interpretable machine learning, the design science approach is also seen as having great potential within the accounting literature (*Sellhorn*, 2020), "offering solutions proactively" (*Fülbier & Sellhorn*, 2023, p. 1104).

Thus, this thesis and in particular section D follow the design science approach. Design science originates from information systems research and focuses on challenges mainly faced by practitioners (*March & Smith*, 1995) and developing specific solutions for these real-world challenges and problems (*vom Brocke et al.*, 2020). *Simon* laid the foundation for this. In his fundamental work "The Sciences of the Artificial", which was first published in 1969, he discusses in particular the different principles to scientific research and research which focuses on artificial objects. These artificial objects originally came mainly from the engineering sciences. In this respect, *Simon* makes it clear that design theory focused on is driven by an expanding use of computers and artificial intelligence tools. And this no longer applied only to engineering, but also to computer science and affected business schools, among others. The necessity to articulate and define design theory with clarity and precision while incorporating computers and their capabilities into research projects was pivotal in establishing its academic legitimacy. In light of this, *Simon* posits that elements of design practice already align with rigorous standards to a desirable extent. However, approaches of these sciences of design are not limited to approaches that exclusively optimize. Rather, the computer-aided generation of alternatives in preparation for decision-making can very well be the subject matter in order to

address the computer-aided solving of real-world problems of various fields of research (*Simon*, 1996). *Hevner et al.* (2004, p. 75) state, design science research "seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts", i.e., it focuses primarily on practical usage and relevance. Besides relevance, another major characteristic is novelty. Novelty refers to innovative ways to address unsolved problems or increase the efficiency of existing approaches (*Geerts*, 2011). What qualifies as an artifact, however, is less clearly defined and not strictly delineated. Thus, an artifact can take various technical forms, including models that explicitly contribute to solving an identified problem (*Peffers et al.*, 2007).

Design science's application has already been established at an early stage in the field of application-oriented AI-based research (*Baldwin & Yadav*, 1995). The growing number of studies based on machine learning has increased its popularity both in accounting (*Kelton & Murthy*, 2023; *Zhang*, G. *et al.*, 2022) and specifically in audit research (*Huang*, S.-M. *et al.*, 2022; *Kogan et al.*, 2019). *Marten et al.* see research projects that follow the design science approach as an explicit opportunity for research to actively shape the implementation of AI-based tools in audit (*Marten et al.*, 2022). However, multiple publications seem to follow design sciences approaches without explicitly mentioning or describing respective methodologies, so that the actual number of publications based on the method can easily be underestimated. Therefore, transparency is also called upon to present steps in a comprehensible manner and to categorize them methodically (*Hevner et al.*, 2024).

Even if rigour is less concretely defined in the context of design science research (*Winter*, 2008), steps have been established that are widely accepted. The thesis is based on the

following key elements of design science research (*Peffers et al.*, 2007) which are the basis for its structure (*Gregor & Hevner*, 2013). As outlined in the following section these elements comprise the problem identification and motivation, definition of a solution's objectives, a description of design and development followed by a demonstration and evaluation before it is finally communicated (*Peffers et al.*, 2007).

4. Outline of the Thesis

As a strictly solution-oriented approach, the necessary steps are designed to address a significant practical problem and develop a proposed solution, the artifact (*Peffers et al.*, 2007). Here, the proposed artifact is a trained financial statement fraud detection model and its derived local explanations, offering insights into which financial statement positions drive the classification as potentially fraudulent or non-fraudulent. In accordance with the steps proposed for design science research projects by *Peffers et al.* (2007), the thesis is structured and outlined as follows.

First, in section B, an overview on financial statement fraud, its detection using machine learning-based approaches, and current advances in interpreting algorithmic predictions by interpretable machine learning is given. This includes the fundamentals of financial statement fraud, its consequences, and theories on favoring circumstances, along with the conceptual foundations of interpretable machine learning. These strands are brought together in the third part of section B, which provides a comprehensive overview of machine learning-based financial statement fraud detection models in the literature, beginning with early approaches, followed by major developments, and culminates in the latest advancements with initial efforts to provide individual explanations for specific predictions. This review of literature identifies

and highlights the scientific and practical problems to be addressed by this thesis. Namely, the detection of financial statement fraud is at least partly the task of or expectation towards multiple potential user groups of such models, especially auditors and enforcement authorities. Numerous machine learning approaches are intended to provide decision support. In addition to unidentified manipulations, a key challenge is dealing with false positives. Even advanced approaches are characterized by too high costs for actual implementation because the effort resulting from addressing numerous false positives is too costly. An approach that allows especially false positives to be manageable could encourage actual implementation of the systems. Furthermore, existing models are often characterized as black box models, thus, more transparent models could further increase trust and support actual usage. Thus, the requirement of classification approaches with both, a high performance and an increase in transparency to generate better manageable predictions, motivates this research.

Second, section C analyzes conceptually the requirements from the perspectives of potential major user groups of algorithm-based financial statement fraud detection tools, namely auditors, enforcement authorities, and investors, against the background of their specific organizational and regulatory circumstances. This analysis aims to conceptually elaborate on the type of interpretability that models must provide to meet the conditions and requirements of different user groups. This defines the objectives of a solution in such a way that global interpretations, such as weights in a simple regression model, would generally not be sufficient. Instead, local explanations at the level of individual observations could offer the potential to enhance practical usability and facilitate more targeted further investigations. This conceptual analysis in section C, parts of the literature review in section B and selected phrases within the

introduction have been separately published together with Barbara E. Weißenberger in 2023 under the title "Using Interpretable Machine Learning for Accounting Fraud Detection – A Multi-User Perspective", in Die Unternehmung – Swiss Journal of Business Research and Practice Volume 77, Number 4, pp. 113–133. Minor changes to the wording have been made for consistency.

Section D covers the design and development of the financial statement fraud detection model and in particular its explanations, a demonstration of local explanations and their evaluation. In more detail, the financial statement fraud detection model is trained, and its performance is evaluated. Subsequently, illustrative examples for post-hoc local explanations are given applying LIME and Shapley Values. A comprehensive analysis of local explanations of the model provided by LIME and Shapley Values enables the evaluation of the model's inherent mechanisms to identify manipulated areas of financial statements as well as of the local explanations' ability to serve as indicators for these misstated areas of a financial statement. Therefore, the analysis focuses particularly on the ability of local explanations to point to actually misstated areas of financial statements and further examines false positive predictions for potential biases in the model or the explanation approaches.

Finally, a conclusion is drawn, limitations are explained and an outlook on potential future research avenues are built. The communication of the research results, as formally proposed by *Peffers et al.* (2007) as the final element of design science research project, is achieved by the publication of this thesis.

B. Literature Review

1. Financial Statement Fraud (Detection)

1.1. Classification of Financial Statement Fraud

The ACFE classifies occupational fraud in the three main categories corruption, asset misappropriation and financial statement fraud (*ACFE*, 2024). First, corruption comprises cases in which employees abuse their influence over business transactions, breaching their duty to the employer in order to gain an advantage. Second, asset misappropriation typically covers stealing and misusing the employing organization's resources, which is most congruent with *Hollinger and Clark*'s fundamental discussions on occupational crime in 'Theft by Employees' (*Hollinger & Clark*, 1983). And third, the ACFE defines financial statement fraud as "a scheme in which an employee intentionally causes a misstatement or omission of material information in the organization's financial reports (e.g., recording fictitious revenues, understating reported expenses, or artificially inflating reported assets)" (*ACFE*, 2024, p. 104). However, the categories are by no means mutually exclusive. Financial statement fraud in particular is typically closely intertwined with asset misappropriations.

The further differentiation of the ACFE is subsequently only partially relevant in the context of this analysis. Financial statement fraud can occur in the form of both overstatements and understatements, especially of a company's net worth or net income. Typically, understatements rather occur in the context of tax fraud, i.e. understated income leads to avoided tax expenses, while overstatements, on the other hand, are aimed at conveying an image of better performance in order to obtain sufficient external capital, but also to make the company more attractive to investors so that they invest in shares of the company.

In addition to this categorization of the ACFE, according to the perspective of the auditing profession, errors are distinguished from fraudulent acts on the basis of intent in accordance with ISA 240.2 or Auditing Standard (AS) 2401.5. However, the auditing profession also states that "fraud is a broad legal concept" and in the context of the ISA the profession primarily focuses on cases of fraud leading to material misstatements in financial statements (ISA 240.3). In contrast, unintentional misstatements are categorized as "errors". If the requirement of intent is met, a misstatement is classified as "fraud". ISA 240.3 makes a further distinction between intentional misstatements relevant to the auditor; these can arise both from misappropriation of assets and from the manipulation of financial reporting. Further, ISA 240.12 clarifies regarding people involved, that fraud is "an intentional act by one or more individuals among management, those charged with governance, employees, or third parties, involving the use of deception to obtain an unjust or illegal advantage."

Hence, for the purpose of the following analyses, I will primarily focus on financial statement fraud as intentional overstatements of net worth or net income as reflected in manipulated financial statements. On the one hand, the restriction to intentional misstatements aims to focus on cases with the most severe consequences and criminal relevance. On the other hand, the limitation to overstatements of earnings or revenues targets classic cases of financial statement fraud, typically characterized by an overly positive presentation of a company's financial condition. By restricting the cases of misstatements to overstatements of earnings or revenues, the training of models can remain consistently focused on identifying overly positive presentation by companies. Typical cases of understatements of earnings or revenues are often driven by deviating motives, e.g., such as reducing tax liability in cases of tax fraud, rather than

maximizing company value through an overly positive presentation. These deviating motives and characteristics of manipulations could negatively impact the performance of detection models when applied to classic cases of financial statement fraud. Therefore, as a proxy for financial statement fraud incorporated observations are limited to cases of intentional overstatements (see section D.2).

1.2. Damages and Consequences of Financial Statement Fraud

According to the ACFE financial statement fraud is the least common category of occupational fraud but leads to the greatest median loss and becomes more serious the longer the fraud remains undetected. Of the cases of occupational fraud considered in the ACFE's Report to the Nations, only 5 % pertain to financial statement fraud. However, these cases are associated with significantly higher financial losses per incident, reflected in a median loss of USD 766,000 – substantially exceeding, for instance, the median loss of USD 200,000 in cases of corruption (*ACFE*, 2024). This is supported by experimental evidence, if fraud remains undetected there is the potential for slippery slopes in management's behavior resulting in exponential growth of fraud damages (*Cheynel et al.*, 2024). Thus, highlighting the importance of preventive functions such as internal auditing (*Bonrath & Eulerich*, 2024) in order to limit potential damages.

With regard to the damage caused by financial statement fraud, it can never be precisely quantified. Rather, quantification is limited to estimates (*Rezaee & Riley*, 2009). The difficulty in quantifying the financial statement fraud costs results from various reasons. On the one hand, different groups of stakeholders can be affected and there is often a lack of detection or insufficiently detailed processing and documentation. And finally, numerous problems and

damages result from financial statements frauds (*Rezaee & Riley*, 2009) complicating the estimations: These include macroeconomic consequences such as loss of confidence in the capital market with a resulting reduction in capital market efficiency which may subsequently be accompanied by stricter regulation. Direct financial losses arise in particular in the form of negative market returns (*Feroz et al.*, 1991; *Gerety & Lehn*, 1997) and costs from insolvencies, i.e. various liabilities to both operating business partners, banks and investors cannot be serviced, as well as costs for subsequent legal disputes. But there is also real economic damage to operations, e.g., production facilities having to be closed, as well as direct damage to the careers of individuals, starting with managers and employees and possibly extending to employees on the investor side (*Rezaee & Riley*, 2009).

Although, for existing employees the damages will outweigh potential previous benefits, because previously higher salaries etc. is subsequently lost due to the more serious consequences, such as losing their job (*Choi & Gipper*, 2024), one of the very few positive side effects could relate to the personnel development of experts. *Carnes et al.* find that when students are confronted with local fraud cases during their formative years, it is more likely that they major in accounting and subsequently becoming a CPA (*Carnes et al.*, 2023). Accordingly, such fraud cases seem to have at least the positive side effect that young prospective experts are sensitized to fraud and pursue more in-depth accounting education, despite the associated economic damage. And thus, may be able to make a small contribution to detecting or preventing future cases of financial statement fraud at an earlier stage.

1.3. Theory of Occupational Fraud

In the context of machine learning based financial statement fraud detection, the selection of features[1] to be incorporated into a detection model is of decisive importance. In most cases, the selection of features is based on previous research and theoretical frameworks, as, e.g., the fraud triangle (*Gepp et al.*, 2021). For better classification, the main theoretical concepts and their development are therefore briefly described below.

The scientific field of research on occupational fraud was initially shaped by *Edwin H. Sutherland* and *Donald R. Cressey* and continues to be so today. *Sutherland*, a US criminologist, was the one who coined the widespread term white-collar crime (*Wells*, 1997). According to *Sutherland*, "a white-collar crime is defined as a violation of the criminal law by a person of the upper socioeconomic class in the course of his occupational activities" (*Sutherland*, 1941, p. 112). Subsequently, it was a student of *Sutherland* who laid the conceptual foundations for what later became known as the fraud triangle illustrated in Figure B-1 (*Wells*, 1997). In 'Other people's money', *Cressey* identified three factors pressure, opportunity and rationalization which favor or can facilitate fraudulent behavior (*Cressey*, 1953). *Cressey* did originally not call it a triangle, but the term fraud triangle was later coined by *Wells*, the founder of the ACFE (*Morales et al.*, 2014; *Wells*, 1997). As not all three conditions of the fraud triangle must be present, but depending on each weights, e.g., high situational pressures and available opportunities might overcompensate an usually high level of personal integrity, the fraud

[1] In this doctoral thesis, the terms "variable" and "feature" are used synonymously. Although the term variable is more inherent to empirical accounting research, the term feature is predominantly used, which is common in the machine learning literature. Both mean the independent variable in the statistical sense. The same applies to the dependent variable, which is also referred to as a "target" in data science.

triangle has also been presented as the so called fraud scale (*Albrecht et al.*, 1984). *Albrecht*, in turn, was the first president of the ACFE (*Morales et al.*, 2014).

Figure B-1: The Fraud Triangle and the Fraud Diamond

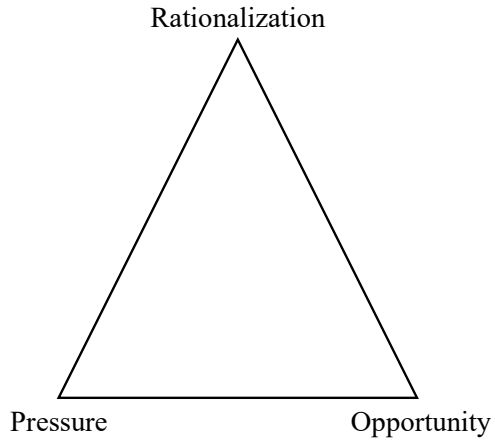


Illustration of the Fraud Triangle
based on Wells (1997)

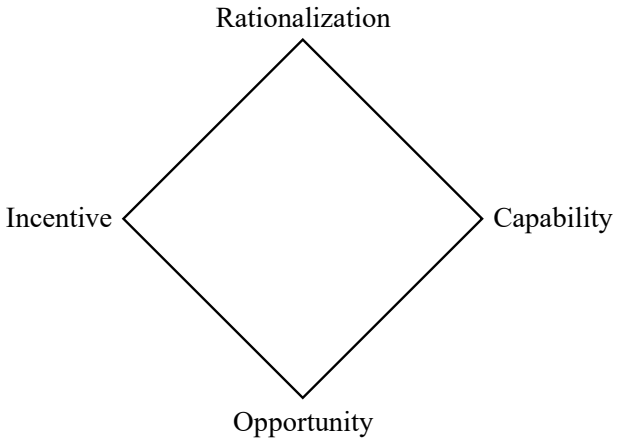


Illustration of the Fraud Diamond
based on Wolfe/Hermanson (2004)

Due to previous limitations, several models and theories have been proposed and subsequently refined (*Dorminey et al.*, 2010). One exemplary extension is the fraud diamond (see Figure B-1). In comparison to the fraud triangle, incentive extends to other motivators than pressure, and capability has been added as a fourth factor and takes into account that personal traits and abilities must be present for fraud to actually occur, even if the other three factors might already have a favorable effect (*Wolfe & Hermanson*, 2004). This extension has recognized an increasing importance and attention to practice and science (*Hermanson & Wolfe*, 2024). In addition, further attempts aim to explain occupational fraud as comprehensively as possible, so that, e.g., a meta-model is proposed, particularly in the context of accounting (*Dorminey et al.*, 2012). Despite these developments the fraud triangle suffers from a limitation in form of the dark triad: The dark triad covers Narcissism, Machiavellianism, and psychopathy (*Paulhus &*

Williams, 2002). In cases of people with a dark triad personality, which are overrepresented in corporate management, opportunity alone might trigger fraud (*Epstein & Ramamoorti*, 2016).

However, as described by the preceding limitation of the fraud triangle it is often difficult to observe motivations or justifications that mainly concern the specific situation of individuals. In order to place the theoretical framework of fraud less on motivations that are difficult to observe, *Albrecht et al.* have established the fraud element triangle, which focuses on investigative methods with regard to the theft act itself, and further associated traces with regard to indicators of efforts to conceal the fraud or indicators for the subsequent spending or utilization of stolen assets (*Albrecht et al.*, 2016). *Gepp et al.* address this shortcoming of the fraud triangle in a different way. They supplement it with the aspect "suspicious information" to the fraud detection triangle (see Figure B-2). Both, the three factors of pressure, opportunity and rationalization, which are often not observable, as well as suspicious information can indicate an increased likelihood of fraudulent financial statements (*Gepp et al.*, 2024).

Figure B-2: The Fraud Element Triangle and the Fraud Detection Triangle

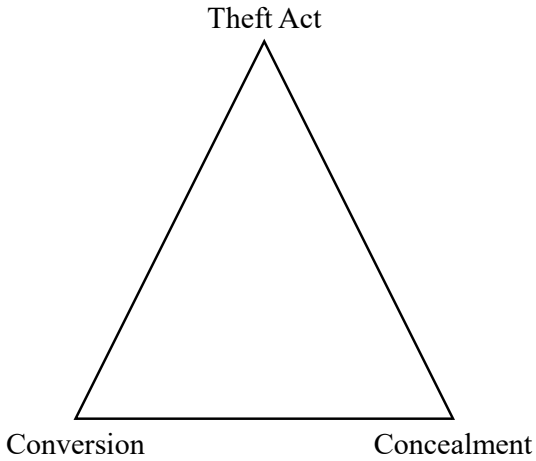


Illustration of the Fraud Element Triangle
based on Albrecht et al. (2016)

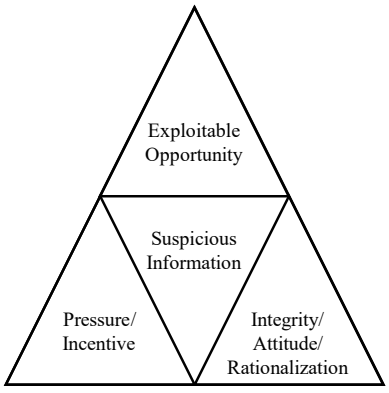


Illustration of the Fraud Detection Triangle
based on Gepp et al. (2024)

1.4. Drivers and Characteristics of Financial Statement Fraud

The preceding remarks on theoretical approaches aiming to explain occupational fraud can be used to derive indicators that can be used to detect financial statement fraud. Depending on the approach, all conceivable variables could be considered, which in turn could be grouped differently. However, for the purpose of this thesis I will subsequently focus on drivers, which relate to financial figures.

Early approaches as, e.g., by *Green and Choi* tended to incorporate already used variables for analytical audit procedures (*Blocher & Cooper*, 1988; *Daroca & Holder*, 1985; *Green & Choi*, 1997). This was due to the fact that there were no formally recognized theoretical guidelines for selecting variables to identify financial statement fraud in either the practitioner or academic literature (*Green & Choi*, 1997). Regarding the selection of variables to be considered in model training, *Beneish* states a crucial point as follows. In introducing his M-Score, he addresses the question if accounting data can be used to detect earnings manipulation. To technically do so, he employs a logistic regression with a dichotomous dependent variable indicating whether a misstatement occurred and eight selected financial ratios as independent variables. For a meaningful selection of variables, *Beneish* identifies two factors to be taken into account**:** "The model's variables are designed to capture either the financial statement distortions that can result from manipulation or preconditions that might prompt companies to engage in such activity" (*Beneish*, 1999a, p. 24).

This means that the theoretical approaches were used in two ways. Firstly, to identify variables that leave possible manifestations or traces of balance sheet manipulations. Secondly, to select variables that may indicate conditions that favor the occurrence or perpetration of

balance sheet manipulations. *Cecchini et al.* clarifies this as the attempt to determine attributes, which are correlated with fraud (*Cecchini et al.*, 2010a). *Beneish*, e.g., selected variables out of three categories. He included variables relating to signals of poor future prospects, cash flow and accruals related variables and variables regarding contract-based incentives (*Beneish*, 1999a). Even if *Beneish* did not yet relate the categories to the fraud triangle at that time, they can be categorized remarkably well against the background of the fraud triangle and thus also theoretically justified: Poor future prospects can exert pressure on decision-makers, cash flow and accruals related variables can be regarded as indicators for the extent to which opportunities were exploited, and contract-based incentives can be classified as incentives or pressure and as well as a starting point for internal legitimization.

The aforementioned analytical procedures can comprise (1) comparisons to prior periods, (2) comparisons with anticipated results, and (3) relationships of financial statements' elements (*Daroca & Holder*, 1985). A comparison to prior periods could, e.g., be a variable for growth, as a rapid growth is a key indicator for irregularities (*Loebbecke et al.*, 1989). Growth variables are already incorporated in early studies, referring, e.g., to the growth of total assets (*Beasley*, 1996) or the growth in sales (*Beneish*, 1999a). Relationships of financial statements' elements comprise a wide range of figures: From simple ratios as return on sales which has early been regarded as a meaningful figure in analytical audit procedures (*Blocher & Cooper*, 1988), up to more complex figures such as Altman Z Score for a company's financial condition (*Summers & Sweeney*, 1998). Thereby, the Altman Z Score is a model based on multiple discriminant analysis used to predict the probability of corporate bankruptcy (*Altman*, 1968). This last mentioned score might also be replaced by current credit ratings, as e.g., rating actions

by Standard & Poor's might offer some predictive power in terms of statistical fraud detection models (*Huang et al.*, 2023). However, those cases are often not replicable based on the actual financial figures of financial statements, as the underlying calculations are typically not transparent. The second mentioned type of analytical procedures, comparisons with anticipated results, would rather be comparable to the results of a detection model rather than as serving as an input factor of a model.

Compared to the previously mentioned, partially commercial credit ratings, most research models focus on using publicly available and transparent input factors. This applies both to early, less complex models, such as the logistic regression-based M Score proposed by *Beneish* (1999a), as well as to increasingly complex models developed over time, employing algorithms like support vector machines or neural networks (*Cecchini et al.*, 2010a). Although contemporary powerful machine-learning models can deal effectively with large amounts of data and a wide variety of features, *Bao et al.* show, for example, that their theory-based selection of variables leads to a better performing model compared to adding all available features. This underlines the value of theory-based variable selection, which is ultimately based on human expertise and (still) appears to be superior to pure machine learning approaches (*Bao et al.*, 2020). Such a theory-driven approach is not only object to current scientific approaches but also deeply rooted in the audit standards. ISA 240.A1 characterizes the circumstances in which fraud occurs on the basis of the three criteria pressure or incentive, opportunity and rationalization in order to explicitly sensitize auditors to these circumstances. It is therefore appropriate, and in some cases necessary, for contemporary models to nevertheless take into account conformity with theoretical knowledge and the associated regulations.

1.5. Responsibilities to Detect Financial Statement Fraud

To prevent financial statement fraud before its occurrence, firms implement, among others, internal control systems by corporate governance exercised via a firm's one- or two-tier board system. Still, these mechanisms might fail due to negligence or top management fraud. It is the role of a firm's statutory auditors to externally monitor the compliance of the accounting systems as well as the resulting financial statements with all existing norms and regulations including professional standards so that they provide a true and fair view of the firm. Thus, undetected fraudulent financial statements are assigned to the auditor's area of responsibility regardless of actual responsibility for the failure, which is commonly known as the audit expectation gap (*Koh & Woo*, 1998; *Ruhnke & Schmidt*, 2014), even if the standards clearly state the limits of a financial statement audit in form of a reasonable assurance (ISA 200.5). Therefore, audit research and practice strive for continuous improvement towards more effective audit procedures and technologies, and the shift from traditional to digital audits including machine-learning based fraud detection which is explicitly seen as an opportunity to reduce this gap (*Fotoh & Lorentzon*, 2023).

The work of auditors is also closely connected to their oversight and the institutions responsible for it. In Germany, the German Chamber of Auditors (Wirtschaftsprüferkammer, WPK) generally conducts professional supervision under § 61a of the Public Accountant Act (Wirtschaftsprüferordnung, WPO) and decides on further professional measures in cases where there are concrete indications of potential breaches of professional duties. Additionally, § 57a WPO stipulates that quality control or peer review procedures are required, at least every six years, for statutory auditors and their auditing firms. The supervision of statutory audits for

public interest entities (PIEs), however, falls under the responsibility of the Auditor Oversight Body (Abschlussprüferaufsichtsstelle, APAS), as outlined in § 66a WPO. The APAS was established in 2016 through the Abschlussprüferaufsichtsreformgesetz (APAReG), which implemented the EU Directive 2014/56/EU and incorporated the provisions of Regulation 537/2014/EU. Under § 66a WPO, APAS is authorized to conduct both event-driven inspections, triggered by concrete indications of professional misconduct, and routine inspections without specific cause. Comparable to this, in the United States, responsibility for peer reviews as a form of self-regulation lies with the American Institute of Certified Public Accountants (AICPA). The US counterpart to APAS, the Public Company Accounting Oversight Board (PCAOB) was established through the Sarbanes-Oxley Act of 2002 (SOX) as an institution for public oversight of the profession with regard to the supervision of auditors of public companies. Concerning the effects of these forms of professional oversight on the quality of audits in general, *Löhlein* (2016) finds that both forms of oversight – peer reviews and PCAOB inspections – are generally associated with improved audit quality in the US context. Complementing this, and with a particular focus on PCAOB oversight, *Gipper et al.* (2020) find that public audit oversight can further enhance reporting credibility. However, *Elshandidy et al.* (2021) note in their literature review that these effects are not consistently observed across all studies. Nonetheless, from a professional practice perspective, as e.g., mentioned in a contribution to the CPA Journal, it is widely asserted that most major accounting firms would likely agree that today's audit quality is generally regarded as better than it was at the beginning of the millennium (*Goelzer*, 2020). Thus, this highlights the significant

advancements and stricter measures in professional oversight that have been introduced since the turn of the millennium.

However, as financial statements play a paramount economic role, and manipulations can never be fully prevented – nor through the described advancements in professional oversight – national enforcement authorities re-examine the audited financial statements on a sample or ad hoc basis. Within the European Union (EU), competencies are codified in Article 4 of Directive 2004/109/EC ('Transparency Directive') and delegated to national authorities, such as the German Federal Financial Supervisory Authority (BaFin). Well-known counterparts outside of the EU are, e.g., the Swiss Financial Market Supervisory Authority (FINMA) or SEC. At the European Union level, the European Securities and Markets Authority (ESMA) holds powers under Article 8(2) of Regulation 1095/2010/EU, which serve a coordination and oversight function concerning the respective national competent authorities.

Besides such regulatory requirements, price mechanisms on capital markets also serve as an indirect external governance mechanism, as investors may act as a kind of market corrective. While the average investor seeks to avoid downside risk by refraining from investing in potentially fraudulent companies, identified risky companies can also be leveraged to profit, if investors sell the shares short and thus speculate on the discovery of possible manipulation. *Massa et al.* (2015, p. 1701), e.g., state "that short selling functions as an external governance mechanism" by a disciplining effect reducing earnings management, with a practical example being Wirecard and the short selling activities of Fraser Perring who published the so called 'Zatarra Report' in 2016 in which he accuses Wirecard of accounting fraud and corruption (*Langenbucher et al.*, 2020).

2. (Interpretable) Machine Learning

2.1. Origin of Machine Learning

Even if the latest trend around machine learning and artificial intelligence seems to be a current development, its origins date back to the middle of the 20th century. *McCarthy et al.* applied the term Artificial Intelligence (AI) when they described "making a machine behave in ways that would be called intelligent if a human were so behaving" (*McCarthy et al.*, 1955). Even the outdated but well known so called "Turing Test", originally named "imitation game", raised the question early on distinguishing between human capabilities and those of machines (*Turing*, 1950). In 1959 *Samuel* coined the term Machine Learning more precisely in the context of the game of checkers. There he refers to the term while stating that in the game of checkers a computer can be programmed in such a way that it can independently learn to play better than the person who initially wrote the program (*Samuel*, 1959).

Overall, AI is used as a broader overarching term, while machine learning represents a subfield of AI, since not every AI approach uses machine learning to achieve competence (*Russell & Norvig*, 2021). AI comprises additional approaches besides machine learning such as expert systems, robotics or natural language processing (NLP) as well. Machine learning as a subfield of AI is characterized by the ability to learn by experience. I.e., machine learning are methods which are able to automatically detect patterns in data and, building on this, perform different kinds of decision making, e.g., predicting future data (*Murphy*, 2012). To further specify and differentiate this from each other, Deep Learning is again a subfield of machine learning. Deep learning approaches also learn from experience; but "understand the world in terms of a hierarchy of concepts" (*Goodfellow et al.*, 2016, p. 1), thus, these approaches can

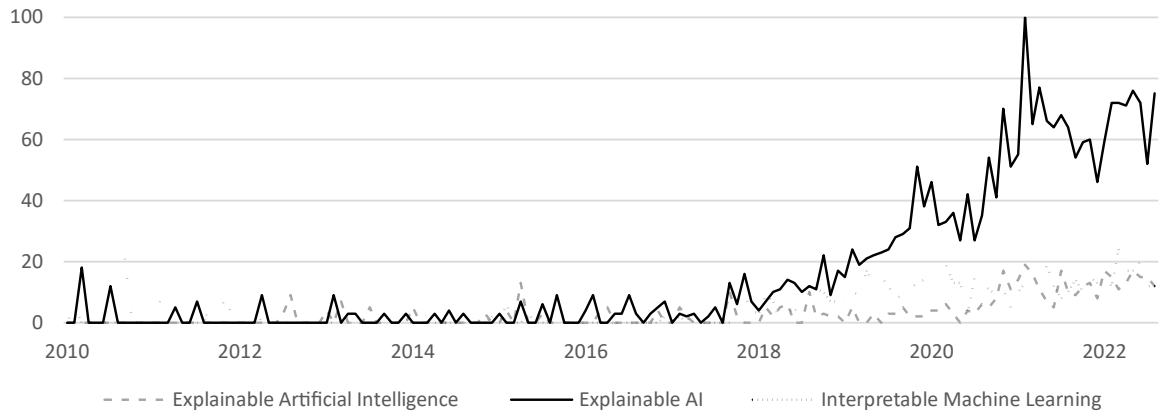
stepwise learn complicated concepts based on simpler ones. Technically, deep learning usually refers to deep artificial neural networks (*Janiesch et al.*, 2021).

With regard to machine learning-based financial statement fraud detection models, especially supervised learning approaches are applied. In contrast to unsupervised learning, which only uses unlabeled input data to discover specific patterns in data, supervised learning trains a model by both, inputs and labeled outputs (*Murphy*, 2012). I.e., input variables, also called features or attributes, are mapped to output variables, also called targets or response. Based on this mapped data, various algorithms can be applied to train a model, which learns the patterns between the input and output variables.

2.2. Recent Developments Towards more Interpretable Approaches

The field of interpretable machine learning has received considerable attention, at the latest since 2017/2018 (Figure B-3). The central object of research is to develop and apply approaches and procedures to open the so-called black box of AI (*Castelvecchi*, 2016). Although the idea of addressing black box models' transparency is not new, demand has grown because of the actual widespread and increasingly simple application of machine learning-based systems (*Samek & Müller*, 2019). In this context, terms comprising explainable AI, also abbreviated as XAI, or interpretable machine learning refer to a common core (*Adadi & Berrada*, 2018). In spite of the popular and widespread usage of the phrase 'explainable' AI, the term 'interpretable' (machine learning) is no less common in science. In the following, both terms are used synonymously.

Figure B-3: Google trends search related to interpretable machine learning (all categories)



A common understanding of interpretability is crucial to concepts like interpretable machine learning. I follow *Miller* (2019, p. 8), who adopts interpretability in the context of artificial intelligence as "the degree to which an observer can understand the cause of a decision" and further equates the terms interpretability and explainability. A widely used approach to substantiate the phenomenon of interpretable machine learning comprises two essential components, namely "produce more explainable models while maintaining a high level of learning performance" and "enable human users to understand, appropriately trust, and effectively manage the emerging generation of artificially intelligent partners" (*Gunning*, 2017).

The first aspect targets the performance interpretability trade-off. In simplified terms, this trade-off means that a higher degree of flexibility of algorithmic approaches, which might enable models to be trained on more complex relationships, is usually achieved at the expense of interpretability (*Arrieta et al.*, 2020; *James et al.*, 2021). Though, this must not always be the case: *Rudin* (2019) states that interpretability does not necessarily have to come at the

expense of more accurate predictions. Particularly in contexts such as medical or criminal justice decisions she cautions against favoring black-box models over inherently interpretable. However, the scientific field of machine learning-based financial statement fraud detection increasingly incorporates black box models to improve prediction performance in particular (see section B.3.2). As these contemporary machine learning applications are increasingly based on complex relationships between input and output data, e.g., by using algorithms based on support vector machines or neural networks, current approaches to interpretable machine learning provide post hoc explanations based on approximations as, e.g., surrogate models in order to draw explanations from the models in retrospect (*Adadi & Berrada*, 2018). This is accomplished by conducting additional analyses to provide insights in how a given classification is achieved, or, in other words, to allow for a peek into to the algorithm's black box. The advantages of such post hoc analyses are twofold. First, they are often model-agnostic, i.e., independent of the underlying algorithm. Second, the trained model itself and its performance remain unaffected because the analyses are applied within the already trained model (*Molnar*, 2022).

From a technical point of view, interpretable machine learning comprises an ever-growing number of approaches to explain models. Feature relevance explanations, e.g., refer to quantitative scores which variables contribute most to a model's prediction. In contrast, visual explanation aims at illustrating mechanisms, e.g., in form of heatmaps which highlight parts of images contributing to a certain prediction, while text explanations go beyond by attempting to provide text-based explanations of models' functionalities in a verbally descriptive way that is more easily understandable for humans (*Arrieta et al.*, 2020). A further aspect of the techniques

is scope. A large part of recent empirical research attempts to analyze higher-level relationships between selected variables in experiments, surveys or archival data-based studies. In this context, weights from linear regressions can be described as globally interpretable as they reflect the context at the level of the entire model. The same applies to the previously listed feature relevance explanations such as feature importance. In contrast to global interpretability, some post hoc approaches generate explanations only at the local level, that is within a very much restricted scope. In that case, interpretations are derived from the level of individual predictions. I.e., the approaches are based on slightly manipulating input data around an observation while observing their effects on the prediction, resulting in simplified locally valid models (*Molnar*, 2022).

With post hoc explanations, the second previously cited aspect by *Gunning* (2017) in form of an human user enablement to understand and build trust into contemporary approaches can be supported. These post hoc analyses can enable improved information transfer. As contemporary algorithms often result in black box models, their trained inherent mechanisms remain opaque for human users. Without a general avoidance of inherently not interpretable algorithms, post hoc explanations are the central approach to still being able to gain explanations about a model's internal mechanisms. To achieve the described necessary traceability and establish a foundation for – when supported by results – justified trust in models, the explanations must meet certain requirements when applied by human experts. According to *Adadi and Berrada* (2018) these requirements are most likely met if they can be classified as human-like or human-friendly. As the terms indicate, human-like refers to the extent to which the approaches' outputs resemble explanations of humans. Human-friendly

describes how well explanations can be understood by humans. This is significantly influenced by how contrastive, selective and social the explanations are, which are considered decisive factors for explanations' quality (*Miller*, 2019).

Against the background of these developments in the field of machine learning and its interpretability, the developments in the specific field of application of financial statement fraud detection of the last decades is subsequently analyzed.

3. Machine Learning-Based Financial Statement Fraud Detection Models

3.1. Early Approaches

The fundamental idea of using statistical approaches to identify anomalies that could indicate human manipulation is not a new one that would only have emerged in the age of big data and machine learning. The so-called Benford's Law is probably the best-known example of this. Since leading digits of numbers do not occur evenly in a wide variety of applications, but lower digits are disproportionately represented, deviations from this distribution can indicate human adaptations (*Benford*, 1938). The early approaches were more likely to be described as statistical approaches and not yet as machine learning as we understand the term today. The Altman Z Score is one of the first statistical prediction models to be developed on the basis of balance sheet financial data, although not in the area of detecting balance sheet manipulation, but related to this for predicting bankruptcy. *Altman* applies a multiple discriminant analysis, using five independent variables, including the working capital ratio and retained earnings relative to total assets, to derive a discriminant function. Based on the resulting Z-score, the dependent variable of the function, *Altman* derives thresholds that can be used to classify companies as either bankrupt or non-bankrupt (*Altman*, 1968). In terms of application, the

models for identifying earnings management are closer to the use case of financial statement fraud detection. Although earnings management is not equivalent to an actual balance sheet manipulation, it is a deliberate shaping or exploitation of accounting rules in favor of the company. In this context, the Jones Model (*Jones*, 1991) or the subsequent Modified Jones Model (*Dechow et al.*, 1995), for example, are still highly relevant in the literature. *Jones* proposed a regression-based model that decomposes a company's total accruals into non-discretionary and discretionary accruals. The dependent variable, i.e. total accruals, is explained by the variables changes in revenue and the property, plant, and equipment. The error term, representing the portion of total accruals not explained by the independent variables, is interpreted as the discretionary accruals. These discretionary accruals are then interpreted as an indicator of potential earnings management (*Jones*, 1991). In addition, *Dechow et al.* develop the model further by adjusting the changes in revenues by the changes in accounts receivable, taking into account an eventual early recognition of revenues (*Dechow et al.*, 1995). Comparable to cases in which earnings management finally resulted in actual manipulations, approaches aimed at detecting earnings management or already violations of the Generally Accepted Accounting Principles (GAAP) as a first filter for subsequent further investigations (*Beneish*, 1997).

In contrast to these regression-based approaches from related fields of research, machine learning models based on more advanced algorithms could exceed human's capabilities not only because the ability to process vast amounts of data in a short time, but especially because they might be able to discover also unknown red flags, i.e., complex relationships between input and output data pointing, in this context, towards financial statement fraud but that are still

unknown to the experts' mind. This flexibility of algorithms beyond regressions, as proposed by *Fanning and Cogger* applying an artificial neural network, was considered to be a key advantage especially since there were but few confirmed theories for identifying financial statement manipulation (*Fanning & Cogger*, 1998). As a result, early machine learning approaches assumed that more flexible algorithms are better able to capture and process more complex changes and relationships between multiple accounts, which humans cannot do due to their limited capacity to absorb and process information in the sense of information overload (*Green & Choi*, 1997). However, research as early as by *Beneish* (1999a) within his so called M-Score, as briefly described in section A.1, emphasizes the importance of theoretically based variable selection to cover manifestations of manipulations or structures that favor them, which is a first indication that the successful application of machine learning is significantly influenced by human expertise.

Thus, early models are therefore characterized by certain common features. Typically, these early approaches aimed at the identification of fraudulent aspects, rather than just only on predictive classifications. *Persons*, e.g., trained stepwise logistic classification models to detect fraudulent financial reporting and identified a.o. financial leverage and capital turnover as indicative aspects for fraud (*Persons*, 1995). Besides the predictions themselves, variables identified based on these stepwise logistic regressions were also considered to be significant information for auditors and useful indicators for their audit (*Spathis*, 2002). As later also found by *Beneish* (1999b), *Summers and Sweeney* (1998), using a cascaded logit model to identify financial statement fraud, can show that, for example, the reduction of insiders' shareholdings can be an useful indicator to differentiate between fraudulent and non-fraudulent financial

reporting. These approaches led early on to rule-based expert systems that were explicitly proposed for the planning stage of audits (*Ragothaman et al.*, 1995). This shows the balance of the first financial statement fraud detection models, which on the one hand developed and proposed models to identify actual manipulations, but on the other hand were also initially inherently interpretable due to the still manageable flexibility of the algorithms used and therefore provided initial clues about driving variables.

From a technical perspective, the models were characterized by relatively small volumes of data used, which resulted in particular from a 1:1 matching of selected fraud and non-fraud cases within the training data (*Fanning & Cogger*, 1998; *Green & Choi*, 1997; *Persons*, 1995; *Summers & Sweeney*, 1998). However, this approach was criticized early on and a ratio with a larger number of non-manipulated financial statements was suggested in order to depict a more realistic setting (*Lee et al.*, 1999). In order to address challenges of small data volumes, where a strict distinction must be made between training data and test data for reasons of the predictions' robustness, approaches such as the so called Jackknife validation approach have already been used here (*Spathis et al.*, 2002). This means that one pair of observations, one fraudulent and one matched non-fraudulent observation, was excluded from the training data and a model was trained on this basis. The omitted observation then formed the test data that was used for classification. This procedure was then repeated for each pair of observations in order to fully incorporate all data into the model training so that the model could learn from each observation and its particularities. In addition to the approaches described above, which were mainly based on logistic regression, more flexible algorithms for model training were also proposed from 2000 onwards. Using these more flexible algorithms, it was shown, for example

in the form of artificial neural networks, that they are capable of outperforming logistic regression models in the identification of SEC investigations (*Feroz et al.*, 2000) or applying fuzzy neural networks (*Lin et al.*, 2003).

3.2. Major Developments

3.2.1 Amount of Data and Target Data Types

For more than two decades digitalization changed topics and methods within the field of accounting research. This has been reflected in an increasing number of corresponding publications and even new journals in the intersection of accounting and auditing research on the one hand and information systems research on the other hand (*Du & Nehmer*, 2024; *Knudsen*, 2020; *Kumar et al.*, 2020). Especially the availability of machine learning approaches led to a versatile growth of the research stream in machine learning based empirical accounting research during the last decade (*Sellhorn*, 2020). The detection of financial statement fraud, as a field of application alongside, for example, the development of bankruptcy prediction models and the creation of financial analysis models, has been the subject of considerable attention within the fields of accounting and information systems research (*Kureljusic & Karger*, 2024).

The developments and progress achieved cover several aspects of the research approaches. First, the amount and types of data have steadily grown. In line with *Gepp et al.*, the phenomenon of big data with all its dimensions and resulting challenges is also reflected in auditing, particularly in the dimensions of volume, variety and velocity in addition to veracity (*Gepp et al.*, 2018). The phenomenon of big data and its dimensions is reflected in the following aspects of machine learning-based financial statement fraud detection models, among others. While previously only financial data was used on an annual basis, approaches have now been

added that integrate quarterly data into their models and thus refer to the velocity of the data that must be incorporated by models that would actually be used and progressively trained (*Abbasi et al.*, 2012; *Hoogs et al.*, 2007). Based on approaches that in some cases only included around 30 to 50 fraud cases and 1:1 matched non-manipulated control cases (*Spathis et al.*, 2002; *Summers & Sweeney*, 1998), the trend is moving in the direction of using all available fraud cases from a jurisdiction over a period of time and also all available financial statements as a control sample in order to simulate a setting that is as realistic as possible adding up to thousands of financial statements (*Purda & Skillicorn*, 2015). The mere volume of data used refers not only to the number of financial statements themselves, but also to the different jurisdictions in which similar research approaches were implemented. While the cases mentioned before 2000 related exclusively to the US (*Beneish*, 1997, 1999a; *Green & Choi*, 1997; *Persons*, 1995; *Summers & Sweeney*, 1998), this has changed in subsequent years and has led, among others, to the finding that the performance varied depending on the data used from the different jurisdictions (*Papík & Papíková*, 2022). In particular, the following jurisdictions were included, in which there was sufficient information about actual manipulations of financial statements to train models: As Asian jurisdictions especially Taiwan (*Chen et al.*, 2017; *Chen et al.*, 2019; *Lin et al.*, 2015; *Liou*, 2008; *Pai et al.*, 2011) and China (*Bai et al.*, 2008; *Ravisankar et al.*, 2011) were subject to the training of such models; in Europe the research was primarily based on data from Greece (*Gaganis*, 2009; *Kirkos, Spathis, & Manolopoulos*, 2007) as well as the UK and Ireland (*Gaganis et al.*, 2007; *Kirkos, Spathis, Nanopoulos, & Manolopoulos*, 2007); additionally, some research models were trained using Turkish data on financial statement fraud (*Dikmen & Kücükkocaoglu*, 2010; *Ogut et al.*, 2009).

Closely related to the jurisdiction is the type of target, which is used, as not every jurisdiction publishes systematically and detailed information about occurred cases of financial statement fraud. Thus, the variety of used target variables covers, e.g., cases identified from qualified audit opinions in Greece (*Gaganis et al.*, 2007; *Kirkos, Spathis, Nanopoulos, & Manolopoulos*, 2007) and confirmed fraud cases by the Taiwanese Department of Justice in Taiwan (*Huang et al.*, 2014). However, the majority of research uses US data which is operationalized by financial statement frauds published in AAER (*Cecchini et al.*, 2010a; *Dechow et al.*, 2011; *Perols*, 2011) which are sometimes supplemented by cases identified out of the press (*Jones et al.*, 2008; *McKee*, 2009). In particular, *Dechow et al.* established a widely used common database of processed AAERs when they proposed their so-called **F-Score** (*Dechow et al.*, 2011). Additionally, restatements due to earnings manipulations (*Dikmen & Kücükkocaoglu*, 2010) are incorporated and depending on the respective research design covering both, restatement due to fraud and error as well (*Dutta et al.*, 2017). In contrast to the approaches mentioned above, there are also approaches that explicitly attempt to identify unintentional errors (*Papík & Papíková*, 2020). From this, it can already be seen from the target variable that both the volume and the variety of data used have increased significantly.

3.2.2 Amount and Variety of Features

A wider range of data types and an additional increase in volume, on the other hand, can be recognized even more clearly on the basis of the input data used. On the one hand, attempts were made to improve the performance of the models by using more different variables: The previously mentioned early US approaches comprise around 10 financial ratio features, sometimes already up to 20 features. This increased later on to around 50 financial ratios (*Hoogs*

et al., 2007; *Liou*, 2008) and, e.g., 109 financial items and ratios (*Perols et al.*, 2017). However, *Bao et al.* question the rationale behind the inclusion of additional features in the model. Their analysis of models with 28 variables revealed no significant improvement in performance when 266 additional raw financial items were incorporated, adding up to 294 raw financial data items (*Bao et al.*, 2020). On the other hand, the variety of input features increased fundamentally: Within the field of fraud detection more diverse input variables were included over the time that went beyond financial and governance variables (*Fanning & Cogger*, 1998) or additional CEO characteristics (*Schneider & Brühl*, 2023), but , e.g., also used textual data from disclosed narratives like management discussions and analysis (MD&A) (*Glancy & Yadav*, 2011; *Purda & Skillicorn*, 2015; *Zhang*, Y. *et al.*, 2022). Incorporating both, quantitative financial and textual variables in detection models can result in superior performance compared to only using one type of input data (*Cecchini et al.*, 2010b). These approaches aim to train models to recognize indicators of potential manipulation hidden in the voice, tone or readability of text from MD&As (*Goel et al.*, 2010) such as complexity or uncertainty (*Humpherys et al.*, 2011). However, incorporated textual data is not limited to published data from MD&A sections of financial reports. *Hobson et al.*, e.g., used also linguistic speech data, a.o. in the form of vocal dissonance markers from CEOs speeches during earnings conference calls associated with irregularity restatements (*Hobson et al.*, 2012), while *Larcker and Zakolyukina* finds predictive power with regard to deceptive reporting explicitly for both, the scripted formal management discussions and spontaneous statements from question and answers session of conference calls (*Larcker & Zakolyukina*, 2012). In this context, *Throckmorton et al.* show superior detection performance when combining accounting risk factors with both, acoustic and linguistic

variables (*Throckmorton et al.*, 2015). Furthermore, sentiment can be indicative for financial statement fraud as, e.g., more subjective expressions are used, higher degree of intensity with regard to sentiment expressions and more pronounced use of both, positive and negative sentiment (*Goel & Uzuner*, 2016). In addition, there are even topic modelling approaches that first identify semantically meaningful topics from 10-K narratives and then use them to improve financial misreporting detection (*Brown et al.*, 2020), as well as proposed detection models trained on mandatory initial public offering (IPO) roadshow videos in China (*Duan et al.*, 2024).

As an interim conclusion with regard to the volume of data, it can be stated that it appears to be advantageous to use a large amount of available data. On the other hand, it does not necessarily lead to superior models if the variety of data is increased across the board. Firstly, as *Bao et al.* (2020) have shown, the theory-guided selection of features already leads to a performance that is not necessarily improved by the mere addition of further variables. Secondly, this inevitably leads to greater use of resources, which does not seem possible or reasonable for every potential user, as the necessary computing capacities and times can increase enormously. And third, this may limit the applicability of such models, especially if data of selected features are not available for certain observations.

3.2.3 Variety of Algorithms

On the other hand, research addressed the issue of selecting optimal algorithms. As described in section B.2.2, the selection of the algorithm with which the model is trained must be considered against the background of the trade-off between interpretability and performance. Even if it is not universally valid for every application, a more flexible algorithm can, in

principle and especially in more complex contexts, adapt and train a model more precisely to the circumstances. However, in addition to the risk of possible overfitting, it must be taken into account that this is regularly at the expense of the interpretability of the model.

It is not only machine learning models for the widespread and initially mentioned research area of earnings management that show the use of more contemporary algorithms, such as neural networks (*Höglund*, 2012), fuzzy linear regressions (*Höglund*, 2013), or the combination of neural networks and decision trees (*Tsai & Chiou*, 2009). The range and type of algorithms used to train financial statement fraud detection models is also diverse and has continued to evolve over the last two decades. Some approaches continue to attach great importance to variable selection or the identification of driving variables while adapting to algorithms such as decision trees, backpropagation neural networks and Bayesian belief networks (*Kirkos, Spathis, & Manolopoulos*, 2007). *Cecchini et al.*, e.g., included support vector machines in addition to regressions and neural networks (*Cecchini et al.*, 2010a). Furthermore, approaches incorporate specific subtypes of algorithms, as e.g., support vector machines with different kernels (*Gaganis*, 2009) and even more advanced algorithms as fuzzy rule-based classifier based on so called evolutionary or genetic algorithms (*Alden et al.*, 2012). These types of comparisons of the performance of models trained with different algorithms is characteristic of the period and covers, e.g., comparisons of decision trees, neural networks and Bayesian belief networks (*Kirkos, Spathis, & Manolopoulos*, 2007) as well as comparisons of probabilistic or artificial neural networks and logistic regressions (*Gaganis et al.*, 2007). However, this did not lead to a uniform picture and generally valid statements about the superiority of individual algorithms in this area of application. While *Liou* trains a model based

on a logistic regression which outperforms more flexible algorithms such as decision trees and neural networks in detecting fraudulent firms in Taiwan (*Liou*, 2008), *Ragothaman and Lavin* finds differing performances depending on the type of restatement, as neural networks show superior performance in predicting revenue restatements and models trained with logit regressions achieve a higher performance in predicting non-revenue restatements (*Ragothaman & Lavin*, 2008).

Almost all of the models considered are based on supervised learning. This means that the models are trained using labelled observations. The model then learns to recognize patterns that are typical for one of the classes based on the specified and known classification. Thus, this type of learning is also known as learning by example (*Hastie et al.*, 2017). In contrast, unsupervised learning algorithms aim to identify patterns within data and classify them into heterogeneous groups, such as clustering (*James et al.*, 2021). There are very few approaches that nevertheless attempt to use unsupervised learning for classifications to detect fraudulent financial reporting (*Huang et al.*, 2014) or respectively in a first preprocessing step to optimize the selection of peer-firms for training purposes by initial clustering of firms (*Ding et al.*, 2019). However, *Tatusch et al.* can also highlight the predictive power of the clustering algorithm DBSCAN in the context of predicting restatements. By assuming that firms perform differently compared to their peers, and that clustering algorithms may therefore be a well-suited approach beside the numerous supervised learning approaches, their model can already achieve a substantial portion of its classification performance efficiently with only two or three variables (*Tatusch et al.*, 2020).

3.2.4 Technical Improvements in Learning and Validation

In the further course of time, however, the isolated use of individual algorithms is extended to combinations of models. *Song et al.*, e.g., find their ensemble model outperforming approaches based on individual algorithms and models (*Song et al.*, 2014). These ensemble models further comprise stacking, i.e. a combination of models based on different algorithms which outperform individual models (*Abbasi et al.*, 2012; *McKee*, 2009). More recent examples are combinations of neural networks or support vector machines with further decision trees or regression-based algorithms (*Jan*, 2018). Bagging or bootstrap aggregating, which is also used in financial statement fraud detection models, refers to a large number of the same type of model, typically tree-based models as, e.g., random forests, and their predictions are combined by majority vote (*Bertomeu et al.*, 2021; *Purda & Skillicorn*, 2015). Further, boosting algorithms have been applied, which subsequently train multiple but models, each of which depends on previous ones correcting for prediction errors made. These include the use of AdaBoost.M1 (*Hajek & Henriques*, 2017) and its subtype RUSBoost (*Bao et al.*, 2020; *Bertomeu et al.*, 2021).

Further, possible optimizations in the training process have evolved. In section B.3.1 the Jackknife approach is already mentioned and briefly explained. Multiple publications used the Jackknife approach for validation purposes (*Bai et al.*, 2008; *Cecchini et al.*, 2010b; *Kaminski et al.*, 2004; *Spathis et al.*, 2002). With an increasing number of observations a change towards 4-5 fold cross validation (*Ogut et al.*, 2009; *Pai et al.*, 2011), and later on a 10-fold cross validation a well-established standard procedure (*Alden et al.*, 2012; *Chen et al.*, 2017; *Goel et al.*, 2010; *Goel & Uzuner*, 2016; *Larcker & Zakolyukina*, 2012; *Song et al.*, 2014) can be

observed. These approaches, such as cross-validation or additionally an undersampling were analyzed to increase the models' performance (*Perols et al.*, 2017). In contrast, *Bao et al.* criticize the fact that the use case of financial statement fraud detection is based on intertemporal data. They therefore consider cross-validation to be inappropriate and instead use explicit periods for training and later annual slices as test data (*Bao et al.*, 2020). This is intended to take account of the realistic assessment of performance so that older fraud cases are not identified using more recent data, i.e. on the basis of data that may not have been available at the time of the fraud case itself.

3.2.5 Actual Usability and Obstacles

Besides all technical improvements one major challenge remains the usability for potential user groups. Usability is closely related to the costs associated with the various classification errors which also vary for different user groups. Early approaches considered relative misclassification costs more frequently (*Beneish*, 1997, 1999a; *Feroz et al.*, 2000; *Lin et al.*, 2003; *Persons*, 1995), however, it seems that subsequent publications focused more on slight improvements in prediction performance instead of attaching as much importance to relative costs as in the early approaches. As *Dechow et al.* (2011) state, all these approaches are critical as they offer on the one hand the potential to improve the efficiency of the capital markets, but on the other hands are costly in the case of classification errors.

In terms of misclassifications financial statement fraud detection is inherently faced with highly imbalanced data. If this is not adequately accounted for, models may tend to predict all observations as nonfraudulent and still achieve accuracy measures which might seem to be outstanding (*Perols et al.*, 2017). However, this would neglect varying costs for different

prediction outcomes and would severely limit the meaningfulness of traditional performance measures as recall or precision (*Powers*, 2011). In this context, *Zahn et al.* (2022) highlight the potential for overall costs to be reduced if the smaller group of imbalanced data, in this case fraudulent firms, are associated with higher costs in false predictions. *Beneish* (1999b, 1999a) assumes, e.g., a cost ratio for investors of false negatives to false positives of about 20:1 to 30:1. Later, *Throckmorton et al.* explicitly mentioned the number of false alarms, respectively false positives, as a prohibitive factor of using financial statement fraud detection models and suggests to lower thresholds to reduce number of false positives at the expense of an (acceptable) increase in false negatives (*Throckmorton et al.*, 2015). Therefore, an approach consisting only of non-fraudulent predictions cannot be an option from an theoretical point of view due to high costs arising from missed fraud cases and from a practical point of view, this would simply deprive the models of their raison d'être. Thus, there was a renewed focus on asymmetric misclassification costs including cost-sensitive learning (*Kim*, Y. J. *et al.*, 2016) and additional cost-related performance measures as, e.g., Normalized Discounted Cumulative Gain at the position k (NDCG@k), which aim at only selecting those observations with the highest probability of fraud, since otherwise the number of false positives would remain too high (*Bao et al.*, 2020).

Furthermore, the costs of error differ not only according to the type of error, e.g., whether a case of manipulation was overlooked (false negative) or a company was wrongly classified as fraudulent (false-positive), but different group of users, i.e. auditors, enforcement institutions, and investors, are also faced with varying misclassification costs. *Beneish and Vorst* (2022) simulate the costs of applying different classification models from the perspective

of these user groups. Their results indicate that even though many machine learning-models are highly sensitive with respect to detecting accounting manipulations, this regularly comes at the expense of numerous false-positive predictions. *Beneish and Vorst* conclude that most models cannot be used cost-effectively for most user groups, particularly due to the enormous number of false positives. To address this problem, they see two options: either to reduce the number of false positives or to make the handling of false positives and the resulting investigations more efficient (*Beneish & Vorst*, 2022). Nevertheless, these challenges have not yet been fully resolved.

3.3. Current Approaches

These previously described numerous developments, analogous to the general development of digitalization in auditing, which evolves not radically but rather incrementally (*Fotoh & Lorentzon*, 2021), illustrate the overall progress that is taking place in the field of machine-learning-based financial statement fraud detection. With regard to the further delay in the actual implementation of the models, the following discrepancy illustrates the area of tension well: Among different technologies with potential use cases in audit, the second largest gap between the assumed importance of a technology and the corresponding level of current knowledge refers to machine learning and its models (*Feliciano & Quick*, 2022). Thus, indicating a high complexity which requires improved education or a reduction of burdens, e.g., in form of additional explanations that simplify the use. However, as *Hajek* highlights, previous approaches primarily focused on accuracy, but neglected interpretability (*Hajek*, 2019). This applies in particular to potential areas of application in auditing: The lack of interpretability of AI systems, besides data availability and technical as well as human resources, is a major

difficulty in terms of implementation (*Seidenstein et al.*, 2024). Taking into account potential additional explanatory information, in terms of dealing with information overload of accounting data *Hartmann and Weißenberger* also stress the importance of decision-making based on big data and the way in which algorithm-based outputs are presented. They argue that the role of the human decision-maker will shift towards dealing with algorithms' outputs, which in turn are subject to the risks of information overload (2024). They further highlight the importance of the trade-off between additional relevant information and an information overload. In this context, explanations should be considered as relevant information, which might be able to improve decision making. In contrast, most models are limited to a simple risk classification and tend to have too little relevant information for further use and are therefore far removed from information overload.

With regard to interpretability through inherent explainability or additional explanations, initial approaches and recent improvement are discussed below. Within the context of a model based on Benford's Law, *Chakrabarty et al.* emphasize that accuracy, scope and simplicity must be considered and weighed up together. In particular, they argue that their approach is superior to others, especially for investors, because it requires few variables that are available for most cases while offering comparable performance, and because the approach is intuitive and easy to understand and thus simplifies application (*Chakrabarty et al.*, 2024). This reasoning also corresponds to the principles of Occam's Razor, in that a minimum of complexity is sought for comparable results (*Blumer et al.*, 1987).

To address this issue of an increasing demand for interpretability, one approach is to use traditional algorithms, e.g., regression analysis, which is inherently interpretable but lacks the

advantages of the machine learning-based approaches. As a compromise, *Gepp et al.* (2021) develop a contemporary ensemble model by training an independent step-wise regression model, thus deriving model coefficients for variables that might drive financial statement fraud. Other approaches take a step towards interpretability in advanced decision-tree-based models, e.g., by using post hoc feature importance to identify variables contributing to the likelihood of fraud (*Bao et al.*, 2020). *Lokanan and Sharma* also use a feature relevance measure to identify variables that make the greatest explanatory contribution and find this particularly for revenues (*Lokanan & Sharma*, 2024). Also comparable, *Vladu et al.* find increasing probabilities of manipulations in case of unusual increase in receivables, increasing leverage or decrease in sales (*Vladu et al.*, 2017). Similarly, *Purda and Skillicorn* use text data to identify the words with the greatest predictive power to detect fraud (*Purda & Skillicorn*, 2015). In addition, rules can be derived on the basis of which fraudulent patterns can be classified – depending on the values or ratios of individual financial items (*Cai & Xie*, 2024). However, all these approaches remain on a level which has previously been described in section B.2.2 as global explanations. These explanations are given independently of the individual case, and it is therefore assumed that these correlations are equally valid for all cases.

Fukas et al. go one step further in the direction of individual explanations. They implement SHapley Additive exPlanations (SHAP) illustrating individual observations and effects depending on their values but aggregating to overall effects of features (*Fukas et al.*, 2022). This makes it possible to show for which variables, e.g., higher or lower values per se contribute to a higher or lower predicted probability or whether it is rather a matter of diffuse effects, which could not be readily deduced from other global explanations.

In contrast, there are local explanations (see section B.2.2). *Craja et al.* (2020) identify clear textual indicators for financial statement fraud in companies' MD&As by using LIME, eliciting certain phrases related to financial statement fraud to provide additional guidance, but it still remains open to what extent concrete starting points for plausibility checks or further investigations arise. *Bhattacharya and Mickovic* also use LIME and additionally BertViz in the context of text data to illustrate the identification of words within MD&As which drive financial statement fraud predictions (*Bhattacharya & Mickovic*, 2024). Concerning financial variables, *Zhang*, C. *et al.* (2022) exemplified the use of different approaches for explainable AI in the audit context, which also offered initial interpretations of individual predictions and thus enabling the identification of the main drivers for flagging and offering starting points for plausibility checks. *Lin and Gao* apply post-hoc explanations using SHAP, first, by separating training data between industries and subsequently aggregating features into groups, as e.g., profitability or liquidity. Their grouped SHAPs show, that the feature groups contribute differently to the respective prediction depending on the industries the models have been trained on (*Lin & Gao*, 2022). These recent approaches emphasize the importance of and interest in interpretable approaches in financial statement fraud detection from an academic perspective. However, these approaches are still regularly limited to exemplary explanations for a few selected observations , i.e. there is no analysis of whether the models do what they are supposed to do – namely, whether the explanations are actually able to accurately represent the manipulated areas of a financial statement as driving features of a risk assessment.

4. Summary of Research Questions

Based on the reviewed literature, the research gap within the current state of research and corresponding research questions derived are outlined as follows. Financial statement fraud remains a rare form of occupational fraud, yet it is associated with the most severe economic and reputational consequences. Various stakeholders have a strong interest in its early detection, particularly auditors, enforcement authorities, and investors, who are expected, even if not obligated – either due to regulatory requirements or economic incentives – to uncover such fraud. Failure to do so may result in substantial financial losses and reputational damage. Since the late 1990s, statistical models, and later applied models classified under machine learning, have been considered highly promising tools for detecting financial statement fraud. Despite the development of numerous models, a major challenge persists: the practical implementation of these models remains largely cost-inefficient (*Beneish & Vorst*, 2022). At this point, two fundamental options exist: First, further improving model performance, for instance, by enhancing the data foundation or optimizing training methodologies to a sufficient degree. Or second, enhancing the interpretability of predictions to make individual model outputs more comprehensible and actionable for potential users (*Zhang*, C. *et al.*, 2022). In recent literature, interpretable machine learning approaches have been proposed to address this challenge in financial statement fraud detection.

Building on these considerations, the first conceptual research questions focus on whether there is a demand for more interpretable predictions in financial statement fraud detection models. As discussed in more detail in section C, the field of application of financial statement fraud detection models is surrounded by a complex and highly regulated

environment. This is particularly the case for auditors and enforcement authorities. From the perspective of professional practice, the potential use of artificial intelligence or machine learning systems in auditing is closely tied to the requirement for transparency and explainability, as these are considered essential prerequisites for a legally compliant implementation (*Thomas et al.*, 2021). Legally required is, e.g., that an auditor gathers sufficient appropriate audit evidence (ISA 200.17), whereby requirements regarding relevance and reliability must be fulfilled (ISA 200.A32). In the context of data analysis, a specific German auditing guideline (IDW PH 9.330.3.78) explicitly states that the traceability of audit results must be ensured independently of the analytical tool used. This exemplifies the kind of regulatory requirements that potential user groups are confronted with. To this end, Research Question 1 investigates the regulatory and organizational conditions that could drive the need for interpretable model predictions among the primary user groups:

RQ1: What legal and organizational conditions drive the need for financial statement fraud predictions' interpretability?

Furthermore, beyond the formal legal and organizational frameworks, criteria such as traceability also encompass a human component – namely, the extent to which a system is perceived as understandable or its results as reliable. In this regard, a certain degree of algorithm aversion can be found especially in the context of systems of artificial intelligence in an audit setting (*Commerford et al.*, 2022). For example, *Bedué and Fritzsche* (2022) emphasize that factors such as transparency and explainability, particularly in systems based on artificial intelligence, can significantly enhance trust in these systems and in their reliability. In this context, and in conjunction with regulatory requirements – especially the assessment of

reliability – the human perception component becomes crucial. Only if users perceive a system as understandable and trustworthy an actual adoption or implementation is likely to follow, even in cases where legal permissibility is already given. Therefore, Research Question 2 considers behavioral interactions, which, in a highly regulated environment with stringent professional requirements, may play a crucial role in the practical implementation of such models.

RQ2: What behavioral interactions must be considered for effective and efficient implementation in a highly regulated setting with high professional requirements?

With regard to the proposed approaches and their technical developments, section B.3 demonstrated that increasingly large datasets and more diverse algorithms are being used to train financial statement fraud detection models. While the focus for a time was placed predominantly on improving classification performance, more recent research has seen a renewed emphasis on approaches aimed at generating more interpretable results. This shift appears necessary, as it becomes evident that truly satisfactory performance levels may remain out of reach – due, among other factors, to the inherent rarity of fraud cases, which fundamentally limits the potential of model training. If further improvements in model performance prove to be limited, enhancing interpretability becomes a crucial alternative approach. Consequently, in this thesis local explanations are examined to provide insight into individual model decisions. As initially described in section A.2, the subsequent research questions are structured around classification outcomes, moving beyond explanations for isolated selected observations (as e.g. illustrated by *Craja et al.*, 2020 and *Zhang*, C. *et al.*, 2022) to systematically investigate whether local explanations can provide a coherent and

meaningful representation of the model's decision-making mechanisms. This allows for an assessment of whether the models are indeed directed at identifying fraudulent patterns. Thus, Research Question 3 focuses on correctly identified fraud cases, examining whether local explanations can effectively highlight variables from the manipulated areas of the balance sheet or income statement as key drivers of the model's predictions, thereby providing indications for starting points of furthermore targeted investigations.

RQ3: With regard to true positive predictions, i.e., detected misstatements: Do features which are related to a certain type of misstatement contribute to the classification as being misstated?

Similarly, Research Question 4 evaluates cases where financial statement fraud was not detected to determine whether, despite incorrect classification, local explanations could still point to the manipulated areas. Even if not classified correctly, this could serve as indicative evidence for a model's inherent mechanisms and ability to point out manipulated areas of financial statements.

RQ4: With regard to false negative predictions, i.e., undetected misstatements: Despite their incorrect classification, do features which are related to a certain type of misstatement contribute to the classification as being misstated?

Finally, Research Question 5 investigates false positive classifications, i.e., non-fraudulent firms that were incorrectly classified as fraudulent. As highlighted by *Beneish and Vorst* (2022) in particular numerous false positive predictions result in high misclassification costs as, e.g., additional audit procedures would be required to address the erroneously identified risk. The

key issue here is whether patterns emerge that could indicate systematic biases in the explanations.

RQ5: With regard to false positive predictions, i.e., false alarms in the absence of an actual misstatement: Do the distributions of the explanations provide indications that potential biases influence the predictions in a way that does not align with the original training objective?

To address these research questions, Research Questions 1 and 2 are first conceptually examined in section C. Subsequently, a machine learning model is trained, and local explanations are computed, providing an analysis of Research Questions 3 to 5 in section D.

C. Requirements from Multiple Users' Perspectives

1. Research Objective on Users' Demand for Interpretable Explanations

I argue that following this avenue of research on machine learning-based financial statement fraud detection models more closely makes it necessary to address the diverging perspectives of different user groups of financial statements as well as their legal and operating environments resulting, e.g., in requiring different types of additional explanations. For example, in a highly regulated sector with outstanding professional requirements for expertise, concerns regarding trust, accountability, and efficient implementation of human-machine-interaction-based applications must be considered.

To provide a detailed analysis on this subject, in accordance with previous research I select audit firms, enforcement authorities, and investors as primary user groups of financial statement fraud detection models. For each group, I consider scientific and professional literature as well as legislation on implications arising from legal requirements or organizational and operating circumstances to answer the previously in section B.4 described research questions on legal and organizational conditions which might drive the need for financial statement fraud predictions' interpretability and the role of behavioral interactions in a highly regulated setting with high professional requirements. Thus, in section C.2, I analyze the legal and operating circumstances of auditors, enforcement institutions, and investors incorporating selected qualitative support and derive individual requirements for interpretable fraud predictions. Subsequently, the need for explanations is discussed against the background of human-machine interaction in a highly regulated field within section C.3.

2. Multi-User Demand for Predicting Financial Statement Fraud

Dechow et al. (2011) highlight that multiple user groups could benefit from effective and efficient machine learning-based fraud detection approaches. I focus on the main groups of users that primarily employ financial statement analysis in practice, which are similarly classified by *Beneish and Vorst* (2022), focusing on the legal as well as organizational environment in which the users operate. On this basis, the individual demand for interpretable models will be derived before discussing their potential in terms of human-machine interaction covering issues of trust, expertise, and accountability.

2.1. Audit

The ISA require auditors to provide reasonable assurance about whether financial statements are free from fraud or error by collecting sufficient evidence to reduce the risk of an erroneous audit opinion to an acceptably low level (ISA 200.5). Risk-oriented planning of audit procedures within a specific audit engagement is essential, where risks have to be identified and determine the audit strategy and program (ISA 300.9 & A8). ISA 315 (Rev.) therefore requires the identification and assessment of risks at both the financial statement and the assertion level while the risk assessment alone does not constitute audit evidence (ISA 315.4–5 (Rev.)). This also implies that an isolated prediction would not suffice. Instead, it must be possible to conclude the prediction's driving factors to obtain sufficient appropriate audit evidence (ISA 240.10b). This particularly applies to potential fraud, where specific indicators must be identified on which further audit procedures can be planned (ISA 240.11). As required by ISA 520.4, analytical procedures involve analyzing plausible relationships. Here, too, driving factors must be identifiable to check relationships and patterns for plausibility.

In each individual audit engagement, the objective is to obtain sufficient audit evidence that "enable[s] the auditor to draw reasonable conclusions" (ISA 200.17). Therefore, according to ISA 200.A32, relevant and reliable evidence is needed. As 'reasonable' implies, both comprehensibility of content and its formal documentation must be ensured. In this context, documentation on matters of risk and related significant professional judgment made is explicitly made essential (ISA 230.A8–A9) which guarantees that the auditor's procedures and decisions can be explained and traced by the documentation at any time. As the *Institute of Public Auditors in Germany (IDW)* specifies within an examination note, these requirements apply irrespective of the technology, i.e. any documentation of data analyses within working papers must guarantee a traceability of drawn conclusions and audit results independent of the underlying technique of an analysis tool or the data used (IDW PH 9.330.3.78).

Moreover, quality assurance standards are relevant at the audit firm level, which address risks regarding to the client structure as well as the associated decisions on acceptance, continuation, or resignation of mandates. An assessment of the integrity of management for the acceptance or continuation of a mandate is required, and there must be no information that would cast doubt on this (ISQC 1.26). In Germany, it is specified that liability risks or risks of loss of reputation must be explicitly considered (IDW QS 1.72), including aspects such as aggressive accounting practices, for which explicit individual drivers must be identifiable (IDW QS 1.74).

While audit firms tend to emphasize the performance of their technology in their external communications, professional associations and regulators take a more critical role. The *American Institute of Certified Public Accountants and Chartered Professional Accountants of*

Canada, e.g., directly question the ability of unexplained approaches being considered as appropriate audit evidence by asking "If the auditor cannot explain or evaluate the results from an AI audit tool, can they conclude that they have obtained sufficient, appropriate audit evidence from the AI audit tool to form an opinion?" (*AICPA & CPA Canada*, 2020, p. 7). In a similar vein, the *Canadian Public Accountability Board* refers explicitly to an "explainability risk", which must be considered when implementing advanced technologies, in particular with regard to two requirements: First, a technology's performance must be evaluated against the background of the actual fields of application with its intended purposes. And second, especially in the audit context, comprehensible documentation is required and therefore presupposes a certain degree of interpretability beyond black box approaches (*CPAB*, 2021).

Professional literature also shows similar challenges across jurisdictions which sound more restrained than the audit firms' communications. E.g., articles from the CPA Journal highlight both, the potential of machine learning-based approaches but as well limiting factors as increased documentation requirements. The latter point, still appears to be uncertain as it has not yet been conclusively clarified what audit documentation must include, in particular how the human or expert understanding can be guaranteed and professional judgment be based on contemporary analyses (*Dickey et al.*, 2019). Examples from the German practitioner audit journal WPg underline these challenges and requirements, e.g., *Marten and Harder* share the opinion, that traceability still is a major obstacle for appropriate documentation (*Marten & Harder*, 2019). Therefore, outputs must be plausibly comprehensible and be put into professional context for a meaningful interpretation of results, and thus, before an actual implementation into decision making processes can be achieved (*Thomas et al.*, 2021).

Otherwise, auditors could run the risk of following non-transparent and misleading outputs (*Rapp & Pampel*, 2021).

In addition to the legal perspective, the organizational circumstances of audit firms must be taken into account. As the quality standards stipulate, decisions on the acceptance, continuation or resignation of mandates must be carefully considered and incorporated into client portfolio risk management. According to *Johnstone and Bedard* (2004), the risk of financial statement fraud is more important than insolvency risks regarding potential liability risks in terms of client portfolio management. Inconsistent results suggest, first, that audit firms generally tend to reject risky engagements rather than actively respond to risk with higher fees (*Johnstone*, 2000), and second, that in case of continuation of a mandate and a higher risk, e.g., for earnings management, higher fees are first enforced as long as the risk is acceptable. Otherwise the mandate is resigned (*Krishnan et al.*, 2013). This risk avoidance can be moderated through expertise. Provided that specialists are available to respond appropriately to the risks, audit firms are more willing to perform audits even under increased risk (*Johnstone & Bedard*, 2003). However, in the extreme case, this development is threatened by market failure due to adverse selection if risk exceeds an acceptable level (*Akerlof*, 1970). That this is not just a theoretical threat is illustrated by the recent case of the Adler Group in Germany, which is no longer able to find an auditor after BaFin announced detected errors (*Bender et al.*, 2022).

Overall, the legal requirements in the context of an individual audit engagement appear to be uniform. The central area of application of the models presented is risk identification and assessment. The various standards require a high degree of traceability here. It must be clearly

documented whether and explicitly which risks have been identified, and how these have been addressed by subsequent audit procedures. The same requirements for interpretability arise from quality standards at the more abstract level of the audit firm, primarily to support fundamental decisions on the acceptance or continuation of mandates. From a client portfolio management's perspective, a general fraud risk prediction can offer a first insight into the overall risk structure. Interpretable predictions on a global level, i.e. identification of driving factors over all firms considered by the model, do not provide sufficient insights for none of the use cases since they only describe which variables contribute to the model's predictions in general but neglect the clients' individual risk profile. In order to address risk properly by deploying specialists in a targeted manner, explicit starting points, and thus locally interpretable predictions, are required.

2.2. Enforcement

Despite all regulatory adjustments and advanced techniques, audited statements may still contain errors or even deliberate manipulations. Therefore, unqualified audit opinions should not provide false assurance, as there may be a residual risk of manipulation (*ESMA*, 2020). To analyze the conditions for enforcement, fundamental European regulations are used and examined based on exemplary implementation in Germany. To this end, Art. 24 of the Directive 2004/109/EC delegates the competence to carry out financial reporting enforcement to national authorities, e.g., the German BaFin.

ESMA provides guidelines for enforcement design, so that a minimum of generalizability is given. The guideline's core combines risk-oriented and random selection of firms to be audited (*ESMA*, 2020). The random selection ensures that each company is audited

at least once within a certain period of time. In German, the risk-oriented selection distinguishes between concrete and abstract risk. Concrete risk presupposes specific indications of possible misstatements and results in ad hoc examinations. The abstract risk-based selection is more general and is intended to ensure that risky companies, albeit without explicit indications, are audited with a higher probability (*FREP*, 2018). Additionally, BaFin only discloses the use of IT-supported market monitoring and largely automated media analysis for risk assessment. Due to confidentiality obligations, the technical design's performance or interpretability remains unclear (*Hanenberg & Kostjutschenkow*, 2021).

Although BaFin does not communicate detailed information about its own systems, it is possible to observe the requirements that BaFin places on risk monitoring systems used by the companies it supervises. It is clear from this, as published in the BaFin Journal, BaFin is aware of these challenges and addresses the lack of explainability of models and considering the explainability of machine learning methods to be a key criterion for a successful use and implementation (*Fahrenwaldt & Nohl*, 2022). One approach which is explicitly recognized for systems of companies supervised by BaFin is the use of interpretable machine learning for validation purposes before implementation and during operation of machine learning models (*BaFin*, 2022).

Enforcement authorities fundamentally differ from audit firms as they are not profit-oriented, do not bear any potential financial losses, and employees can only be personally prosecuted in the case of at least grossly negligent behavior. But they are virtual trustees for the reputation, i.e., trust and efficiency of the capital market within a jurisdiction. According to *Ewert and Wagenhofer* (2019), more vigorous enforcement can lead to improved quality of

financial reporting, e.g., due to reduced earnings management. But stricter enforcement can also have a preventive effect in other areas. The areas have in common that stricter enforcement brings a higher probability of detection and is anticipated by the companies to reduce their misconduct (*Shimshack & Ward*, 2005).

From the regulatory perspective, for identifying abstract risk candidates a simple prediction without any explanation might suffice. But a high false-positive rate could also make the use impracticable if too many companies are classified as risky. In contrast, a simple risk prediction is insufficient when assessing the concrete risk for subsequent ad hoc examinations. This applies equally, taking limited resources into account, which should be allocated as targeted as possible to maintain a high reputation of a jurisdiction's capital market. For these purposes, specific indications must be identifiable on the firm level. I.e., locally interpretable predictions are required for plausibility checks by human experts or offering starting points for further investigations.

2.3. Investors

Unlike auditors and enforcement authorities, investors operate in an environment that does not legally expect them to identify accounting manipulations. Therefore, the focus is more on economic aspects than legal framework conditions.

Research on non-professional investors shows that awareness and inclusion of the fraud risk assessment are associated with better overall returns (*Brazel et al.*, 2015). Overall returns can, firstly, be reduced by direct losses due to investments in fraudulent firms, and secondly, by foregone profits due to investments that have not been made because of false-positive fraud predictions (*Beneish & Vorst*, 2022). To evaluate fraud risk, one approach assesses the

credibility of management's disclosure by checking the disclosure's inherent plausibility (*Mercer*, 2004). In contrast to non-professional investors, institutional investors have more resources and expertise to assess these risks and take a significantly stronger position towards a company. When institutional investors are distracted, this might, e.g., lead to increased governance risks (*Liu et al.*, 2020). Therefore, especially institutional investors should intensively monitor their current and potential investments.

Instead of avoiding risks, potentially fraudulent firms can be identified as short-selling targets. *Massa et al.* (2015) suggest by titling, "the invisible hand of short selling" can prevent risk due to anticipation and reduced earnings management. In addition to reducing earnings management, short selling can also contribute to detecting fraud (*Fang et al.*, 2016). According to *Karpoff and Lou* (2010), the added value of short selling lies in an earlier detection and price correction closer to the fundamental value.

Overall, demand is more heterogeneous. For some non-professional investors, simple fraud risk predictions might be sufficient. This is in line with *Beneish and Vorst*'s (2022) findings, suggesting that some models might be appropriate for cost-efficient risk assessment under very narrow assumptions. Further requirements can be concluded, especially for institutional investors. Locally interpretable predictions can be used for three purposes. First in terms of risk management, more detailed explanations offer additional guidance that helps to differentiate between fraud and non-fraud cases. The statement's plausibility can be better evaluated by guided human expertise and thus improve investors' decisions. Second, short sellers can improve their identification of potential targets, if locally interpretable predictions do not provide a plausible explanation other than for potential fraud. Third, only locally

interpretable predictions allow detailed investigations on how models make accurate predictions. It is essential to rule out the possibility that, e.g., high revenues or profit figures per se lead to a fraud prediction, as fraudulent firms generally tend to overstate these figures.

3. Impact of Accountability and Domain Expertise on Trust and Implementability

As machine learning-based financial statement fraud detection is affected by technical and legal conditions, the key success factor is an effective human-machine interaction. As behavioral components are crucial, the demand for interpretable predictions therefore interacts with accountability, expertise, and trust in this highly regulated setting.

First, trust in the underlying functionality is a necessary condition for the use of any technical device. As research suggests, auditors tend to rely more on humans than on machine learning models in case of contradictory information (*Commerford et al.*, 2022). This indicates that existing models, taken in isolation, are rather unsuitable for actual deployment. Therefore, ways must be adapted to increase trust. In this vein, *Glikson and Woolley* (2020) argue that transparency and reliability interconnect. Increased transparency can contribute in two ways: in the evaluation of a model and in its actual application.

Alongside traditional prediction performance measures, post hoc analyses of interpretable machine learning can significantly supplement a model's evaluation. Accuracy measures provide information about whether the model correctly predicts test data. However, models can be biased and might not adequately adapt to future changes or unknown cases. Therefore, it is of key interest to examine the models' inherent mechanisms. Evaluating these mechanisms especially enables an examination of possible biases. E.g., a model which correctly

identifies fraudulent firms but is characterized by a high false-positive rate requires further investigations. It must be ruled out that false positives are simply driven, for example, by high sales or high-profit levels but are not able to target manipulated accounts. If investment strategies otherwise falsely exclude these profitable and growth companies on a large scale, i.e., forgone profits, it would make these models inapplicable.

For this detailed examination variables' weights in a global scope, as estimators of a regression, would only offer first insights how a model overall might work. *Bao et al.* (2020), e.g., incorporate feature importance in supplementary analyses to compare the highest scores with most frequently manipulated accounts. However, this approach cannot ensure that the corresponding variable also drove the prediction of a given manipulation. As fraud cases are more complex and could cover opposing effects of different types of manipulations, the expressive power of global explanations is significantly limited. In contrast, approaches offering locally interpretable predictions allow for each individual company to understand why a specific fraud risk was stated. The identification of contributing features for individual predictions can subsequently be compared to actual manipulated accounts. If a model succeeds in indicating manipulated accounts, this will offer supportive indication for a model's quality in detecting fraud and thereby increase trust in a model even before the operative application. Conversely, any significant biases identified could rightly point to a lack of fit of a model which could justifiably entail a lower level of trust or even warn against the use of inappropriate models.

In addition to the previously discussed complementary dimension of evaluation, transparency of systems can further enhance trust during the operative decision making

processes (*Mercado et al.*, 2016). Local explanations have a higher degree of transparency, since not only global mechanisms are considered, but explanations at the level of the individual company are made possible. Thus, users can observe, at least in a simplified way, how a model arrived at a certain prediction outcome.

Second, primary users operate in a highly regulated setting when assessing fraud risk. Even if sufficient technical trust is given, it is questionable whether legal certainty exists or whether the lack of it inhibits models' use. According to *Bedué and Fritzsche* (2022) public or legal accountability raises concerns in AI applications. This is likely to be the case in highly regulated environments. As *Beneish and Vorst* (2022) indicate, litigation risks could, e.g., arise in case of positive fraud predictions in previous years if the fraud has not been discovered immediately.

Regarding legal certainty for auditors, audit standards are deliberately formulated in a technology-independent manner to cover a wide range of future developments. However, requirements for traceability and documentation are also imposed on all applications, irrespective of the technology. The extent to which interpretable machine learning can ultimately guarantee legal certainty cannot be conclusively clarified. It is conceivable that the standards could be concretized to explicitly cover these kinds of technologies to ensure legal certainty. Otherwise, it remains to be seen how courts or professional regulators would decide in proceedings. Technically it can be stated that globally interpretable explanations, e.g., feature importance, are rather unsuitable for legal justifications, especially for further audit planning to address risks, because they try to explain the effects of the model as a whole but neglect the particularities and possibly diverging effects with respect to individual observations. However,

local interpretable predictions explicitly highlight why a certain prediction has been made. Clear documentation on the prediction, its driving factors, and how risks are addressed, e.g., by further audit procedures, might reduce the risk of personal liability (*Krieger et al.*, 2021).

Third, the users' expertise is indispensable. Technical expertise in machine learning, along with transparency, and liability, is a driver of trust in applications (*Bedué & Fritzsche*, 2022). The domain expertise seems to have more complex effects. *Bayer et al.* (2022) show that domain expertise alone has a negative effect on the intention to trust. In contrast, if the level of expertise is high, additional explanations can increase the trusting intention. In this context, *Dikmen and Burns* (2022) point out the danger that additional explanations could convey false certainty and that predictions are followed uncritically. To mitigate the risk, they point out the need for accurate interpretation of the explanations, which requires professional domain knowledge. *Shin* (2021) shows that explanations are helpful for trust building but that a comprehensible possible causality can additionally increase emotional confidence in the application. This assessment is explicitly part of the main task of applying financial statement fraud detection models: It is not a matter of blindly following predictions but of questioning them and their drivers, checking their plausibility, and, if necessary, initiating further investigations. This is particularly necessary when developments could not yet be learned from models. Since both, data on financial statements and fraudulent firms, are available with a time lag, time-lagged learning is unavoidable to a certain extent. Therefore, human expertise incorporating recent developments of a company's business environment is essential in addition to the system.

As illustrated and concluded in Table C-1, especially in this use case of machine learning models in a highly regulated context, human-machine interaction is crucial. It is essential to properly assess the potential and the limits of a model with technical know-how to be able to place a basic level of trust in it. Appropriate domain expertise and critical consideration of possible liability risks complement the assessment of an application's abilities. As *Liu* (2022) concludes, in an adjacent application area, machine learning applications should be used as complementary guidance, which outperform humans in terms of covering large amounts of hard information to identify conspicuities. The true potential is raised when humans with expertise incorporate additional, partly soft, information to make the most comprehensive assessment, which neither the machine nor the human alone would have been able to do.

Table C-1: Framework on the demand for interpretable accounting fraud predictions

		Legal Conditions	Organizational Conditions	→	Need for Explanation	Behavioral Implications
Audit	Engagement	Highly regulated ▪ risk assessment for planning of audit program and strategy ▪ detailed documentation of audit procedures and reasonable conclusions	Client acceptance, continuation or rejection decisions ▪ acceptable risk (covered by increased effort/higher fees) vs. unacceptable risk (rejection)		▪ local interpretability is required for risk-oriented efficient and effective audits ▪ documented audit procedures and judgments must be comprehensible	Trust ▪ necessary and can be increased by transparency Accountability ▪ personal liability can prevent deployment
	Audit Firm Level	Quality assurance with respect to ▪ clients' structure's risk ▪ assessment of integrity ▪ liability and reputational risks	Client portfolio management ▪ adequate audit of all mandates must be ensured ▪ cluster risk, time & personnel resource constraints		▪ local interpretability required ▪ identify potential indicators for cluster risks in the overall client structure ▪ enable checks for plausibility	▪ legally compliant use must be ensured Expertise ▪ enables appropriate critical questioning of explanations
Enforcement	Concrete Risk	Must be investigated ▪ specific indication of possible accounting fraud ▪ leads to ad hoc examinations	Effective selection ▪ missed fraud cases lead to reputational damages and ▪ financial losses for various market participants		▪ specific indication requires local interpretability ▪ explicit starting points for further investigations or plausibility checks	Trust ▪ necessary and can be increased by transparency Accountability ▪ justification in case of legal or technical supervision
	Abstract Risk	Should be investigated ▪ ad hoc or sampling examinations ▪ should be audited with a higher probability	Efficient selection ▪ limited resources must be used accurate and focused ▪ increased risk orientation to improve sample quality		▪ risk prediction might suffice ▪ interpretable predictions allow to increase the proportion of ad hoc-examinations	Expertise ▪ domain knowledge might improve the differentiation between fraud and non-fraud firms in the case of abstract conspicuities
Investors	Non-Professional	In general, no legal justification required	Simple investments ▪ diversification ▪ avoid accounting fraud cases vs. risk of foregone profits		▪ no explanation is needed ▪ limited expertise and resources reduce the potential for further analysis ▪ few approaches cost-efficient	Trust ▪ necessary and can be increased by transparency ▪ improved through detailed evaluation that can rule out potential biases which would lead to forgone profits
	Professional	In general, no legal justification required	Portfolio risk management ▪ avoid accounting fraud cases vs. risk of foregone profits Short selling ▪ identify potential targets		▪ local interpretability required ▪ check for plausibility (avoid profit forgone) ▪ initial points to investigate (avoid or target fraud firms)	Expertise ▪ enables appropriate critical questioning of explanations

4. Discussion and Conceptual Findings

The previous section identifies factors that inhibit implementing machine learning-based financial statement fraud detection models for relevant user groups. Specifically, the perspectives of auditors, enforcement institutions, and investors are considered. Factors are classified as legal or arising from organizational and operating conditions. Particular emphasis has been put on discussing further behavioral implications resulting from the highly regulated setting and outstanding requirements for professional expertise.

The first research question addresses legal and organizational conditions which could drive the need for more interpretable financial statement fraud predictions. Despite intense research on enhancements of financial statement fraud detection models, actual use does not yet appear to have been equally implemented. Beyond the actual performance of the models, there are regulatory and organizational considerations that must be taken into account. While these may not necessarily be addressed solely through improved performance, they could potentially be mitigated by enhanced transparency and interpretability. Against this background of the first research question, the analysis provides reasons to assume a significant demand for more transparent models. For auditors and enforcement institutions, transparency of their procedures is a regulatory requirement. This especially intensifies the demand for auditors for interpretable approaches and associated legally compliant documentation. From a business perspective of audit firms, client portfolio management could be improved by more transparent risk assessments. From a national perspective and its enforcement, explainable predictions could enable more efficient allocations of resources. Intensified risk-oriented selections could result in a higher reputation of capital markets, thus increasing trust and overall market

efficiency. Investors, on the other hand, are in general not affected by legal obligations. Their need for explanations results from other reasons. Institutional investors could, e.g., benefit from additional and transparent fraud risk assessments to reduce financial losses. However, forgone profits from avoiding investments in false-positive predicted firms might be prevented if conspicuity can be checked for plausibility with professional judgment.

The second research question addresses possible behavioral interactions which must be considered for effective and efficient implementation in a highly regulated setting with high professional requirements. The question specifically focuses on behavioral interactions that extend beyond regulatory and organizational frameworks from a human perspective. It aims to highlight the additional behavioral factors that financial statement fraud detection models must account for in their results. Regarding this second research question, behavioral interactions between accountability, expertise, and trust must be appropriately considered. In general, technical know-how can increase trust in technology. For auditors, it must be ensured that, with appropriate documentation, they can justify conclusions from machine learning-based predictions with legal certainty. Otherwise, even though there might be sufficient trust in technology's ability, non-compliance would still prevent its use. In contrast, domain know-how can reduce confidence in technology due to stronger questioning of its abilities compared to human expertise. Since these application cases are in a setting of high professional expertise, transparency in the form of additional explanations is required to reverse this effect.

To conclude, this section offers a conceptual framework on conditions and behavioral requirements of users' needs for an explanation of machine learning models. The findings can serve as a basis for future applications that focus more intensely on the human-machine

interaction. Transparent predictions could offer insights into which variables contributed to the prediction. These indicators can subsequently be evaluated by human experts in terms of plausibility. On the one hand, this might increase trust in machine learning applications. On the other hand, potential weaknesses could be identified, and areas pointed out where a critical basic attitude towards the application seems reasonable.

D. Local Explanations on Financial Statement Fraud Predictions

1. Overview of the Training and Analysis Procedure

In section A.3, the proposed approach of this thesis is contextualized within the Design Science Approach. The problem identification and motivation are subsequently derived from the literature review. The key challenge of previously proposed models for financial statement fraud detection based on publicly available data is that a cost-efficient application is feasible only in exceptional cases. Consequently, two potential further developments exist to make such approaches more practical: Either classification performance is further improved, or predictions are made more interpretable and thus better manageable. This thesis focuses on analyzing the second alternative. In section C, a conceptual analysis of the regulatory and organizational conditions of key potential user groups demonstrates that a viable solution in most scenarios, assuming otherwise constant classification performance, would require local explanations of model predictions. Building on this, the initial objective is to train a contemporary and performant model for financial statement fraud detection. However, the resulting predictions should extend beyond only classifying firms as fraudulent or non-fraudulent. To be truly useful for experts, predictions must allow conclusions to be drawn about which specific areas within the balance sheet or income statement drive a fraud risk classification for an individual observation. Only with such insights can human experts assess whether the key financial statement items contributing to a classification are plausible or indicate anomalies that require further investigation.

Therefore, this section focuses on the development of a model, including the generation of local explanations, as a so-called artifact in the Design Science Approach. An artifact can take various technical forms, including models that explicitly contribute to solving an identified problem (*Peffers et al.*, 2007). This artifact seeks to address the identified problem by providing local explanations alongside model predictions, offering insights into which financial statement positions drive the classification as potentially fraudulent or non-fraudulent. The explanations are first demonstrated using selected individual cases (see section D.5) before a comprehensive analysis and evaluation of the local explanations is conducted in section D.6. From a technical point of view, the approach does not differ materially from the widely used Cross Industry Standard Process for Data Mining (CRISP-DM) approach. This approach describes a comprehensive process model for data mining projects covering six phases of business and data understanding, data preparation, modeling, evaluation and deployment (*Wirth & Hipp*, 2000), which in turn has an even more practice-oriented focus and does not claim such a high degree of scientific rigor. This is mainly due to the embedding in scientific literature, as especially conducted in section B, and less to technical process steps. In contrast from a broader and theoretical point of view, the research approach applied can be categorized according to the framework's dimensions by *Booker et al.* as a classification task, applying supervised learning, as most classification models in accounting literature, and being based on inductive reasoning as being favored and driven by technological progress (*Booker et al.*, 2024).

For better contextualization of the approach, I will briefly summarize the subsequent procedure. The models trained in the following sections and used as the basis for the explanations are primarily based on the model proposed by *Bao et al.* (2020), which has gained

wide acceptance in literature. This includes, in particular, the underlying algorithm. However, several modifications have been made, justified by factors such as the use of different software, which allows for adjustments in model parameters, as well as the application of different training horizons. The first step involves the dataset description, including the financial items from COMPUSTAT and the AAER data from *Dechow et al.* (2011), as well as the selection of variables and their technical preprocessing. In section D.3.3, the actual model training is conducted. Two fundamental model variations are trained in parallel: One using normalized data and one using non-normalized data. Beyond these two core alternatives, an additional variation is introduced: The models are trained once using all available historical data and once using a rolling 10-year training window (as described in section D.3.3.2). Following this, the hyperparameter tuning process is described. The tree-based algorithm used in this study includes parameters that must be optimized for the specific application. The relevant parameters and their selection are presented in section D.3.3.3.

To evaluate the trained models as intermediate results, particularly to assess whether classification performance comparable to that of *Bao et al.* (2020) is achieved, the trained model is benchmarked against selected reference models in an analogous manner. Up to this point, the models are not explicitly cost-sensitive, as exclusively a threshold-independent performance metric is used to compare for a model's classification ability in general. Building on the discussion regarding the importance of misclassification costs, section D.3.4 focuses on determining cost-sensitive thresholds, which decide at what point a model finally classifies an observation as fraudulent or non-fraudulent. After determining these thresholds, four classification models are established, each explicitly distinguishing between fraudulent and

non-fraudulent cases. The variations result from the use of either normalized or non-normalized data and the application of rolling 10-year training windows versus all available previous historical data. The classification performance of each of these four model variants is then presented and discussed in detail. Based on the classification performance results, I have decided to continue the further analyses only with the two variants trained on a rolling 10-year training window.

In section D.5, the two applied interpretability methods, LIME and Shapley Values, are described. Section D.5.4 then illustrates the local explanations using two selected observations – one serving as a positive example and the other as a negative example. Taking into consideration that two model variants are used as the basis for generating the explanations, this results in four sets of explanations – one generated by LIME and one by Shapley Values for each of the two model variants. Building on this, a comprehensive analysis of all classification errors as well as correctly classified fraud cases follows. This includes an initial descriptive analysis of the classification results, focusing on the type and complexity of the misstatements. To evaluate the explanations, a matching process is conducted between the identified misstatement types and the financial data items that could be affected by each specific misstatement type. This step is crucial for assessing whether the explanatory features provide reliable indicators of the manipulated areas within the financial statements. To enable a direct comparison between the local explanations provided by LIME and Shapley Values – two approaches that are not inherently comparable – the explanations are transformed into a ranking of the most influential features (section D.6.2). This ranking facilitates a comparison between the two interpretability methods. The presentation of the explanations is concluded in two steps.

First, classifications of actual fraud cases are analyzed, including the true positives and false negatives, where a direct assignment between the explanatory features and the type of misstatement is possible. This is followed by an analysis of the false positives to identify potential patterns that may indicate biases within the models or the explanation methods.

2. Data Sample

Several considerations need to be made when operationalizing financial statement fraud. While some research focuses on financial reporting reliability, operationalized by restatements, highlighting the potential of not only using material errors, but also immaterial errors which occur more frequently and with less severe consequences (*Choudhary et al.*, 2021), in contrast, it is inherent to financial statement fraud that rare cases are considered with all the more serious consequences. In this regard, *Hennes et al.* emphasize the importance of differentiating between different categories of erroneous financial reporting. They find improved power of statistical tests by strictly separating between misstatements resulting from irregularities, i.e. intentional misstatements, and errors, i.e. unintentional misstatements (*Hennes et al.*, 2008). Therefore, the focus should be exclusively on a field of financial statement fraud that is as narrowly operationalized as possible. With regard to events with severe consequences, the Center for Financial Reporting and Management (CFRM) database is found to best in capturing value-relevant events and providing a relatively good base for research questions that require a full account of misconduct (*Karpoff et al.*, 2017). Today, this database is provided by *Dechow et al.* from the USC Marshall School of Business and USC Leventhal School of Accounting and initially provided introducing their F-Score predicting material accounting misstatements (*Dechow et al.*, 2011). Several reasons support the use of this database as a proxy for financial

statement fraud but also point out the limits (*Dechow et al.*, 2011, p. 25): First, a consistent methodology and common data source compared to other research is achieved, and second, the AAERs are "likely to capture a group of economically significant manipulations", which to detect is the major objective of the approach provided in this thesis; on the other hand, only cases actually detected by the SEC are contained. However, as stated above, many of those that are economically significant shall be covered. Not least because those cases were identified based on multiple reasons, including surveillance programs of stock exchanges, public complaints and hints, indication from financial press as well as SEC's reviews of 1933 and 1934 Securities Acts filings (*Feroz et al.*, 1991). Thus to the best of my knowledge, the database[2] of selected AAERs by *Dechow et al.* provide the most reliable proxy available for the research on financial statement fraud. The database covers the SEC's AAERs number 1 to 4,278 and thereof 1,816 firm misstatement events. Besides general information on the firm name, identifiers and affected periods, information on balance sheet and/or income statement accounts affected by the violation is also included.

The AAER data is matched with the financial data from COMPUSTAT[3]. COMPUSTAT is characterized in particular by the lowest number of unavailable data items and the highest proportion of matches between the database's items and the actually disclosed XBRL data, e.g., compared to Yahoo! Finance and Google Finance (*Boritz & No*, 2020). This makes the database

[2] The AAER dataset can be purchased on the joined project website of the USC Marshall School of Business and USC Leventhal School of Accounting (https://sites.google.com/usc.edu/aaerdataset/home). The dataset was initially described by Dechow et al. in their work "Predicting Material Accounting Misstatements" (2011) as referenced in the text. The current and updated status of the data was provided via email by the project team on April 5th, 2023.

[3] The COMPUSTAT data was retrieved via Wharton Research Data Services (WRDS) on March 17th, 2023 (https://wrds-www.wharton.upenn.edu/). Originally, selected items (see section D.3.1) were retrieved for both, active and inactive firms covering the time range from 1979 up to and including 2019.

a suitable choice. Since a majority of accounting research projects also use COMPUSTAT database, this further contributes to an increased comparability with other research. The matched dataset was filtered in several steps as shown in Table D-1. When the two data sets, the AAER financial misstatements and the COMPUSTAT financial statement data, were joined, only those observations with a GVKEY were retained. The GVKEY is the COMPUSTAT identifier, which is also part of the AAER data set. Based on this matched raw data, those with missing values for the financial data items were removed first. Companies from the finance, insurance and real estate sectors were then excluded based on the Standard Industrial Classification (SIC). Finally, the period under consideration was limited to the years from 1990 to 2019. This resulted in the final dataset, which comprises a total of 166,144 observations, including 1,015 misstated firm years.

Table D-1: Filtering steps for joined financial statement and misstatement dataset

Filtering steps and (sub)totals	**Total number of observations**	**Thereof misstatements**
Total number of observations of the joined dataset of AAER and COMPUSTAT (1979–2019)	472,306	1,809
_ Removing observations with missing values for raw financial data items	250,777	599
Subtotal	221,529	1,210
_ Removing observations from Finance, Insurance and Real Estate Industry (SIC between 6000 and 6999)	5,559	39
Subtotal	215,970	1,171
_ Filter for period between 1990 and 2019 (removing observations before 1990)	49,826	156
Total number of observations after filtering steps	**166,144**	**1,015**

The table lists the filtering steps for the dataset used. The total number of annual observations of the AAER dataset sums up to 2,195, thereof 386 observations lack a GVKEY. After matching the financial statement data from COMPUSTAT with the annual AAER data, the initial joined dataset contains 1,809 misstated firm years. Within the bottom line, there are 29 misstatements included, which refer to understatements, which will not be used for training purposes.

Table D-2 summarizes the number and percentage of observations in general and those with misstatements within the joined dataset per year. In contrast to previous research, such as *Bao et al.* (2020), the figures per year slightly differ due to deviating filter steps. The figures show that the highest proportion of financial misstatements in the overall observations was recorded in the period from around 1990 to 2010. The falling figures thereafter may be due to cases that, on the one hand, may have not yet been detected, as well as to less intensive prosecution by the SEC (*Bao et al.*, 2020). In particular, the rare cases covered by the data in recent years will make both model training and the detection of such rare cases considerably more difficult for the most recent periods. Nevertheless, these periods are explicitly included, as detection models that would only have been effective in the past, but would not be transferable to current circumstances, would only provide a limited practical contribution.

Table D-2: Frequency of financial misstatement related AAERs within the used dataset

Year	Number of firms	Number of firms with financial misstatements	Percentage of firms with financial misstatements
1979	3,670	4	0.11%
1980	3,889	9	0.23%
1981	4,358	12	0.28%
1982	4,550	20	0.44%
1983	4,828	13	0.27%
1984	4,845	14	0.29%
1985	4,764	10	0.21%
1986	4,836	20	0.41%
1987	4,851	15	0.31%
1988	4,690	18	0.38%
1989	4,545	21	0.46%
1990	4,467	16	0.36%
1991	4,574	27	0.59%
1992	4,805	26	0.54%
1993	5,222	27	0.52%
1994	5,531	20	0.36%
1995	6,045	20	0.33%
1996	6,549	32	0.49%
1997	6,593	45	0.68%
1998	6,512	56	0.86%
1999	6,609	74	1.12%
2000	6,549	90	1.37%
2001	6,181	84	1.36%
2002	5,896	78	1.32%
2003	5,809	71	1.22%
2004	5,766	62	1.08%
2005	5,693	48	0.84%
2006	5,734	31	0.54%
2007	5,704	25	0.44%
2008	5,456	19	0.35%
2009	5,215	29	0.56%
2010	5,240	25	0.48%
2011	5,244	18	0.34%
2012	5,485	25	0.46%
2013	5,504	15	0.27%
2014	5,498	17	0.31%
2015	5,252	11	0.21%
2016	4,937	12	0.24%
2017	4,804	6	0.12%
2018	4,724	4	0.08%
2019	4,546	2	0.04%
Total	**215,970**	**1,171**	**0.51%**

The table lists the number of firms and the occurrence of financial misstatements for each year. The observations comprise those, for which a matching of AAER and Compustat data is possible. Observations with missing financial variables and firms from the finance, insurance and real estate industries (SIC first digit = 6) are removed. For the final dataset only firm years from 1990 on, as separated by the dotted line, are used.

In addition to the general information on financial misstatements identified in the AAERs, *Dechow et al.* have also complied detailed information on the nature of the underlying manipulations. For the period under review, Table D-3 lists the areas and types of balance sheet or income statement items that were misstated. Overall, revenues were misstated in approximately half of all cases, as well as other expenses and/or shareholder equity accounts. While capitalized costs as assets, accounts receivable or inventories were also relatively frequently misstated, payables, allowances for bad debt or marketable securities were affected by manipulations far less often.

Table D-3: Accounts affected by financial misstatements

Account affected	Frequency	Percentage
Misstated revenue	529	52.12%
Misstatement of other expense/shareholder equity account	511	50.34%
Capitalized costs as assets	233	22.96%
Misstated accounts receivable	193	19.01%
Misstated inventory	152	14.98%
Misstated cost of goods sold	126	12.41%
Misstated liabilities	119	11.72%
Misstated reserve account	113	11.13%
Misstated payables	42	4.14%
Misstated allowance for bad debt	15	1.48%
Misstated marketable securities	9	0.89%

The table lists the account categories which are subject to manipulation. It covers the time range from 1990 to 2019, which is subject to subsequent analyses. The percentage column sums up to more than 100 % because many fraud cases affect more than one account category.

Since the impacts of manipulation do not necessarily have to be limited to a single position or area of accounts within the balance sheet or income statement, Table D-4 shows how many of the areas listed in Table D-3 are affected in each case of misstatement. In slightly less than half of the cases, the impacts are limited to one of the areas. Approximately 28 % extend to two areas of accounts, while only 14 % of the cases show misstatements in three of the areas. The

information of accounts affected by financial misstatements provides the basis for subsequent analyses on the question, if the classifications of interpretable machine learning approaches are significantly driven manipulated accounts, or, if those predictions reveal potential biases of models.

Table D-4: Number of accounts affected by financial misstatements

Number of accounts affected	Frequency	Percentage
1	465	45.81%
2	281	27.68%
3	139	13.69%
4	68	6.70%
5	46	4.53%
6	16	1.58%

The table lists the frequency and percentage of the number of affected account groups for fraud observations. It covers the time range from 1990 to 2019, which is subject to subsequent analyses.

After providing an overview of the databases used and a descriptive classification of the financial misstatements, the following section outlines which items were selected and will subsequently be included in the model training.

3. Model Training

3.1. Feature Selection

In section B.3 I provided an overview of the types and quantities of data used in models to identify financial statement fraud. The first models (see section B.3.1) attempted to select variables based on theory as far as possible. These included theory-based constructs as, e.g., the Altman Z Score, and further detailed described and theory-based inputs (*Summers & Sweeney*, 1998). Also comparable, *Beneish* incorporated selected features based on previous research on cash (*Healy*, 1985) and accruals (*Jones*, 1991). The selected variables should either capture distortions resulting from manipulations or circumstances which might encourage or prompt

manipulations of financial statements (*Beneish*, 1999a). In the following time, additional features were subsequently added into the models (see in particular section B.3.2), albeit with a seemingly decreasing significance of the underlying theory. However, this does not necessarily mean that, e.g., the selection of features by models is inherently better merely because more and more data is available. With regard to this aspect, *Bao et al.* find, that including all available data items, rather than only those, which were theoretically based selected, does not further improve a model's detection performance (*Bao et al.*, 2020). This indicates that, at least with regard to the model applied, the theory-led selection does have a relevance for the performance of a model and, in contrast, a model with all available data items does not necessarily improve its classification performance. Further, it must also be taken into account that, as shown in section C, the actual applicability of the models depends, among other things, on the degree of interpretability. In combination with a possible information overload, which must also be taken into account in the output of models (*Hartmann & Weißenberger*, 2024), a selection of all available items would counteract this and limit comprehensibility and usability. Finally, numerous algorithms can handle a large number of variables when training a model. In contrast, the processing requirements for local explanations such as LIME or Shapley Values are very computationally extensive. Particularly with a large number of variables, this means that the operations can no longer be parallelized well, but above all a considerable amount of working memory must be guaranteed. In order to keep models practicable for as wide a range of users as possible, the emphasis here is on the fact that larger computing clusters are explicitly not a necessary requirement for model training and their explanations.

Table D-5: Compustat data items used as features for model training

Abbreviation	Feature
act	Current Assets, Total
ap	Account Payable, Trade
at	Assets, Total
ceq	Common/Ordinary Equity, Total
che	Cash and Short-Term Investments
cogs	Cost of Goods Sold
csho	Common Shares Outstanding
dlc	Debt in Current Liabilities, Total
dltis	Long-Term Debt Issuance
dltt	Long-Term Debt, Total
dp	Depreciation and Amortization
ib	Income Before Extraordinary Items
invt	Inventories, Total
ivao	Investment and Advances, Other
ivst	Short-Term Investments, Total
lct	Current Liabilities, Total
lt	Liabilities, Total
ni	Net Income (Loss)
ppegt	Property, Plant and Equipment, Total
pstk	Preferred/Preference Stock (Capital), Total
re	Retained Earnings
rect	Receivables, Total
sale	Sales/Turnover (Net)
sstk	Sale of Common and Preferred Stock
txp	Income Taxes Payable
txt	Income Taxes, Total
xint	Interest and Related Expense, Total
prcc_f	Price Close, Annual, Fiscal

The table lists the 28 COMPUSTAT data items identified by Bao et al. (2020) and used for the model training.

From the aforementioned reasons, I follow *Bao et al.* (2020) and limit the features to their already theory-based selection, for which they show that expanding the scope of the features used does not increase the model's classification performance. Table D-5, which is also

attached to this thesis in Appendix F, lists these COMPUSTAT data items which I will use, in line with the feature selection by *Bao et al.* (2020), as features to train the classification models.

Table D-6: Financial ratios used for benchmark model

Abbreviation	Feature
dch_wc	WC accruals
ch_rsst	RSST accruals
dch_rec	Change in receivables
dch_inv	Change in inventory
soft_assset	% Soft assets
ch_cs	Change in cash sales
ch_cm	Change in cash margin
ch_roa	Change in return on assets
issue	Actual issuance
bm	Book-to-market ratio
dpi	Depreciation index
reoa	Retained earnings over total assets
EBIT	Earnings before interest and taxes over total assets
ch_fcf	Change in free cash flows

The table lists the 14 financial ratios including actual issuance and book-to-market as market related incentives identified by Bao et al. (2020) based on Dechow et al. (2011) and Cecchini et al. (2010a). These are solely used to replicate a benchmark model to ensure that the classification performance for the subsequently trained models is also superior for those being trained on raw financial data items compared to financial ratios.

The models trained in the following sections, used to examine the extent to which explanations are actually able to identify manipulated areas of the balance sheet or income statement, are not 1:1 replications of the *Bao et al.* model (*Bao et al.*, 2020). The individual steps are detailed below. The objective, however, is to maintain similarity in key aspects to achieve a comparably high performance based on raw financial data items. In this context, I replicate the comparison to regression-based benchmark models to assess whether the higher performance of raw financial data items over financial ratios also applies to the model trained here, which, in turn, is fundamental to its practical relevance. Thus, the financial ratios from *Dechow et al.* (2011) and *Cecchini et al.* (2010a), from which the raw financial data items were derived by *Bao et al.*

(2020), are also calculated and listed in Table D-6. These are used solely to ensure that the models trained here with raw financial data also achieve better performance than a benchmark model based on financial ratios.

3.2. Data Preprocessing

The following data preprocessing, as well as the subsequent training of models and the computation of explanations is conducted on a regular notebook. The hardware used is characterized in particular by the CPU Intel(R) Core(TM) i7-8565U and 16 GB RAM. This guarantees potential use by the widest possible range of users without having to make special demands on server infrastructure or computing clusters. I used the opensource software RStudio. The version 2023.6.1.524 was used at the time of processing and calculations. Separate additional packages are available for RStudio, particularly in the area of interpretable machine learning, which cover the necessary functions. For traceability, reference is made to the specific packages used at the relevant points and these can also be found in the code in the Appendix A till Appendix E.

The preprocessing steps of the data are conducted via the code in Appendix A. As mentioned in section D.2 with regard to the data sample used, first, the AAER and COMPUSTAT datasets were matched using the financial year and the GVKEY as the common identifier provided in both databases. I then recalculated the financial ratios described in section D.3.1 (see also Table D-6). Variables are formatted accordingly based on their nature, so that, for example, the target variable “misstatement” is treated as a factor and not as a “numeric”. Next, observations with missing values for the raw financial data items, in particular those without a value for total assets, are removed from the dataset. In some cases, special

values occur (e.g. NaN, Inf, -Inf), which can result from the calculation of financial ratios, for example, depending on which data is available for the calculation of lagged variables. These are replaced as missing values (not available, "NA") in order to prevent any calculation errors that may subsequently occur. In addition to these technical aspects, as already described in the context of the data sample, the period from 1990 to 2019 is filtered and all observations from the finance, insurance and real estate industries are excluded.

In contrast to previous research, I explicitly use two different versions of the raw financial data items for the model training. On the one hand, I use normalized features, as it is common practice, whose values are then restricted to the interval between 0 and 1. On the other hand, I also train the models with non-normalized values, i.e. with the values as they are explicitly disclosed in the financial statements. As the data preprocessing regarding a possible standardization or normalization is sometimes still considered a subjective choice, it is important to be aware that normalization can also have drawbacks, as some methods of normalization struggle with handling outliers (*Singh & Singh*, 2020). Further, *Sujon et al.* show differences in SHAP depending on if data was normalized or not (*Sujon et al.*, 2024). Thus, incorporating both normalized and non-normalized data, a potential loss of information with regard to potentially informative absolute values is addressed. This enables, first, a comparison of the classification performance of financial statement fraud detection models in a first step, and second, a comparison of the adequacy of explanations provided by post-hoc approaches.

Within the code objects based on the normalized features, e.g., datasets and trained models, are supplemented by the suffix "_norm". Those objects without this explicit suffix refer to the non-normalized features.

3.3. Algorithm and Hyperparameter Tuning

3.3.1 Algorithm Selection

The data described above is incorporated into the training of the models. First, comparable to and based on the findings by *Bao et al.* (2020) two logit benchmark models are trained. One benchmark model is trained with financial ratios and one with raw financial data items. These two logit models are trained solely for the purpose of serving as a benchmark in an intermediate step. As will be demonstrated in section D.3.3.4, and consistent with the findings of *Bao et al.* (2020), the replication conducted here confirms that RUSBoost models trained on raw financial data items yield superior classification performance compared to logit models – regardless of whether those are based on financial ratios or raw financial data. Second, the RUSBoost models that form the core of the subsequent analyses are trained. The RUSBoost models are trained in two variants, based on normalized and non-normalized data. Technically, as subsequently in section D.3.3.2 described in detail, for each test year individual models – one based on normalized, and one based on non-normalized data – are trained.

The two logit benchmark models are trained using the R inherent glm function. The main models are trained using the AdaBoost algorithm (*Freund & Schapire*, 1996). The AdaBoost algorithm is a boosting algorithm. It is characterized by the fact that a large number of weak classifiers are trained one after the other. Each of these learns from the errors made by the previous classifiers. The result is therefore an ensemble model. RUSBoost is a sub form of the AdaBoost algorithm (*Seiffert et al.*, 2010): Random UnderSampling (RUS) addresses challenges that occur with imbalanced data. Random under sampling establishes comparability between the two classes, in this case fraud and non-fraud, by reducing the class with more

observations through random selection. For reasons of compatibility, I do not choose a package with an explicit built-in RUSBoost function. These would not be fully supported by the functions of interpretable machine learning that are subsequently required. Therefore, I operationalize the RUSBoost algorithm by using the basic AdaBoost.M1 algorithm of the 'adabag' package (*Alfaro et al.*, 2013). Random undersampling, on the other hand, is manually implemented with the widely used 'caret' package and the trainControl function it contains (*Kuhn*, 2008).

3.3.2 Training, Validation and Test Periods

In the case of financial statement fraud detection models, time series data is used. This requires specific considerations to split the dataset into training, validation and test data. For purposes of hyperparameter tuning and performance evaluations of an initial model with the two benchmark models Panel A of Figure D-1 illustrates the data splits. The training data covers 10 years, starting from 1990 on till 1999. For the subsequently described hyperparameter tuning two validation periods are used, i.e. 2000 and 2001. To determine cost-efficient cut-off thresholds of the classification models I use additional unknown data from 2002.

To address the problem of overfitting in supervised learning (*Dietterich*, 1995), performance evaluation is conducted on out-of-sample test data which can reduce biases resulting from overfitting (*Clark*, 2004). As in prior research I select data from 2003 on year per year as test data, i.e. leaving out two years between training and test data since according to *Dyck et al.* (2010) a typical period of time until a case of financial statement fraud is publicly disclosed covers approximately 2 years. With a view to a realistic use case, this prevents financial statement fraud cases that have not yet been discovered at the fictitious point of time

of training from being considered as such during training. This applies to both approaches as illustrated in Panel B.1 and Panel B.2 of Figure D-1.

Accounting data is characterized by a number of special features that distinguish them from other applications of classification models. The data is not data that would always arise in the same form, e.g., as in other fields of application according to the law of nature but is the representation of business transactions according to certain man-made rules, here GAAP. This includes two aspects that need to be considered over time. First, the significance of the individual underlying transactions can change over longer periods of time. Analogous to technical progress, there is, e.g., an increasing importance of intangibles and challenges of standards to account for these (*Eckstein*, 2004). Furthermore, the type of financing, for example, can change. These changes are reflected not only in the nature of the transaction, but also in changes to the rules for recognition itself. A recent and global example of this is accounting for leases. Both, the IASB and FASB aligned their GAAP related to leases (*Biondi et al.*, 2011). Against this background, *Morales-Díaz and Zamora-Ramírez* find, that the application of IFRS 16 Leases affects key financial ratios (*Morales-Díaz & Zamora-Ramírez*, 2018). Thus, both the change of business transactions themselves as well as specific GAAP result in a change of accounting data over longer periods of time. Therefore, I address these potential changes of the underlying structure and relations within accounting by two approaches:

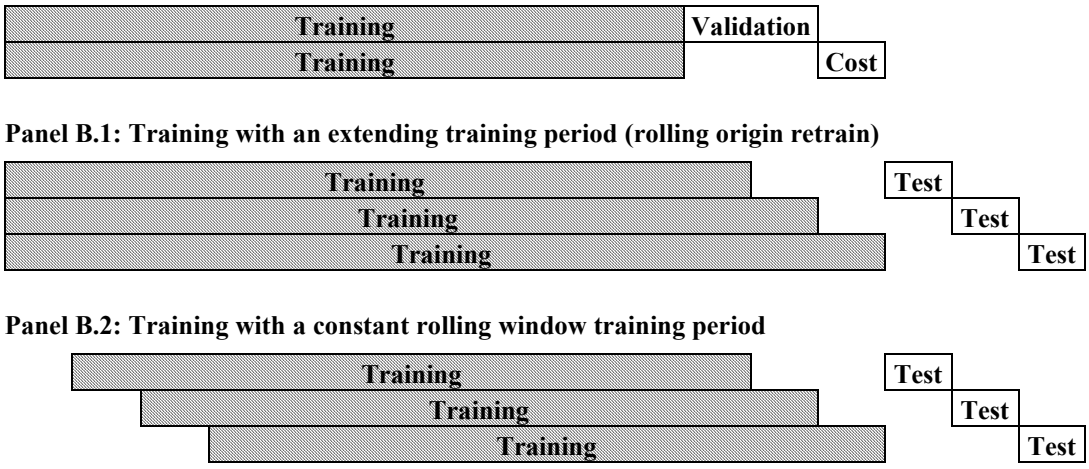
First, Panel B.1 illustrates an evaluation on a "rolling-origin-recalibration" (*Bergmeir & Benítez*, 2012). It should be made clear that the term "forecasting origin" by *Tashman* (2000) refers to the point of time of the most recent observation included in the training (here e.g. 1999) and not the oldest time used (here e.g. 1990). This first approach extends the time and amount

of data incorporated into the training of models. As this procedure is also referred to as a "rolling origin retrain" (*Meisenbacher et al.*, 2022), it highlights the fact that for each step of the test period a whole new model is trained based on the extended training period. A significant advantage is that all available data, and thus also rarely occurring financial statement fraud cases, are included in their total number over long periods of time. This may be advantageous in many cases, but as described, accounting data is data that is subject to structural changes over time – due to changes of the underlying characteristics of transactions as well as to changes of GAAP, thus, the rules applied in recognizing accounting transactions. Therefore, I propose additionally applying a "rolling window evaluation" (*Bergmeir & Benítez*, 2012). A rolling window evaluation has neither a fixed forecasting origin nor a fixed starting point for the training data. Since for each year of test data a separate model is trained, this means the starting point shifts one year forward for each additional trained model. This results in training data, which covers for each model a time frame which remains equal (*Tashman*, 2000). Here, the training data comprises 10 years for each trained model. This is illustrated in Panel B.2 of Figure D-1. On the one hand, models cannot learn from financial statement fraud cases which occurred more than 12 years before the test period (10 years of training data and 2 additional years as a gap in between). The algorithm is therefore supplied with a smaller number of fraud cases on the basis of which the model can be trained. However, the practical relevance of this approach might arise due to the characteristics of accounting data. Changes in GAAP and shifts of the relevance between financial positions might disturb model's predictions over longer periods of time. Thus, I argue that not using older cases in training as a potential disadvantage

at first sight, can nevertheless lead to a more considered model in this accounting setting, as it is better suited to the current framework conditions.

Therefore, the models are subsequently trained as follows: First, the hyperparameters and the cost-efficient thresholds are validated using the time periods as shown in Panel A (Figure D-1). Subsequently, the actual RUSBoost models are trained in parallel in two versions, on the one hand in the sense of an evaluation on a "rolling origin retrain" (Panel B.1) and on the other hand as proposed on the basis of a "rolling window evaluation" (Panel B.2).

Figure D-1: Division of periods into training, validation and test data



The figure illustrates the division of the periods as training data, validation data and test data.
Panel A: For purposes of comparing a first RUSBoost model with benchmark models, for hyperparameter tuning and the determination of classification thresholds, the division of the data is selected according to the illustrated periods in Panel A. For general training the decade from 1990 to 1999 is used. Specific hyperparameters are determined by further incorporating the two consecutive years 2000 and 2001 as validation data. Based on the determined parameters, a first performance evaluation is conducted for the test year 2002 in order to find cost-efficient cutoffs as thresholds for the classification between potentially fraudulent and non-fraudulent observations.
Panel B.1: The performance evaluation is conducted for two separate training approaches. The first training approach includes all available data from 1990 onwards with the exception of the two years prior to the test year under consideration.
Panel B.2: The second approach trains the models with a constantly long rolling period of 10 years of training data, which in turn excludes the last two years before the test year.

3.3.3 Hyperparameter Tuning

Algorithms have various parameters that cannot be determined for a model directly by learning from the data during the training process, but must be defined explicitly – known as tuning parameters (*Kuhn & Johnson*, 2013). These are also referred to as hyperparameter (*Yang & Shami*, 2020). In order to select the optimal values for each hyperparameter multiple aspects might be taken into consideration. I have set the hyperparameters primarily on the basis of their contribution to the models' performances, and from a practical point of view, with certain limitations in terms of computational effort and time (*Kuhn & Johnson*, 2013). To identify a suitable set of hyperparameters I conducted a grid search. This means that a range of possible values, or specific coefficients, was first defined for the hyperparameters, which were then tried out in certain steps in the various combinations for the model. Since the number of three available hyperparameters, which can be set within the applied package, remains relatively small, the most commonly hyperparameter optimization method, a grid search, is assumed to be applicable under these circumstances (*Yang & Shami*, 2020). For this purpose, the training data comprises the years from 1990 to 1999. The performance of the various models for all possible combinations of the hyperparameters is validated using data from 2000 and 2001, which would otherwise remain unused (see Panel A of Figure D-1).

For the model training I use the AdaBoost function of the 'adabag' package (*Alfaro et al.*, 2013). Other software or packages might differ in the specific parameters which need to be set. The 'adabag' package comprises three tuning parameters for the AdaBoost algorithm. First, as AdaBoost is a boosting algorithm, i.e. classifiers subsequently learn from previously made misclassifications, it is necessary to set a learning coefficient ('coeflearn'), which constitutes

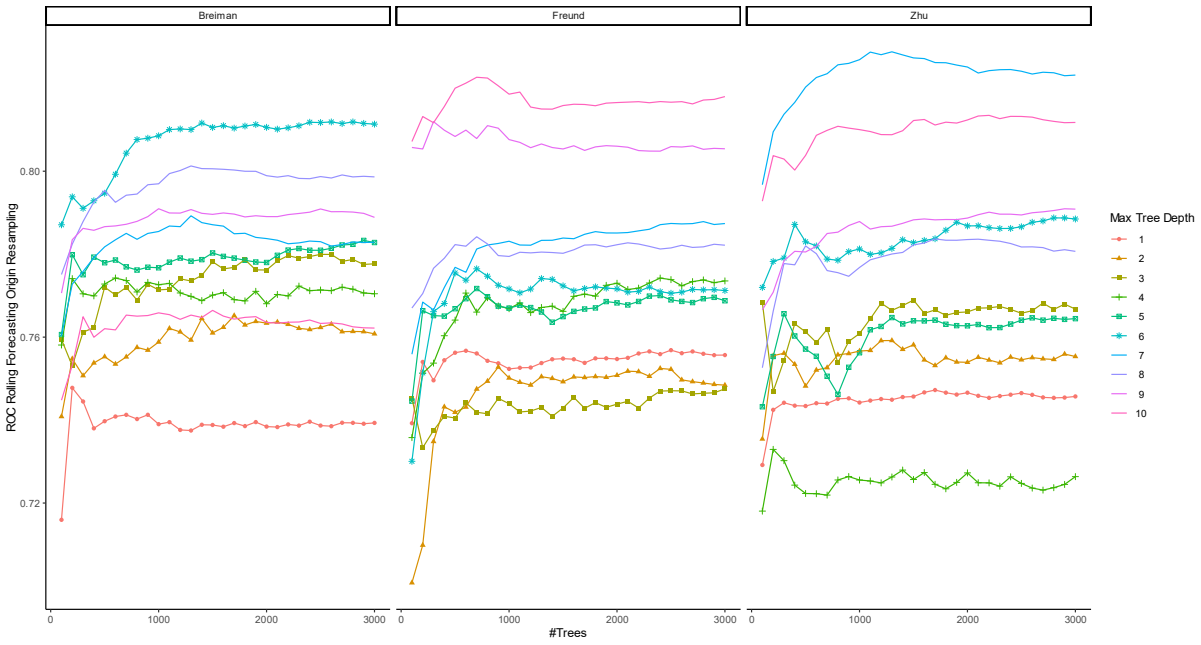
how subsequent classifiers incorporate these misclassifications and how these adjust the following classifier. These learning coefficients, also called 'weight updating coefficients', are named after the authors who proposed the respective coefficient calculations. Three different approaches of learning coefficients by Breiman, Freund, and Zhu are available and considered during the grid search. The second and third hyperparameter result from the type of algorithm, as AdaBoost is a tree-based classification algorithm. Therefore, second, the maximum depth of trees is required, i.e., the maximum depth defines the number of levels or layers of a decision tree. It specifies how many decision nodes the longest path from the root to a leaf may contain. For reasons of computational effort, I limit the interval to be tested to 1:10. Third, the number of trees which are subsequently grown must be determined ('mfinal'). I set a range of trees between 100 and 3,000 which are applied stepwise in intervals of 100. Technically the grid search is conducted using the 'caret' package and its train function and setting the parameters by the objects 'expand.grid' and 'trainControl'. This guarantees a high degree of flexibility and compatibility. I have selected 'roc' in the package as the performance measure on which the hyperparameter tuning is based (*Kuhn*, 2008). This refers to the so-called AUC, i.e. Area Under the Curve, which in turn refers to the Receiver Operating Characteristic (ROC) curve. The AUC is one of the most widely used performance metrics for binary classification tasks representing a "classifier's ability to avoid false classifications" (*Sokolova & Lapalme*, 2009, p. 430). Technically, the ROC curve represents the graph that results when the True Positive Rate (TPR, also called sensitivity) is plotted on the y-axis and the False Positive Rate (FPR, equivalent to 1 – specificity) on the x-axis. In a binary classification model, an output value is initially calculated, which is then allocated to one of the two classes based on whether it exceeds a

specified threshold. Each point on the ROC curve represents the combination of TPR and FPR corresponding to a particular threshold value. This means that the AUC serves as a general performance metric for the model, independent of a specific threshold that is later to be defined. However, the subsequent selection of a specific threshold allows for a concrete trade-off between true positives and false positives to be determined (*Fawcett*, 2006). This subsequent trade-off is discussed in section D.3.4 based on different relative costs of classification errors in the form false positives and false negatives.

The results of the grid search are illustrated in Figure D-2 and Figure D-3. The grid search validates the classification performance for all three adjustable parameters provided by the applied function, i.e. all combinations of hyperparameters, including three different learning coefficients, maximum tree depths from 1 to 10 and a stepwise increase of the number of trees from 100 to 3,000. In principle, both figures contain the same information of the conducted grid search but allow individual characteristics of the three hyperparameters to be better recognized depending on the type of illustration. First, Figure D-2 serves especially to assess a suitable number of trees. The y-axis represents the classification performance of the models, measured by the AUC. Accordingly, higher values on the y-axis indicate a better selection of hyperparameter values from the perspective of classification performance. For each of the three learning coefficients, a separate coordinate system is presented. A graph is plotted for each maximum tree depth within the range of 1 to 10. Last, the x-axis indicates the number of sequentially trained trees in a model. Across all three learning coefficients, and initially irrespective of the depth of the trees, a certain pattern is generally recognizable. Starting from 100 consecutively grown trees, an increase in the number of trees results in a strong increase in

classification performance according to the AUC in almost all constellations. This increase in performance develops in the sense of a positive but decreasing marginal benefit. Across all constellations, the performance seems to continue to increase slightly up to the range of 1,000 to 2,000 trees. in the range of 2,000 to 3,000 trees, however, a plateau formation takes place. A number of trees beyond this no longer seems to be able to make a major contribution to better classification performances.

Figure D-2: Hyperparameter tuning of RUSBoost model (1/2)



The figure illustrates the hyperparameter tuning for the RUSBoost model based on the model trained to finally detect misstatements in 2003. As illustrated in Figure D-1, Panel A, the underlying model is trained on data from 1990 to 1999 and for validation purposes data from 2000 and 2001 is used. The performance is shown on the y-axis using the AUC, which the applied package labels as ROC. The hyperparameter tuning covers three parameters. First, each of the three coordinate systems illustrates a different learning coefficient ('Breiman', 'Freund', and 'Zhu'), set to control the way in which consecutively grown trees learn from previous errors. Second, within each coordinate system each graph illustrates a different level for the maximum depth of a single grown tree, covering a range of the maximum depth between 1 and 10. Third, on the x-axis, the number of consecutively grown trees is stepwise validated for a range between 100 and 3,000 trees.

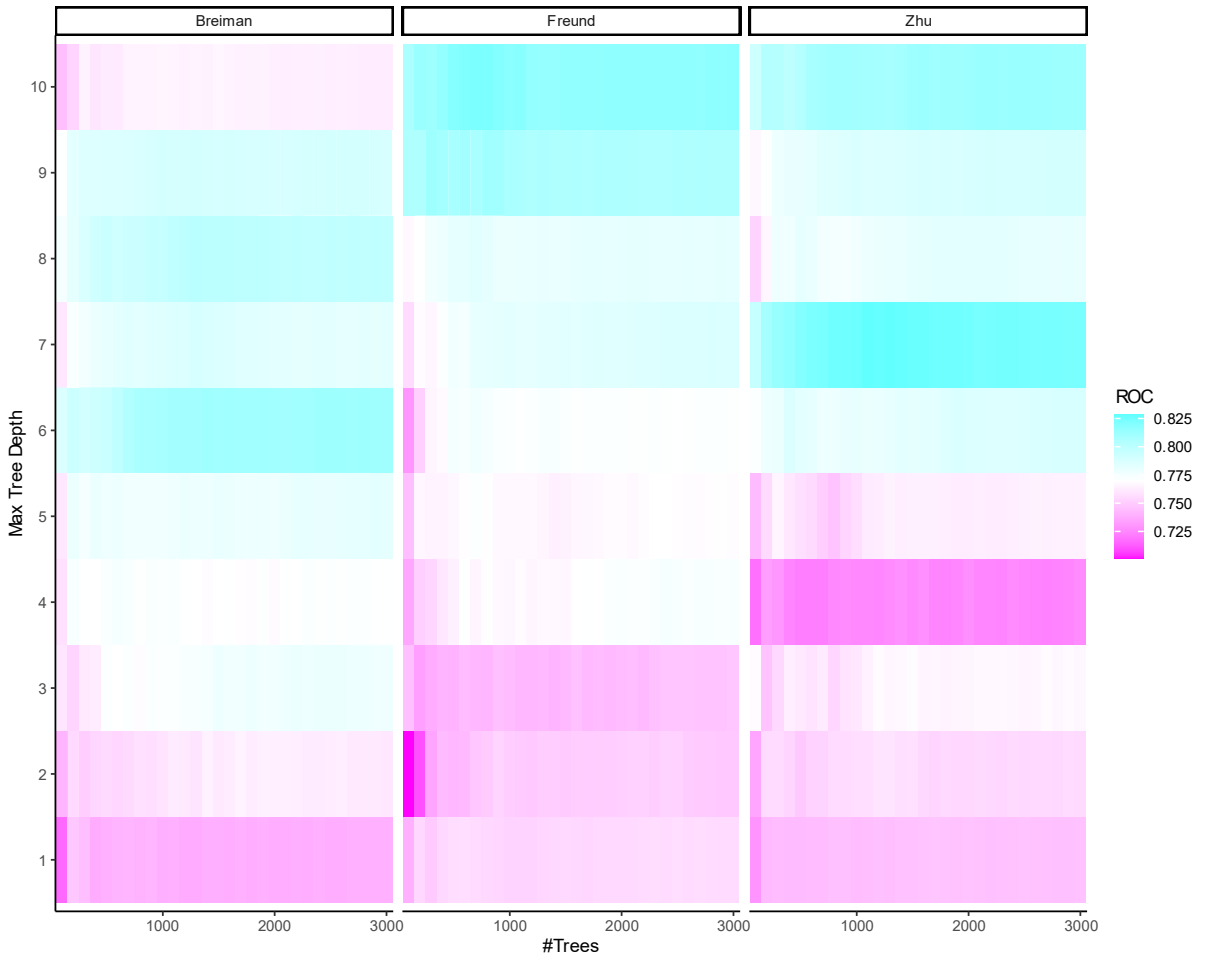
Second, due to the divergent presentation Figure D-3 is more suitable for assessing the two remaining hyperparameters, i.e. a suitable learning coefficient and an appropriate depth of the decision trees used. In contrast to Figure D-2, the general classification performance is illustrated by color. Again, three coordinate systems for each learning coefficient are included

with the number of grown trees on the x-axis. However, the maximum depth of trees on the y-axis enables a better assessment of the contribution of this parameter to the classification performance. In a direct comparison of the learning coefficients, the two variants 'Freund' and 'Zhu' appear to be superior to the 'Breimann' coefficient, especially as the depth of the trees increases. Although the latter has a relatively good performance for a maximum tree depth between 6 and 9, it does not reach to the maximum values of the other two learning coefficients. I therefore decided against using the 'Breiman' coefficient, although it could still lead to solid results if the depth is set to a medium level. The direct comparison between the learning coefficients 'Freund' and 'Zhu' shows that 'Zhu' can also achieve very strong performance overall. However, this performance appears to vary more significantly depending on the depth of the trees. In contrast, the performance of 'Freund' seems to increase more consistently with greater tree depth, rising relatively steadily up to the previously set maximum depth limit of 10. Particularly due to this consistency at greater tree depths, I choose the learning coefficient 'Breiman' for the final model training.

This decision is also accompanied by the choice of the maximum tree depth. For reasons related to computational requirements, I previously determined that the maximum depth of each individual tree should not exceed 10. Overall, the classification performance tends to increase with the maximum tree depth, even if there is, e.g., an outlier as a relatively poor performance for the ‘Breiman’ coefficient and a maximum tree depth of 10. In particular, the learning coefficient 'Freund' demonstrates that a high performance is achieved at a depth of 9, with performance slightly higher at the previously set limit of 10. The general tendency of better performance with deeper trees can also be seen with 'Zhu', but the consistency of this effect

appears to be higher with the learning coefficient 'Freund' in the range of a maximum tree depth around 9 or 10. Taking these aspects into consideration, I therefore opt for the upper limit of the predefined range, i.e. a maximum tree depth of 10.

Figure D-3: Hyperparameter tuning of RUSBoost model (2/2)



The figure illustrates the hyperparameter tuning for the RUSBoost model based on the model trained to finally detect misstatements in 2003. As illustrated in Figure D-1, Panel A, the underlying model is trained on data from 1990 to 1999 and for validation purposes data from 2000 and 2001 is used. The performance is illustrated by a color scale using the AUC, which the applied package labels as ROC. The hyperparameter tuning covers three parameters. First, each of the three panels illustrates a different learning coefficient ('Breiman', 'Freund', and 'Zhu'), set to control the way in which consecutively grown trees learn from previous errors. Second, within each panel on the y-axis the level for the maximum depth of a single grown tree is illustrated, covering a range of the maximum depth between 1 and 10. Third, on the x-axis, the number of consecutively grown trees is stepwise validated for a range between 100 and 3,000 trees.

To summarize, the hyperparameters for the subsequently trained RUSBoost models are defined as follows: The number of sequentially trained trees is set to 3,000, the learning coefficient

'Breiman' is used to account for learning from prior misclassifications, and the maximum depth of each individual tree is limited to 10.

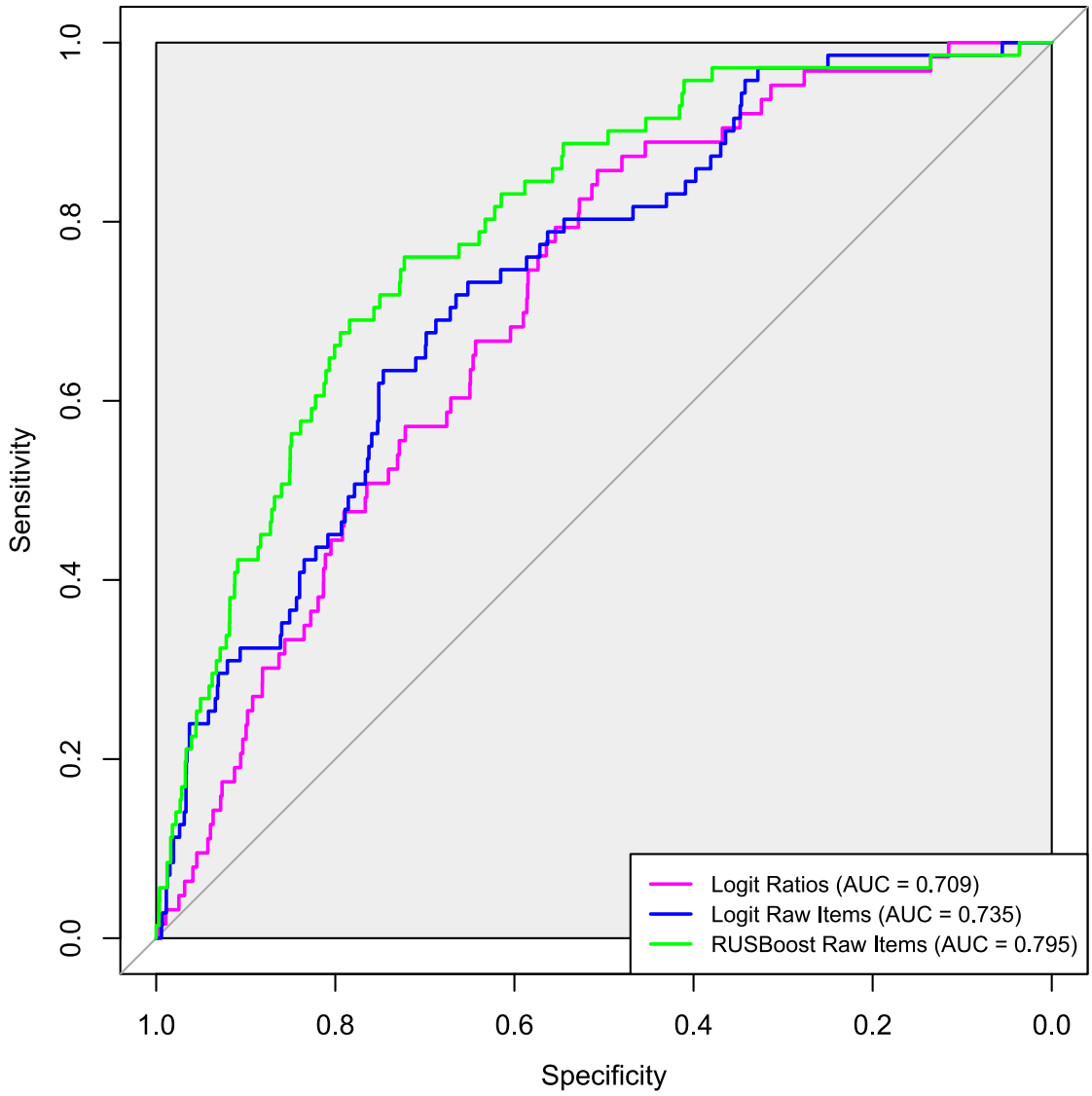
3.3.4 Interim Evaluation and Comparison with Benchmark Model

As mentioned earlier in section D.3.3, I follow the approach of *Bao et al.* (2020) and compare an exemplary RUSBoost model for the test year 2003 with two logistic regression models as benchmark models – one trained using financial ratios and the other with raw financial data items. This intermediate step aims to ensure that the RUSBoost models trained here, consistent with previous literature, achieve comparable results, namely a generally superior performance compared to the less complex logistic regression models.

As previously described, the AUC is a measure of a classification model's performance. It represents the area under the ROC curve, while the ROC curve illustrates the relationship between the TPR, also called sensitivity or recall, and the FPR, equivalent to 1 – specificity, at various threshold levels (*Fawcett*, 2006). Here, the TPR represents the proportion of actual fraud cases that a model correctly classifies as fraud. A high TPR indicates that many actual fraud cases are correctly identified, while only a few remain undetected as False Negatives. In contrast, the FPR represents the proportion of non-fraud cases that are incorrectly classified as fraud. To illustrate the ROC curves above, I chose the approach to reverse the scale between 0 and 1 on the x-axis in order to directly plot the specificity as a more familiar measure instead of the specificity minus 1, while this has no influence on the structure of the curves themselves. As can be observed in Figure D-4, exemplarily presented for the test year 2003, the AUC for the RUSBoost model is larger, making it superior to the logistic regression models for most thresholds. This is in line with the findings by *Bao et al.* (2020), first, superior results for

incorporating raw financial data items, and second, superior results for applying a tree-based boosting algorithm instead of regression-based approaches.

Figure D-4: Classification performance compared to benchmark models



The figure illustrates the ROC curve and respectively below the AUC for three models. Based on training data from 1990 to 1999, the figure shows ROC curves for the models' general ability to detect misstatements in test year 2003. The ROC curves contrast the performance of a RUSBoost model trained on raw financial data items with two regression-based benchmark models, trained on financial ratios and raw financial data items, as proposed by Bao et al. (2020) in order to test for superior classification performance of a tree-based boosting approach trained with raw financial data items.

Up to this point, the AUC has only been used to assess the overall classification performance, i.e. across all possible thresholds. However, particularly in the edge areas of Figure D-4, it becomes evident that a high TPR, while minimizing false negatives, comes at the cost of an increased number of false positives. To address the trade-off between these two types of errors, false negatives and false positives, the following section examines the different costs associated with each and uses this analysis as the basis for determining a cost-sensitive threshold.

3.4. Cost-Sensitive Predictions

Costs incurred by the misreporting company, such as penalties or reputation losses (*Karpoff et al.*, 2008), are not the focus of the following analysis. The costs addressed here are those incurred by potential users of financial statement fraud detection models. These primarily include investors, auditing firms, and enforcement authorities (see section C.2).

However, simply considering the AUC alone can be misleading, as the AUC represents the area under the ROC curve, which again is a graph illustrating the classification results for a range of potential threshold values. It does not account for the differing costs associated with different types of misclassification errors. This limitation is particularly pronounced in cases of class imbalance, where the AUC's informative value is restricted (*Beneish & Vorst*, 2022). Costs for these potential user groups may primarily arise due to two different types of errors. *Beneish and Vorst* (2022) describe and categorize the occurring costs as follows: First, false positive costs, i.e. those costs resulting from incorrectly flagged non-fraud firms. And second, false negative costs for missed fraud cases. False negative costs are somewhat more apparent and include, for instance, liability costs for auditors, reputational damage, and the resulting loss of clients. For investors, false negative costs in the context of capital markets refer to the losses

incurred from stock price declines. False positive costs, on the other hand, for auditors are primarily associated with the additional effort required to address risks resulting from a fraud classification. For investors, these could represent missed profits from potentially lucrative investments that were wrongly classified as high-risk (*Beneish & Vorst*, 2022).

As early as 1995, *Persons*, building on *Dopuch et al.* (1987), incorporated the expected costs of misclassification into a logistic classification model for fraudulent financial statements. In this approach, the costs of Type I and Type II errors are multiplied by their respective probabilities. Based on this framework, cost minimization is achieved by iteratively determining the optimal cutoffs for classification. The Type I errors here refer to the false negatives and Type II errors are false positive cases (*Persons*, 1995), while other research in this field of application defines the error types vice versa (*Hajek & Henriques*, 2017). Crucial for determining cost-optimal values for classification cutoffs are the relative costs associated with the different types of errors that may occur.

For example, based on incurred losses and audit fees *Hajek* (2019) employs a cost ratio of 2:1, assigning twice the weight to the costs of a false negative classification compared to those of a false positive classification from the perspective of an auditor. In contrast, *Persons* had earlier argued for a higher relative weighting of false negatives, i.e., undetected fraud cases. She considered a higher relative cost ratio of 30:1 to be more realistic for potential users of such classification models (*Persons*, 1995). This weighting is also more aligned with the approach taken by *Beneish*, particularly for investors. For instance, he assumed a range of 1:1 to 40:1 (*Beneish*, 1997) and later extended this to higher ranges, up to 100:1, while identifying a relative cost ratio of 20:1 to 30:1 as a more realistic spectrum for investors (*Beneish*, 1999a). This

perspective has been revisited in later studies, with some extending the considered range to relative costs as high as 200:1, however, still referring to ranges of about 20:1 to 30:1 as being recognized as applicable weightings of relative costs (*Cecchini et al.*, 2010a). Against this backdrop, *Beneish and Vorst* emphasize that the selection of relative costs is heavily assumption-driven. They address this by calculating the absolute costs arising from different perspectives of potential user groups of financial statement fraud detection models. In particular, their analysis reveals relatively higher costs for false negatives from the perspective of auditors (*Beneish & Vorst*, 2022).

More recent studies applied, e.g., MetaCost as a thresholding method to address relative costs which relabels training data based on calculated class averages by the model over multiple samples and considers diverging costs for different error types (*Kim*, Y. J. *et al.*, 2016). Another approach, a cost-sensitive cascade forest, is proposed by *Huang*, L. *et al.* which penalties heavily for false negatives. Although, this approach does not in particular consider estimated relative costs but instead adjusts the weights depending on the frequency of fraud cases within the training samples, thus, assuming relative costs to be comparable to the frequencies in which fraud occurs in drawn samples (*Huang*, L. *et al.*, 2022). These approaches share the common characteristic of being thresholding methods. Their goal is to make so-called "cost-blind" classifiers sensitive to costs *Ling and Sheng* (2008). For example, a model's classification performance can be represented across all possible thresholds using the AUC. A thresholding method would then select a threshold – as illustrated by a specific point on the ROC curve – where the assumed costs, based on a given ratio, are minimized. Instead of using estimated

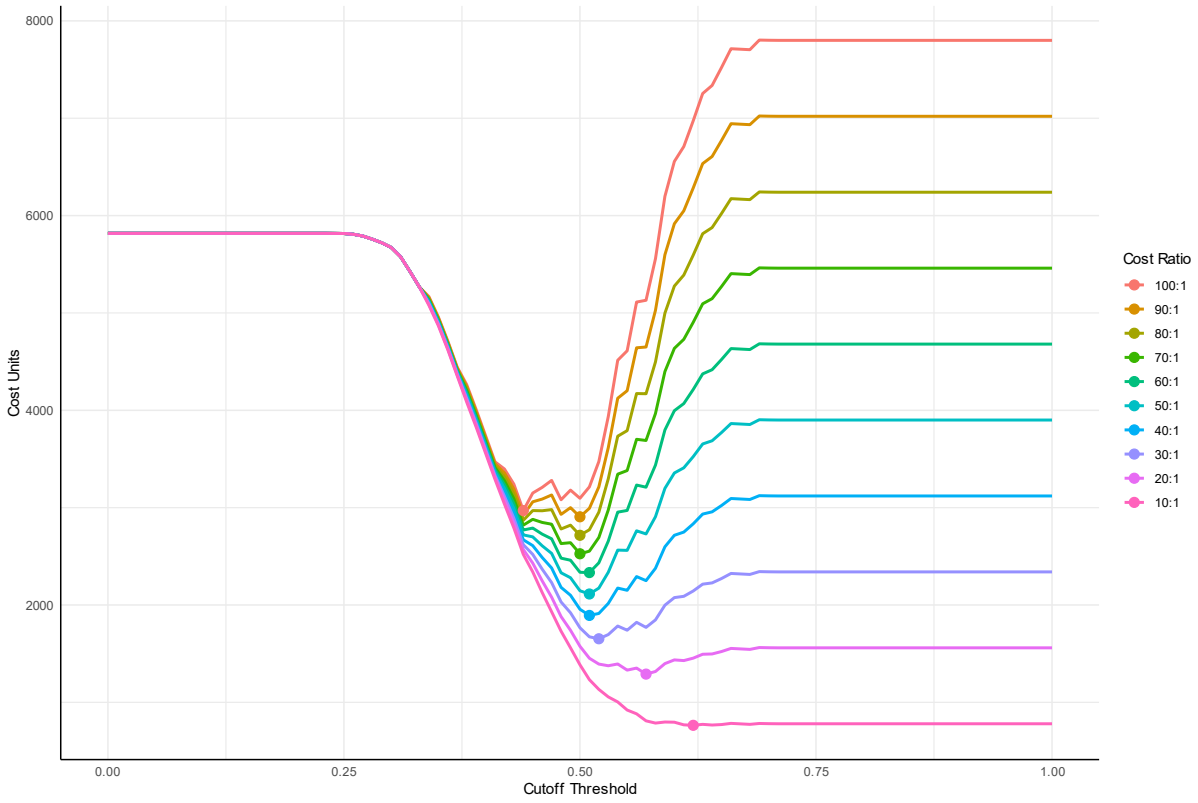
probabilities, as used in calculating the expected costs of misclassification, I calculate costs on actual predictions as follows.

A binary classification model typically calculates a score, e.g., between 0 and 1 for each observation. Here, as fraudulent cases are labeled as 1, a score closer to 1 indicates a potential fraud risk. However, the final decision of the classification model – whether an observation is predicted as a fraud case – primarily depends on the chosen threshold, which determines the point at which the score is considered high enough for an observation to be classified as a fraud prediction. As described subsequently, I select this cost-efficient threshold taking different cost ratios into consideration. Based on a model trained on data from 1990 to 1999, which's hyperparameter have been previously validated based on data from 2000 and 2001, I use the "new" unused data from 2002 to make predictions (see Panel A of Figure D-1). I take these prediction results and consider different cost ratios for the two error types, false negatives and false positives, to calculate an abstract cost measure. The objective is to minimize this abstract cost measure by analyzing the range of possible thresholds. The process for determining the cost-efficient thresholds is illustrated in Figure D-5 (here illustrated for the version based on non-normalized data). For various assumed cost ratios, represented as individual graphs, the abstract absolute costs are plotted as a function of the threshold value of the underlying classification model. Each plot highlights the point corresponding to the threshold that minimizes the costs for the respective cost ratio. As previously discussed, I consider the investor perspective to represent the lower bound for relative costs that is realistic for a wide range of user groups. This is because auditors, in particular, would likely operate with a cost ratio that exceeds this baseline (*Beneish & Vorst*, 2022). Therefore, I expect that, in line with previously

discussed literature, the relevant range for analyzing relative costs lies particularly between a cost ratio of 20:1 and approximately 50:1; for the sake of completeness, the range from 10:1 to 100:1 is calculated and presented.

Figure D-5 illustrates that in the range of very low thresholds (approximately 0.00 to 0.25), the cost level remains constant across all assumed cost ratios. In this range, almost all observations are classified as fraud because the threshold is set so low. Consequently, there are no false negatives, and thus no associated costs for undetected fraud cases. Instead, the costs incurred in this range result only from false positives. In contrast, in the range of high thresholds (approximately 0.75 to 1.00), almost all observations are classified as non-fraud, since the threshold is set too high for fraud classification. Here, the costs of false negative predictions become decisive, varying in magnitude depending on the assumed cost ratios. The shape of the graphs in the range of thresholds between 0.25 and 0.75 reveals the area where the total abstract misclassification costs are minimized. As the threshold increases, the number of false positives and their associated costs initially decreases. However, depending on the assumed cost ratio, the total abstract costs begin to rise again at higher thresholds, driven by the increasing cost of undetected fraud cases. Only when assuming low relative costs for false negatives (e.g., 10:1 or 20:1) does this increase remain marginal. In contrast, for cost ratios to be considered relevant (30:1 and higher), a U-shaped pattern emerges. This pattern corresponds to the threshold range where a trade-off is made between minimizing the costs of undetected fraud cases and limiting the additional workload from false positives. As a result, the cost-efficient thresholds for relevant cost ratios are found to be slightly above 0.5. The minimum cost values identified in this process are marked in the graphs and numerically presented in Table D-7.

Figure D-5: Relationship between cutoff thresholds and misclassification cost ratios



The figure illustrates the search for cost-efficient cut-off thresholds. A classification model typically calculates a specific score, e.g., between 0 and 1. The threshold determines above which value an observation is classified as a misstatement and is shown on the x-axis. Taking into account different cost ratios between the relative costs for false negative and false positives predictions, an abstract cost measure is calculated and shown on the y-axis. Each graph represents the abstract cost measure for a different assumed relative cost ratio between the two types of classification errors. The dot on each graph points out the minimum value of the abstract cost measure, and thus, identifies the cost-efficient cut-off thresholds for the assumed cost ratios.

As listed in Table D-7, for the various assumed cost ratios, the minimum values of an abstract cost measure are calculated. These values are determined for models using both normalized and non-normalized data. The results indicate that for the model with non-normalized data, the cost-minimizing threshold in the range of the assumed relevant cost ratios is approximately 0.51. For models with normalized data, the threshold is slightly higher and can be quantified as 0.53 within the relevant range. These threshold values are subsequently used in the models to perform classifications that account for the differing costs of the various types of misclassification errors.

Table D-7: Cost efficient classification cutoff thresholds

	Non-normalized Features		Normalized Features	
Cost Ratio	**Total Cost Units**	**Associated Threshold**	**Total Cost Units**	**Associated Threshold**
10:1	764	0.62	712	0.65
20:1	1,290	0.57	1,231	0.56
30:1	1,653	**0.52**	1,519	**0.53**
40:1	1,893	**0.51**	1,749	**0.53**
50:1	2,113	**0.51**	1,979	**0.53**
60:1	2,333	0.51	2,196	0.51
70:1	2,526	0.50	2,386	0.51
80:1	2,716	0.50	2,576	0.51
90:1	2,906	0.50	2,760	0.49
100:1	2,970	0.44	2,920	0.49

The table lists for each cost ratio the lowest determined cost measure and the corresponding threshold. The cost ratio refers to the assumed ratio of costs occurring for false negatives compared to false positives. Total Cost Units are an abstract measure which calculates costs based on prediction results and the costs for different types of misclassification costs. The Associated Threshold is the cutoff value, which leads to the lowest misclassification costs for a certain cost ratio.
On the left, values are listed for the model trained on non-normalized financial data, on the right, values are listed for the model trained on normalized data.

4. Classification Performance Evaluation

The previously established thresholds take into account the discussed cost ratios, which are considered reasonable across different user groups. These thresholds play a crucial role in determining the point at which a model classifies an observation as either fraudulent or non-fraudulent. The classification results for the different model variants, based on the assumed cost ratios and their corresponding thresholds, are presented in Table D-8. This table contains aggregated performance metrics spanning several years. A detailed presentation of the performance measures for each individual yearly slice can be found in Appendix H. The AUC, as described in detail in section D.3.3, is the only performance metric in the presentation that is independent of the thresholds. In contrast, the other listed performance metrics – accuracy, sensitivity, precision, specificity, and FPR – are explicitly dependent on the cost ratios and

thresholds. Accuracy is considered the most simple classification performance metric calculated as the overall percentage of correct classifications (*Ferri et al.*, 2009). The remaining performance metrics are categorized and calculated according to *Fawcett* as follows. The sensitivity, respectively the true positive rate or also called recall or hit rate, represents the percentage of correctly classified positives over all actual positive cases, i.e. in this setting the percentage of detected fraud cases. In contrast, precision or positive predictive value calculates true positives as a percentage over all positive predictions. Specificity, on the other hand, presents the proportion of true negatives among all actual negative cases, here, meaning it measures the correct classification of non-fraud cases relative to the total number of actual non-fraudulent cases. Last, the false positive rate or FPR represents the counterpart to specificity as the specificity can be calculated as 1 – false positive rate. However, direct calculation considers the FPR as the proportion of incorrectly classified negative cases over all actual negative cases. Therefore, the FPR is also called false alarm rate (*Fawcett*, 2006). These classification performances are reported for the four different variants of the trained models. On the one hand, they are differentiated based on whether normalized or non-normalized input data were used. On the other hand, they are presented in two variations regarding the time periods of the training data: either using all available data with a retrain for each additional test year in the sense of a "rolling origin retrain," or with a fixed-length training period of 10 years in the sense of a "rolling window" (see section D.3.3 for details).

As shown in Table D-8, the model variants differ only slightly in terms of their respective AUC values, with no specific trend observable between them. However, a trend is evident in the development of the AUC when the test period is extended. When results from

additional years are aggregated, it becomes apparent that the AUC tends to decrease slightly. This effect is even more pronounced when examining the individual yearly slices presented in Appendix H. Considering the number of fraud cases included in the dataset, this development is to be expected. Starting with a proportion of firms with financial misstatements of over one percent, this share steadily decreases over time, reaching only 2 fraud cases (0.04 %) in the dataset for 2019 (see Table D-2). This trend complicates further training, as fewer fraud cases are known or included within the dataset's recent years, and it can also lead to individual undetected fraud cases having a disproportionately negative impact on performance metrics.

An examination of the remaining performance metrics reveals the following: When comparing the models using normalized versus non-normalized data, the normalized data variants demonstrate superior classification performance in terms of accuracy, specificity, precision, and FPR. Specifically, accuracy, specificity, and precision are higher in the normalized models, while the FPR is lower. However, it is important to note that sensitivity is explicitly better in the non-normalized variants. Here, sensitivity represents the proportion of correctly classified fraud cases out of all fraud cases, indicating how effectively a model identifies actual fraud cases. This means that the improved performance metrics of the normalized models come at the expense of weaker identification of actual fraud cases. Conversely, models based on non-normalized variables tend to better identify fraud cases, but this is accompanied by a higher rate of false positive classifications.

With regard to the extent of the training data used, both variants – using normalized and non-normalized data – show that the "rolling window" approach tends to result in higher accuracy, precision, and specificity, as well as lower FPR, compared to training with a "rolling

origin retrain". Only the sensitivity appears to be slightly higher in the case of the "rolling origin retrain".

Overall, there is no single superior model variant; each has specific strengths and weaknesses. Considering the overall performance, models trained using the "rolling window" approach appear to perform better across most performance metrics, with only a slight disadvantage in terms of sensitivity. Meanwhile, the normalized and non-normalized variants differ primarily in their ability to identify actual fraud cases, albeit at the cost of additional false positives. Therefore, for the subsequent analyses, I have decided to use the "rolling window" approach, applying it to both options: one based on normalized input data and the other on non-normalized input data.

Table D-8: Aggregated classification performance of RUSBoost models

			Aggregated Measures for each Test Period							
Features	**Training Periods**	**Performance Measure**	**2003–2005**	**2003–2007**	**2003–2009**	**2003–2011**	**2003–2013**	**2003–2015**	**2003–2017**	**2003–2019**
Non-normalized	1990 – (testyear–3)	AUC	0.805	0.800	0.779	0.781	0.776	0.775	0.766	0.763
		Accuracy	72.4%	72.8%	74.3%	75.1%	75.7%	76.1%	76.3%	76.6%
		Sensitivity	73.6%	73.5%	66.8%	66.5%	63.6%	62.4%	60.4%	58.1%
		Precision	2.8%	2.2%	1.9%	1.8%	1.6%	1.5%	1.3%	1.2%
		Specificity	72.4%	72.8%	74.4%	75.1%	75.8%	76.1%	76.4%	76.7%
		FPR	27.6%	27.2%	25.6%	24.9%	24.2%	23.9%	23.6%	23.3%
	rolling 10 years	AUC	0.796	0.782	0.758	0.767	0.770	0.770	0.765	0.754
		Accuracy	75.3%	74.9%	76.5%	76.5%	77.5%	77.9%	78.1%	78.3%
		Sensitivity	70.5%	68.5%	58.9%	59.8%	57.1%	56.6%	56.8%	53.6%
		Precision	3.0%	2.3%	1.9%	1.7%	1.6%	1.4%	1.3%	1.2%
		Specificity	75.4%	75.0%	76.6%	76.6%	77.6%	78.0%	78.2%	78.4%
		FPR	24.6%	25.0%	23.4%	23.4%	22.4%	22.0%	21.8%	21.6%
Normalized	1990 – (testyear–3)	AUC	0.797	0.789	0.760	0.777	0.779	0.778	0.768	0.757
		Accuracy	82.2%	81.7%	82.5%	82.9%	83.8%	84.2%	84.7%	84.8%
		Sensitivity	60.9%	63.7%	52.1%	56.7%	53.0%	51.8%	49.6%	45.8%
		Precision	3.5%	2.8%	2.2%	2.2%	2.0%	1.8%	1.6%	1.5%
		Specificity	82.4%	81.9%	82.7%	83.1%	83.9%	84.4%	84.8%	85.0%
		FPR	17.6%	18.1%	17.3%	16.9%	16.1%	15.6%	15.2%	15.0%
	rolling 10 years	AUC	0.802	0.778	0.760	0.775	0.779	0.783	0.781	0.773
		Accuracy	84.0%	83.2%	84.3%	84.4%	84.8%	84.7%	84.7%	84.7%
		Sensitivity	59.3%	56.0%	49.7%	52.4%	49.5%	49.0%	49.7%	47.2%
		Precision	3.9%	2.9%	2.5%	2.3%	2.1%	1.9%	1.7%	1.6%
		Specificity	84.3%	83.5%	84.5%	84.6%	85.0%	84.9%	84.9%	84.8%
		FPR	15.7%	16.5%	15.5%	15.4%	15.0%	15.1%	15.1%	15.2%

The table lists the classification performances for the RUSBoost model. First, measures for the model trained on non-normalized data items are shown, and second, for a model trained on normalized data. Sections with training periods "1990 – (testyear–3)" cover, e.g., for the test year 2003 the financial years from 1990 to 2000. The "rolling 10 years" covers for each test year 10 years of training but leaves out the 2 preceding periods before the test year. Performances are displayed by:
1) Area Under the Receiver Operating Curve (AUC)
2) Accuracy = (TP + TN)/(TP + TN + FP + FN)
3) Sensitivity = TP/(TP + FN)
4) Precision = TP/(TP + FP)
5) Specificity = TN/(FP + TN)
6) False Positive Rate (FPR) = FP/(FP + TN)
Performance measures 2) to 5) are based on classifications with the previously determined cost efficient thresholds of 0.51 for the models based on non-normalized financial items and 0.53 for normalized financial items. The table shows aggregated results for different test periods. Aggregated measures are computed relative to the number of observations for each year.

5. Applied Approaches of Local Explanations

5.1. Model-Agnostic Interpretability and Prediction-Level Approaches

Following the analysis of the actual classification performance of the model variants, the focus now shifts to the interpretability of the models. The interpretability addressed here does not concern the general contribution of a feature to the overall model across all observations but rather focuses explicitly on individual observations, referred to as local explanations (see section B.2.2). The objective is to examine whether these local explanations can reliably identify manipulated areas within a financial statement.

AdaBoost or RUSBoost models are inherently non-interpretable and qualify as black-box models. Therefore, the functioning of a model cannot be readily understood or represented in a human-interpretable manner, such as through weightings. Instead, additional analyses of a model are required, which attempt to retrospectively understand and explain the model's mechanisms in a comprehensible way. Approaches to so-called model-agnostic interpretability offer several advantages. Model-agnostic interpretability refers to methods that operate independently of the type of underlying model, meaning it does not matter which algorithm was used to train the model. This applies whether the model is an inherently interpretable regression model, a tree-based model, or even a deep neural network. This approach allows for maintaining the high degree of flexibility of complex models while addressing and analyzing interpretability as a separate concern (*Ribeiro et al.*, 2016b).

In addition to model-independence, the focus on individual classifications is crucial. Prediction-level approaches are particularly relevant when practical users are interested in understanding how a specific classification decision was made (*Murdoch et al.*, 2019). In the

present application, this is of critical importance, as the analysis aims to determine whether local explanations can provide valuable insight into which areas of a financial statement have been manipulated. This aims to support effective financial statement fraud detection and efficient resource allocation. Fundamentally, there are now several approaches that allow for local explanations: e.g., Individual Conditional Expectation curves (ICE), which primarily provide a graphical representation of the individual effects of features; Counterfactual Explanations, or the two approaches used here – LIME and Shapley Values (*Molnar*, 2022). The selection of the approaches used was driven by two key considerations. First, it was essential to generate numerical explanations that are not primarily dependent on graphical representations. This ensures that not only exemplary explanations can be provided but also a quantitative summary of the explanations can be performed. According to *Molnar*, what particularly distinguishes LIME and Shapley Values among the approaches mentioned earlier is that they are attribution methods. This means that a prediction can be numerically described as the sum of the effects of the individual features rather than being example-based (*Molnar*, 2022). Second, practical technical constraints had to be taken into account. This means that approaches were chosen for which established software packages are available and which are highly compatible with other widely used tools. For these two reasons, I chose the approaches LIME and Shapley Values to generate the explanations for the models in the following analysis.

5.2. LIME

Ribeiro et al. proposed Local Interpretable Model-agnostic Explanations (LIME) in 2016. LIME is a model-agnostic method for interpreting machine learning models, meaning it can be applied to various models regardless of the algorithms used to train them. The core idea of

LIME is not to explain the entire black-box model but to derive surrogate models around individual observations. In other words, for each analyzed observation, LIME generates its own simplified local model. The influence of individual features on the prediction is then represented as values in this local surrogate model, making them interpretable for humans (*Ribeiro et al.*, 2016a). Technically, the derivation of the local model is achieved by using the trained classification model and the specific observation, without incorporating to the full training data or similar. LIME perturbs the feature values of the observation, uses these perturbed data as inputs, and analyzes how changes in these feature values affect the prediction. From this, it draws conclusions about which features contribute the most to the prediction in the observed region (*Molnar*, 2022) – in this case, for example, a classification as potentially fraudulent.

In principle, different types of simplified models could be used to derive an interpretable local model. This includes regression-based models as well as simple tree-based models or others. For the technical implementation, I use the R package 'iml' (Interpretable Machine Learning), which is one of the most widely used and compatible R packages in the field of interpretable machine learning. The 'LocalModel' function in this package is limited to deriving linear regression models, which excludes alternative approaches, such as tree-based models, from consideration (*Molnar et al.*, 2018).

5.3. Shapley Values

As the second approach for analyzing local explanations, I use Shapley Values. This is a game-theoretic approach originally based on *Shapley* (1953). As with LIME, Shapley Values provide a means to address the attribution problem. This refers to the challenge of tracing specific predictions back to individual input features and their contributions to the prediction.

Attribution, in this context, provides informational or explanatory value by identifying which features influenced decisions, such as classification outcomes (*Sundararajan & Najmi*, 2020).

Shapley Values are based on a game-theoretic approach. In this context, the binary classification model for financial statement fraud detection represents the "game," while the features – i.e., the input variables – are the "players" who collectively contribute to forming a prediction. As *Molnar* highlights, it is important to distinguish between marginal contributions and the actual Shapley Values. In the first step, a marginal contribution for a feature is calculated by considering a specific coalition of features. Within this coalition, the contribution of the individual feature is determined by assessing the effect of a random change in the value of the respective features on the output. In the second step, the Shapley Value for a feature is then computed as the average of all marginal contributions across all possible coalitions. Thus, the Shapley Value is explicitly not the difference between the model output with and without the feature, but rather the average of the calculated marginal contributions (*Molnar*, 2022).

Practically, it is sometimes argued that the use of Shapley Values, due to their computational complexity, is only efficiently feasible for models based on decision trees (*Messalas et al.*, 2019). This highlights the significant computational effort required. In the present case, I use AdaBoost respectively RUSBoost models, which are trained using tree-based algorithms, making them a practical application for Shapley Values. From a technical perspective, I implement Shapley Values, like LIME, using the R package ‘iml’, specifically employing the ‘Shapley’ function (*Molnar et al.*, 2018).

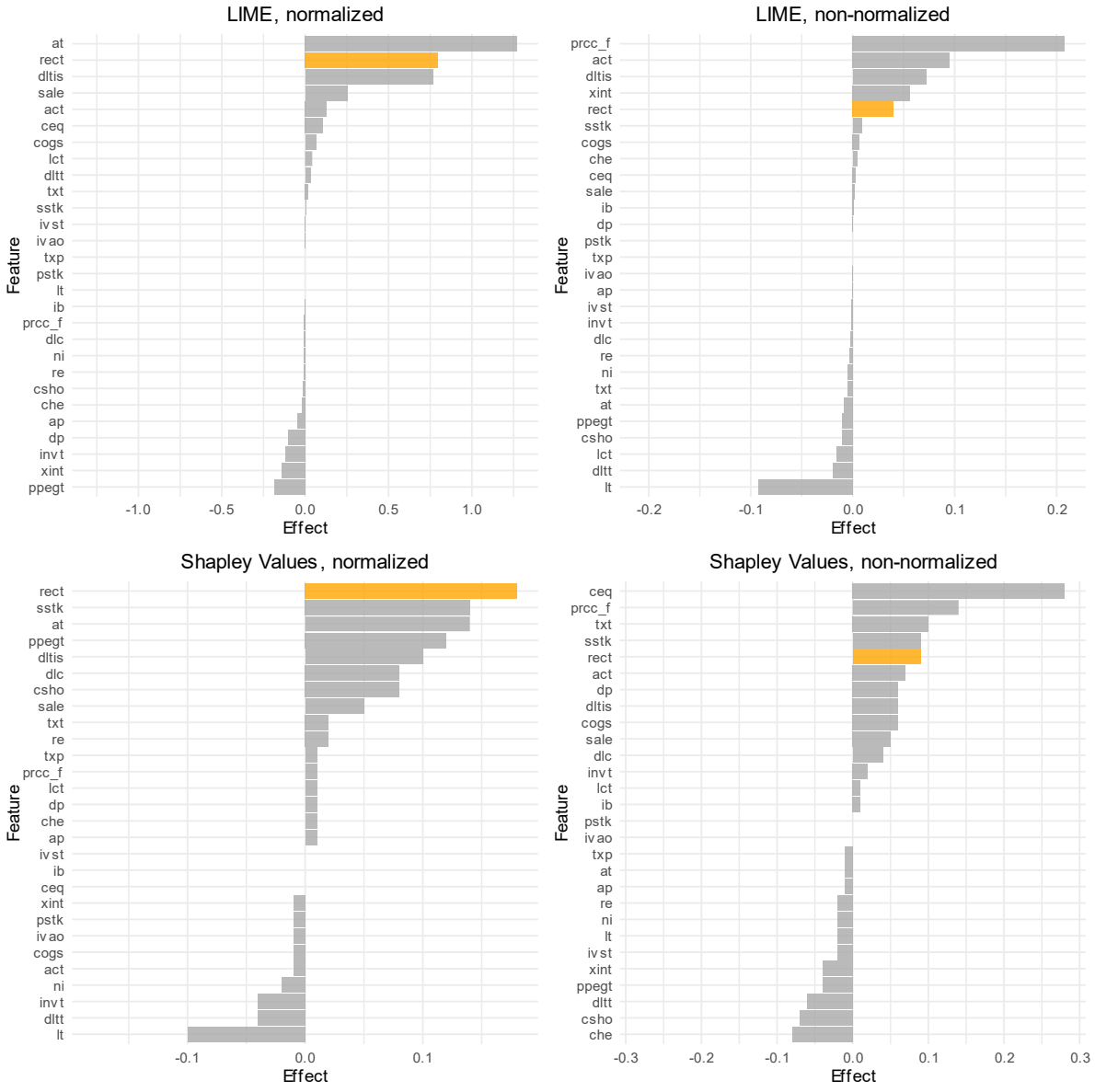
5.4. Analysis of Exemplary Local Explanations

Local explanations for two observations are presented using both LIME and Shapley Values, each for models trained on normalized and non-normalized data. The focus here is on illustrating the relationship between the type of manipulation and the highlighting of feature effects for this specific classification associated with the manipulated area. For this purpose, the categories from the AAER dataset are used, such as "Misstated accounts receivables" or "Misstated inventories" (see Table D-3). If a selected classification of a specific type of financial statement fraud is particularly driven by features linked to the manipulated area, this could potentially provide insights for a more targeted investigation of financial statement fraud. To illustrate this, two cases are examined in detail below.

Figure D-6 illustrates the explanations for a true positive prediction. The explained case (fyear: 2014; GVKEY: 14,303; AAER No.: 3,931) is categorized in the AAER dataset as "Misstated accounts receivable" and described slightly more precisely as "overstated accounts receivable estimates". Corresponding to this category, the feature Receivables, Total (rect) is included among the financial data items used to train the models. If the explanations indeed indicate manipulated areas, this variable should have contributed to the prediction that a misstatement is present. For this reason, the effects of the feature Receivables, Total (rect) are highlighted in color in Figure D-6. The results show that for both model variants applying LIME, this feature is among the top five features with the greatest effect on the classification as a misstatement. In particular, in the model trained on normalized data, Receivables, Total (rect) is the feature with the second strongest effect, immediately following "Assets, Total", which would also be inflated as a result of overstated receivables. The explanations provided

by Shapley Values are even more precise. Specifically, in the case of the normalized model variant, receivables emerge as the feature with the greatest influence on the prediction of potential financial statement fraud.

Figure D-6: Explanations for a true positive prediction of misstated receivables



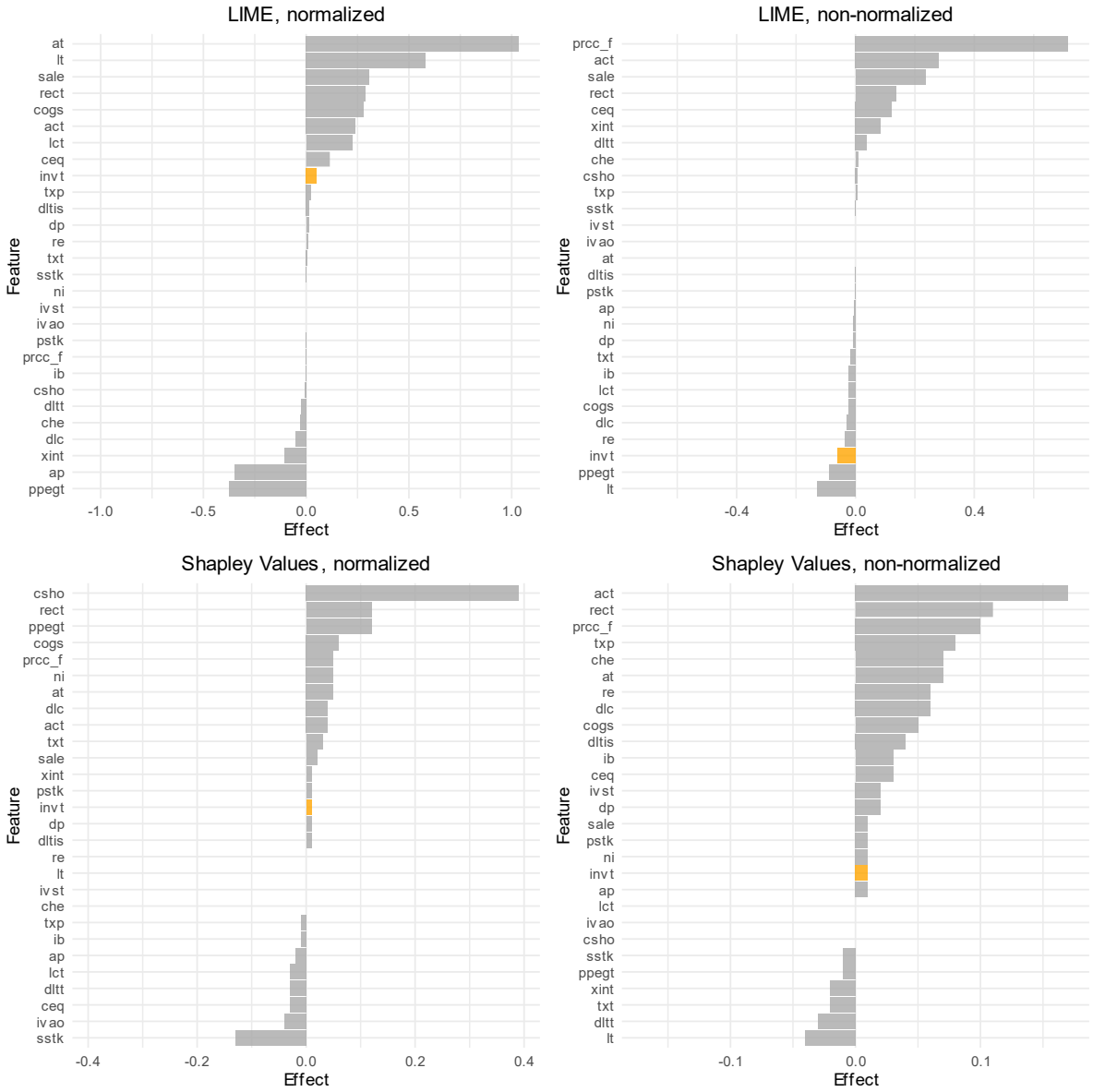
The figure illustrates LIME and Shapley Values for both, models trained on normalized and non-normalized data, for a specific observation (fyear: 2014, GVKEY: 14303, AAER: 3931). The AAER dataset specifies the category "Misstated accounts receivable" for this case and includes the description “overstated accounts receivable estimates”. Accordingly, the variable "rect", representing "Receivables, Total", is highlighted in color.

This could initially be seen as an exemplary indicator that the explanations relatively accurately describe the manipulated areas. This could be evidenced by the fact that features corresponding to the manipulated areas rank among those with the strongest effects on the classification. However, such a case is not necessarily representative of the majority of cases. Contrary to previous research, where local explanations were illustrated using examples without delving into the overall relationship between the type of manipulation and the driving features, the aim here is to present as transparent a picture as possible. This includes positive examples, such as in Figure D-6, as well as cases where features associated with the manipulated area do not make significant contributions to the classification as misstated.

In contrast to the explanations in the previous example, Figure D-7 illustrates a case where the local explanations do not reveal a connection between the type of manipulation and the associated features. This case is categorized in the AAER dataset as "Misstated inventory" and further described as "improperly overstating inventory and disclosure violations" (fyear: 2010; GVKEY: 25,405; AAER No.: 3,840). The Compustat data item “invt” represents "Inventories, Total", which relates to the account or subtotal reported in the AAER as overstated. A reliable local explanation for the manipulation in inventories would indicate that this line item contributes noticeably to the classification as potentially fraudulent. However, in the model trained with normalized data, according to LIME, it ranks only 9th out of 28 features in terms of its contribution to this correct classification. In the model based on non-normalized data, the effect of the feature is reversed, and Inventories, Total (invt) has a negative impact on the prediction. A similar picture emerges when examining the Shapley Values. Depending on the variant, the effect marginally points in the right direction – contributing to the classification

as fraudulent – but ranks only as feature 14 respectively 18 out of a total of 28. This places Inventories, Total (invt) as a potential indicator for "Misstated inventories" well outside the focus for potential users.

Figure D-7: Explanations for a true positive prediction of misstated inventories



The figure illustrates LIME and Shapley Values for both, models trained on normalized and non-normalized data, for a specific observation (fyear: 2010, GVKEY: 25405, AAER: 3840). The AAER dataset specifies the category "Misstated inventory" for this case and includes the description “Improperly overstating inventory and disclosure violations”. Accordingly, the variable "invt", representing "Inventories, Total", is highlighted in color.

This second case serves as a negative example, demonstrating a scenario where potential users of such a model might receive a correct classification but where the local explanations would provide unreliable clues for identifying the actual areas of manipulation. Thus, these two cases are a positive and a negative example to illustrate the local explanations provided by LIME and Shapley Values. To go beyond individual examples and identify potential patterns or biases, cases will be analyzed based on their classification outcomes.

6. Aggregated and Comparative Analysis of Explanations

6.1. Descriptive Analysis of Classification Results

The preceding examples illustrate how LIME and Shapley Values can represent the contributions of individual features to classification decisions. Up to this point, these are two examples intended to serve as illustrations and do not yet allow for overall conclusions. Therefore, the following analysis will examine the explanatory power of LIME and Shapley Values in a broader context. After a descriptive summary on the models' ability to detect certain types of misstatements and depending on the misstatements' complexity, the analysis will focus on 1) evaluating how well the local explanations serve as indicators for manipulated areas in cases of correct fraud classifications, 2) analyze if patterns can be identified in cases where fraud remains undetected (false negatives), and 3) analyzing drivers of false positive classifications to identify potential biases (false alarms).

Table D-9: Classification performance depending on the type of fraud

		normalized		non-normalized	
Account category affected	**Frequency**	**TP** (TPR)	**FN** (FNR)	**TP** (TPR)	**FN** (FNR)
Misstated revenue	261	136	125	169	92
		(52%)	(48%)	(65%)	(35%)
Misstatement of other expense/ shareholder equity account	266	162	104	198	68
		(61%)	(39%)	(74%)	(26%)
Capitalized costs as assets	107	37	70	45	62
		(35%)	(65%)	(42%)	(58%)
Misstated accounts receivable	81	46	35	48	33
		(57%)	(43%)	(59%)	(41%)
Misstated inventory	74	44	30	49	25
		(59%)	(41%)	(66%)	(34%)
Misstated cost of goods sold	62	35	27	32	30
		(56%)	(44%)	(52%)	(48%)
Misstated liabilities	81	42	39	45	36
		(52%)	(48%)	(56%)	(44%)
Misstated reserve account	46	34	12	37	9
		(74%)	(26%)	(80%)	(20%)
Misstated payables	23	13	10	12	11
		(57%)	(43%)	(52%)	(48%)
Misstated allowance for bad debt	0	0	0	0	0
		n/a	n/a	n/a	n/a
Misstated marketable securities	2	1	1	0	2
		(50%)	(50%)	(0%)	(100%)

The table lists the frequencies of account categories being affected by misstatements according to the AAER dataset for the test years from 2003 to 2019. The frequencies sum up to more than the number of misstated firm years as one misstated firm year can have multiple affected categories. For each account category affected by a misstatement the frequency of true positive and false negative classifications are illustrated for both model variants. Below the frequencies, the Sensitivity (i.e. True Positive Rate, TPR) and the False Negative Rates (FNR) are calculated based on the following equations:
TPR = TP/(TP+FN)
FNR = FN/(TP+FN)

In addition to the general classification performance discussed in section D.4, here, a more detailed description is given with regard to the complexity of fraud cases and their type. Table D-9 summarizes the classification results of the models, focusing on the types of misstated accounts according to the AAER dataset by *Dechow et al.* (2011) for which the models produced correct classifications (i.e., true positives) and the number of misstatements that remained undetected (i.e., false negatives). The total in the column "Frequency" exceeds the number of misstated firm-years within the test period from 2003 to 2019, as a single misstated firm-year can involve manipulations in multiple areas. As seen in Table D-3 for the full dataset, the most frequent types of manipulation during the test period are misstated revenues and misstatements of other expense/shareholder equity accounts, followed by capitalized costs as assets and misstated accounts receivable. The category "Misstatement of other expense/shareholder equity accounts" is particularly heterogeneous and is best understood as a catch-all category when a more specific assignment to one of the other categories is not possible. Overall, the model trained on non-normalized data tends to correctly classify more fraud cases and thus exhibits a higher TPR (True Positive Rate) for most types of misstatement.

However, as already shown in Table D-8, this comes at the expense of a significantly higher FPR, meaning the model produces substantially more false alarms in the form of false positives. When examining the types of misstatements, the categories "Capitalized costs as assets" and "Misstated reserve account" stand out the most. While cases of "Misstated reserve account" are relatively well classified, with TPRs of 74 % and 80 %, the TPR for "Capitalized costs as assets" is significantly lower, at 35 % and 42 %. In the remaining categories, the TPRs

predominantly range between 50 – 60 % for the model trained on normalized data and 50 – 65% for the model trained on non-normalized data.

In addition to the substantive types of misstatements, Table D-10 differentiates between misstatements based on the number of different types of misstatements. The "number of account categories affected" indicates how many of the types of misstatements listed in Table D-9 occurred simultaneously for a manipulated firm-year in the test data. This number can thus be understood as a measure of the complexity or extent of a manipulation, though not necessarily its monetary magnitude. For example, if only one category is affected, it could represent a case where inventories are misstated largely in isolation. However, if, e.g., three categories are involved, the same firm year could simultaneously exhibit misstated inventories, misstated revenues, and misstated receivables as an example. If the number of simultaneous types of misstatements is interpreted as a measure of complexity, one might have expected that increasing complexity could involve more intensive efforts to obscure individual issues, making it more challenging for models to correctly classify these manipulation cases. A detailed analysis reveals differences between the models in this regard. The model trained on non-normalized data performs particularly well in classifying manipulations involving fewer affected areas, achieving a TPR of 73 %, but shows a lower TPR for cases with more affected categories. In contrast, the model trained on normalized data appears better at identifying more complex manipulations involving multiple manipulated areas. While the TPR is only 54 % for the major group of cases with a single affected category, it rises to 70 % and 69 % for cases with five and six misstated account types, respectively. This could potentially be interpreted as an indicator that these cases may not involve "well-executed" obfuscations that could have

misled a model but rather represent more extensive manipulations with overall more pronounced characteristics.

Table D-10: Classification performance depending on the complexity of fraud

Number of account categories affected	Frequency	normalized TP (TPR)	normalized FN (FNR)	non-normalized TP (TPR)	non-normalized FN (FNR)
1	224	121 (54%)	103 (46%)	163 (73%)	61 (27%)
2	138	65 (47%)	73 (53%)	79 (57%)	59 (43%)
3	69	32 (46%)	37 (54%)	46 (67%)	23 (33%)
4	17	11 (65%)	6 (35%)	7 (41%)	10 (59%)
5	30	21 (70%)	9 (30%)	20 (67%)	10 (33%)
6	13	9 (69%)	4 (31%)	8 (62%)	5 (38%)

The table lists the frequency of fraudulent observations between 2003 and 2019 within the test data, grouped by the number of categories being affected by manipulations according to the AAER dataset. For each number of affected categories, the frequency of true positive and false negative classifications are illustrated for both model variants. Below the frequencies, the Sensitivity (i.e. True Positive Rate, TPR) and the False Negative Rates (FNR) are calculated based on the following equations:
TPR = TP/(TP+FN)
FNR = FN/(TP+FN)

Overall, these classification results, differentiated by types of misstatements and their number within a case, do not suggest that there are specific types of manipulations that are particularly well detected by these models, nor do they reveal a consistent pattern for cases involving more extensive manipulations. The only discernible tendencies were a relatively good classification for "Misstated reserve accounts" and weaker classification for "Capitalized costs as assets". Regarding the number of manipulated areas, there is only a slight tendency toward better

classification of more complex or extensive manipulations by the normalized model and better classification of less extensive manipulations by the non-normalized model.

6.2. Prediction Explanations and Related Feature Rankings

A challenge within design science approaches is the evaluation of the artifact, as there has been a lack of guidance in this regard. However, early on, a combination of illustrative examples and, where applicable, technical experiments emerged as the most common form of evaluation (*Peffers et al.*, 2012). As proposed by *Venable et al.* (2016) in their framework, two dimensions should be considered when determining an appropriate approach to artifact evaluation. The first dimension addresses the question of "Why to evaluate", i.e., whether the aim is to improve the artifact itself (formative evaluations) or to assess the outcomes in terms of their alignment with established expectations (summative evaluations). The second dimension concerns the underlying paradigm of "How to evaluate". Here, a distinction is made between artificial evaluation and naturalistic evaluation. For the evaluation of the models trained here and the derived explanations, the evaluation can be classified as follows: The analysis is not intended to explicitly improve the models or explanations in the sense of a formative evaluation. Instead, it seeks to examine the extent to which the explanations are fundamentally capable of providing targeted insights into actually manipulated areas of financial statements. Thus, the approach pursued can be categorized as a summative evaluation. Furthermore, regarding the second dimension and the question of "How to evaluate," the approach taken here falls under artificial evaluation and explicitly not under naturalistic evaluation. The analysis focuses on assessing the ability of the explanations to highlight manipulated areas of financial statements based on

the available datasets. The involvement of potential user groups and their interaction with such models in real-world applications is not addressed.

Similar to the findings of best practices by *Peffers et al.* (2012), *Sutton et al.* (2021) explicitly highlight two distinct aspects as separate steps: A "demonstration" using individual cases and a more extensive evaluation, which may involve either a technical or a scientific evaluation with human interaction. For the analysis of the model, the evaluation of classification performance in section D.4 should be regarded as an intermediate result. The key output to be critically assessed are the explanations provided by LIME and Shapley Values. Especially in the context of the interpretability of machine learning-based models, it is emphasized that it is insufficient to demonstrate a model's correctness using only prototypes or isolated cases. For a more comprehensive understanding of the models, it is essential to also examine cases where predictions fail, to identify the limitations of the models and the methods used (*Kim*, B. *et al.*, 2016). In line with the required demonstrations or illustrative examples, in section D.5.4 I presented both positive and negative examples of the ability of LIME and Shapley Values to provide targeted insights into misstated areas in this context. Furthermore, to cover more than an example-based overview of a few selected cases, the following analysis focuses on examining the explanations in their entirety.

In the context of interpretable machine learning, there is a notable lack of clear guidance on evaluating explanations and their interpretability (*Vilone & Longo*, 2021). *Doshi-Velez and Kim* (2017) address this issue, responding to the question, "Should we be concerned about a lack of rigor?" with "Yes and no." They emphasize that the evaluation of explanations often hinges on whether they appear reasonable, a judgment that is heavily influenced by human

perception and subjectivity. However, they also propose initial approaches for classifying different methods to evaluate interpretability. In addition to two approaches that involve assessments by human experts, they describe the concept of functionally-grounded evaluation, which does not require human judgment. This method, however, relies on the availability of validated data or results that can be used as a benchmark for evaluating explanations (*Doshi-Velez & Kim*, 2017). In this context, *Guidotti* (2021) states, that if the actual reasons for a classification result are known – i.e., the ground truth is available – then the evaluation of local explanations can be conducted technically, rather than relying solely on human experts.

To establish what qualifies as a good explanation, a clear evaluation framework must be defined. To enable comparability between the approaches – both in terms of normalized versus non-normalized data and between the two explanation methods LIME and Shapley Values – absolute values are not used. Instead, the values are transformed into a ranking, where the feature with the highest effect, according to LIME or Shapley Values, is assigned Rank 1, followed by the remaining features with decreasing ranks down to Rank 28 for the features contributing least to a classification as being misstated. In a medical context, explanations provided by LIME, for example, have been evaluated using such a ranking approach. While no ground truth was available, physicians identified and ranked the most important driver features for a medical diagnosis, which were then compared with the explanations provided by LIME to assess the extent of overlap between the two and thus, the quality of the explanations (*Kumarakulasinghe et al.*, 2020). Applied to the present use case, to assess whether explanations provided by LIME or Shapley Values can be helpful in identifying manipulated areas within financial statements, it is necessary to establish a connection between the features

and the manipulated areas. This enables the matching or comparison between the explanations of individual features and the ground truth in the form of the misstated accounts within the financial statements.

Table D-11: Matching of misstatement type and related financial data items

Account category affected	Variable	Related financial items
Misstated revenue	c_rev	sale
Misstated accounts receivable	c_rec	rect
Misstated cost of goods sold	c_cogs	cogs
Misstated inventory	c_inv	invt
Misstated reserve account	c_reserve	ceq
Misstated marketable securities	c_mkt_sec	ivst
Misstated payables	c_pay	ap
Misstated allowance for bad debt	c_debt	*n/a*
Capitalized costs as asset	c_asset	act, at, che, ivao, ivst, ppegt
Misstated liabilities	c_liab	dlc, dltis, dltt, lct, lt, txp
Misstatement of other expense⁄shareholder equity account	c_inc_exp_se	ceq, csho, dp, ib, ni, pstk, re, sstk, txt, xint, prcc_f

The table lists the types of misstatements as categorized by Dechow et al. (2011) in their AAER database. Similar to Bao et al. (2020), I conducted a matching between the types of misstatements and the financial data items used in the models. However, my approach differs in certain assignments, as I adopt a more conservative stance. For example, in the case of Misstated Revenues and Misstated Accounts Receivable, I do not consider financial data items such as Net Income (Loss) (ni) and Retained Earnings (re). For the purposes of this analysis, the informational value of these aggregate figures regarding the actual manipulation – here specifically, Revenues or Accounts Receivable – would be too limited to provide meaningful insights.

Table D-11 illustrates the matching of misstated account categories and financial data items used for the models' training. To perform this comparison, the features are assigned to categories that best correspond to areas such as the balance sheet or income statement. For example, the categorization within the AAER dataset by *Dechow et al.* (2011) includes the class "Misstated Revenues." Among the financial data items from Compustat, Sales/Turnover (Net) (sale) is considered the most relevant as a potentially manipulated variable in this category.

Thus, this assignment is made. If a "Misstated Revenue" case occurs, a strong contribution from Sales/Turnover (Net) (sale) would be the best indicator of the specific area in which manipulation may have occurred. For this reason, broader or derived totals, such as Net Income (Loss), are explicitly excluded from this assignment. For most categories, such an assignment was relatively straightforward with regard to the features used. However, the AAER dataset also includes categories that are less specific. For these, precise assignments are not possible, and multiple features are considered. For example, the broadly defined category "Capitalized costs as asset" (asset) relates to multiple items associated with asset positions, such as Current Assets, Total (act), Assets, Total (at), or Property, Plant and Equipment, Total (ppegt). As a result, categories with a single assigned feature can be compared effectively, but comparisons with categories containing multiple assigned features are significantly limited. This is due to two main factors: First, multiple variables inherently increase the likelihood that one of them will randomly rank highly, and second, a manipulation might occur in, for example, a current assets position and thus be categorized as “Capitalized costs as assets”, but a strong explanation contribution from Property, Plant, and Equipment, Total (ppegt) might be misinterpreted as a "good" explanation since it is assigned to “Capitalized costs as assets”. Therefore, the last three categories should be interpreted with great caution and compared only with categories that have the same number of assigned features. For this reason, the categories with different numbers of assigned features are visually separated in the subsequent presentation.

6.3. Detected and Undetected Fraud Cases

For the actual misstatements, two different outputs from the classification models are generally possible. Either a model correctly classifies a misstatement observation as a true positive, or it incorrectly fails to classify the observation as a misstatement, resulting in a false negative prediction. For all cases of true positives and false negatives, I derived local explanations using both LIME and Shapley Values. As explained in the previous section, these explanations were transformed into ranks from 1 to 28 based on their effect sizes to enhance comparability between the approaches. The following results are discussed in the context of the two questions posed for the true positives and false negatives:

RQ3: With regard to true positive predictions, i.e., detected misstatements: Do features which are related to a certain type of misstatement contribute to the classification as being misstated?

RQ4: With regard to false negative predictions, i.e., undetected misstatements: Despite their incorrect classification, do features which are related to a certain type of misstatement contribute to the classification as being misstated?

If true positive predictions were significantly driven by the features related to the type of misstatement, the explanations could provide a targeted indication for identifying the manipulation. In case of comparably contributions by related features for false negative predictions, this could indicate that the functioning of the classification models is indeed driven by the manipulated positions of a financial statement and is therefore genuinely aligned with the objectives for which they have been trained, even if the overall threshold for a classification as being misstated would not have been met.

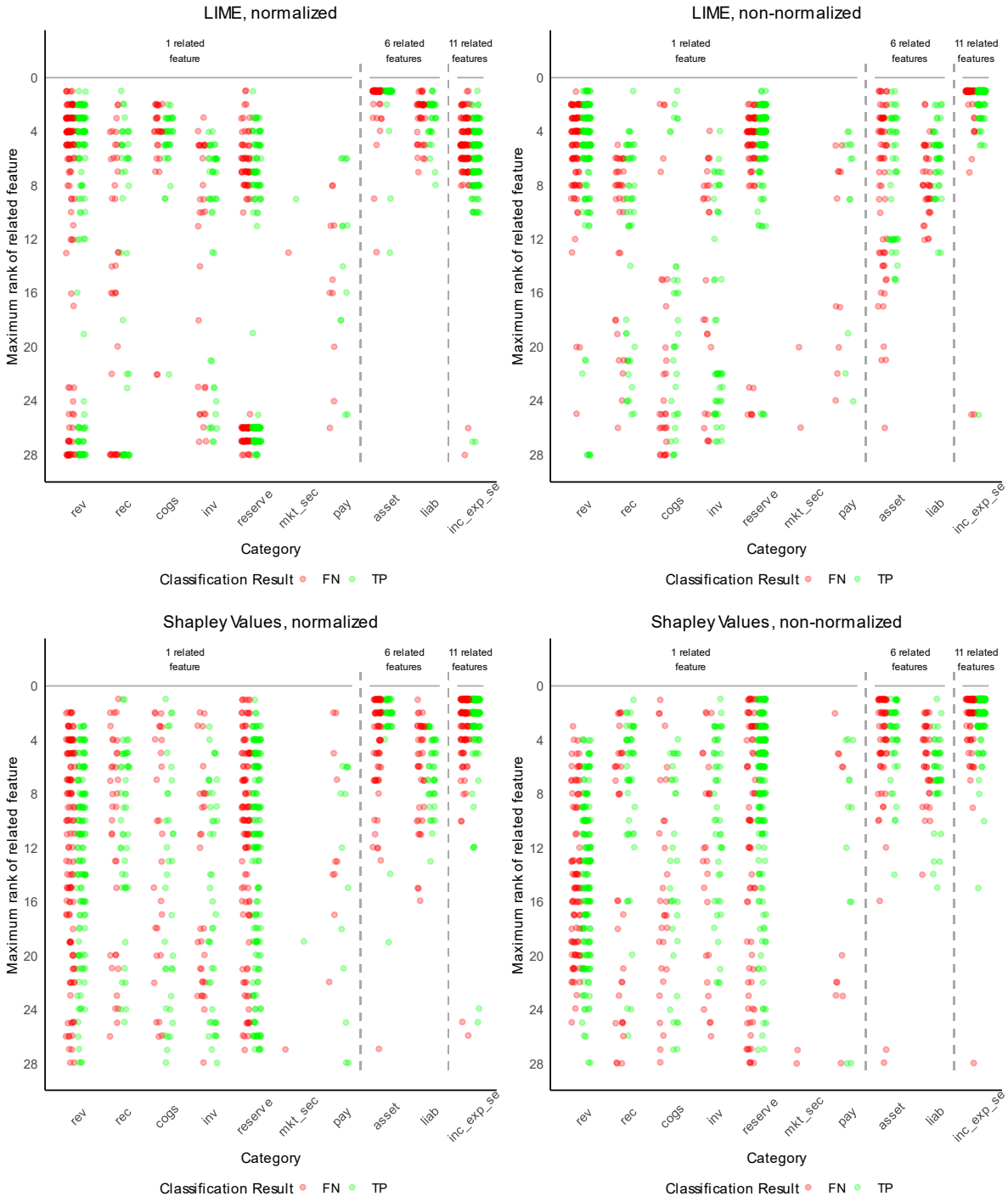
Figure D-8 illustrates the explanation ranks for all true positive and false negative predictions for four different variants. A conceptually identical figure, but presented using boxplots, can be found in Appendix I. The advantage of the chosen representation using jitter plots lies primarily in the improved visibility of how individual data points are distributed, making it easier to identify any potentially emerging patterns. Thereby, two panels each present the explanations for the classification models trained on normalized and non-normalized data. Additionally, two of these panels illustrate the explanations provided by LIME and Shapley Values, respectively. The abscissa represents the types of misstatements. As previously described in section D.6.2, categories with varying numbers of assigned features should be interpreted with caution and are clearly separated with vertical dotted lines from the other categories with the reference to the number of assigned features above. To clarify once again, a direct comparability in Figure D-8 is only possible between misstatement types that have been assigned the same number of features. This is because, in cases where a misstatement type is associated with multiple features, the probability that at least one of these features' explanation will have a high effect and therefore a high corresponding rank – even by chance – is naturally higher than in cases where only a single feature is assigned. If this were not taken into account, misstatement types with multiple assigned features would inherently appear as the better-explained types of fraud. Therefore, the interpretation of results regarding which misstatement types are better or worse explained should be focused within each group that has the same number of related features. The quality of an explanation is presented on the ordinate as the highest rank of an assigned feature of the certain type of misstatement. For misstatement types with only one assigned feature, this corresponds to its rank among the explanations of all 28

features. For categories with multiple assigned features, the highest rank of these features in relation to all 28 features is presented.

First, it is notable that the distributions of the green-marked true positives and the red-marked false negatives are relatively similar across all variants. In isolation, this could suggest that the explanations work equally well (or poorly) in both cases, with the false negatives primarily resulting from the classification threshold simply not being exceeded. However, it is also important to consider how the explanations behave in cases such as false positives and whether these differ significantly from or align with the patterns observed here. To address this, the following discussion will also reference Figure D-9, which will be revisited during the separate analysis of false positives.

Starting with the first type of misstatement, the "Misstatement of other expense/shareholder equity account" (inc_exp_se), as the most diverse category in terms of content it has the highest number of assigned features (11). This category consistently records a very high maximum rank for at least one assigned feature across all variants. Since the large number of assigned features inherently increases the likelihood that any one of these features will have a high rank, the interpretive value is significantly limited, and a differing result would have been contrary to expectations.

Figure D-8: Highest ranks of explanations depending on the type of misstatement



Each panel illustrates the highest rank of a feature's explanation which is related to the type of misstatement of the analyzed observation. As the first six categories are matched to a single feature, "asset" and "liability" in contrast are matched with 6 features and "inc_exp_se" with 11 features. The category "debt" is not included as there is no misstatement case for this type between 2003 and 2019 in the database.

When considering the misstatement types as groups based on the number of assigned features, as visually separated in Figure D-8 by the vertical dotted lines, misstated assets and liabilities form the second internally comparable group. The misstatement types "Capitalized costs as assets" (asset) and "Misstated liabilities" (liab) each have 6 assigned features. Here too, the large number of features results in a high level of the maximum rank, which is generally in line with expectations. Nevertheless, the maximum ranks for misstated assets generally appear to score slightly higher compared to misstated liabilities in most cases. The only exception is the LIME (non-normalized) variant, which exhibits greater variability in the maximum rank of the assigned features. This could initially suggest that the classification as a misstatement in the case of misstated assets is more likely driven by asset-related positions than is the case for misstated liabilities. Furthermore, this effect appears to be more pronounced in the model variants with normalized data. However, given that both categories have six assigned features, a considerable degree of uncertainty remains. Furthermore, it is important to consider how other predictions are explained by LIME and Shapley Values and to what extent these explanations differ from those in cases of actual misstatements. For instance, regarding the seemingly strongest explanatory contributions for misstated assets in the case of the LIME (normalized) variant, Figure D-9 reveals that "Assets, Total" also ranks first in the vast majority of cases involving false positive predictions. This suggests that, rather than providing a precise, targeted explanation in cases of actual misstatements in this area, this feature may be driving the classification as a misstatement regardless of whether a misstatement is actually present in the observations.

In contrast to the previously discussed groups of categories with more than one assigned feature to the type of misstatement, the explanatory power of the ranks in categories with only one assigned feature is far less limited. In these cases, the ranks of the specific feature assigned in Table D-11 are directly represented in Figure D-8. When comparing the panels of the four variants, the following observation stands out: The maximum ranks for explanations generated by Shapley Values seem to be more dispersed than those generated by LIME. However, the boxplots in Appendix I, with their interquartile ranges extending far downward in the form of the actual boxes, also show that for the categories “Misstated revenue" (rev), “Misstated accounts receivable” (rec), “Misstated inventory” (inv), and “Misstated reserve account” (reserve), many data points are situated at the lower end of the rank spectrum in case of LIME (normalized). This suggests that while these ranks may appear less dispersed in the case of LIME (normalized) at first glance, they tend to exhibit extreme values at both ends of the rank range. Overall, the Shapley Values appear to be so widely dispersed that no strong connection between the area of manipulation and the contribution of the related feature to the classification as a misstatement seems evident. Consequently, for none of the categories it can be confidently asserted that the driving effects of the respective feature provide a reliable indication of that specific type of misstatement, and thus, serve as a targeted guidance for potential users’ further investigations.

A more differentiated consideration is required for the two upper panels representing the explanations generated by LIME. In the LIME (normalized) panel, for the three categories “Misstated revenue” (rev), “Misstated accounts receivable” (rec), and “Misstated reserve account” (reserve), the explanations, while tending toward both extremes, also show a

significant concentration of ranks in the upper range. This pattern extends to the additional categories “Misstated costs of goods sold” (cogs) and, to some extent, “Misstated inventory” (inv), where many explanations are found at higher ranks. This holds true for both true positive and false negative predictions. In direct comparison, the LIME (non-normalized) panel reveals that, while the explanations for “Misstated costs of goods sold” (cogs) are notably less accurate, the explanations for “Misstated revenue” (rev), “Misstated accounts receivable” (rec), and “Misstated reserve account” (reserve) are the most accurate across all four panels. For these categories, nearly all explanations rank within the top 10, and particularly for “Misstated revenue” (rev) and “Misstated reserve account” (reserve), the explanations effectively identify the driving features, such as Sales/Turnover (Net) (sale) for “Misstated revenue” (rev). In these cases, the explanations could provide a strong initial indication and a valuable starting point for a more focused and in-depth investigation by potential users. However, it is essential to also consider the decision-making behavior of the models with regard to the explanations in cases of false positives to provide a conclusive assessment of the capabilities and usefulness of these approaches in this application.

6.4. Misclassified Non-Fraudulent Cases

To better contextualize the previous insights regarding the explanations of actual misstatement cases and to identify potential biases in the models based on the explanations, a more detailed analysis of the false positive predictions and their explanations is necessary. The following results are discussed therefore discussed against the findings of the previous section and in the context of the following question posed for the false positives:

RQ5: With regard to false positive predictions, i.e., false alarms in the absence of an actual misstatement: Do the distributions of the explanations provide indications that potential biases influence the predictions in a way that does not align with the original training objective?

In the trained models, false positive predictions are classifications in which the model identifies an observation as fraudulent, even though no fraud was identified in the underlying observation according to the AAER dataset. These predictions can therefore be understood as false alarms, mistakenly indicating a risk. Unlike Figure D-8, Figure D-9 does not include a breakdown by types of misstatements, as no such types are available in the case of false positives because no misstatement was observed. However, besides the explanations for false positive predictions, the previous explanations of the cases with actual misstatements are included as well.

Accordingly, Figure D-9 contains the four panels for the different model variants, each showing explanations provided by LIME and Shapley Values. The ordinate represents all features, while the abscissa plots the rank of the explanation for each feature. This means each analyzed observation is represented by one data point per feature, resulting in a total of 28 data points per panel. Given the high computational cost, it is impractical to analyze all false

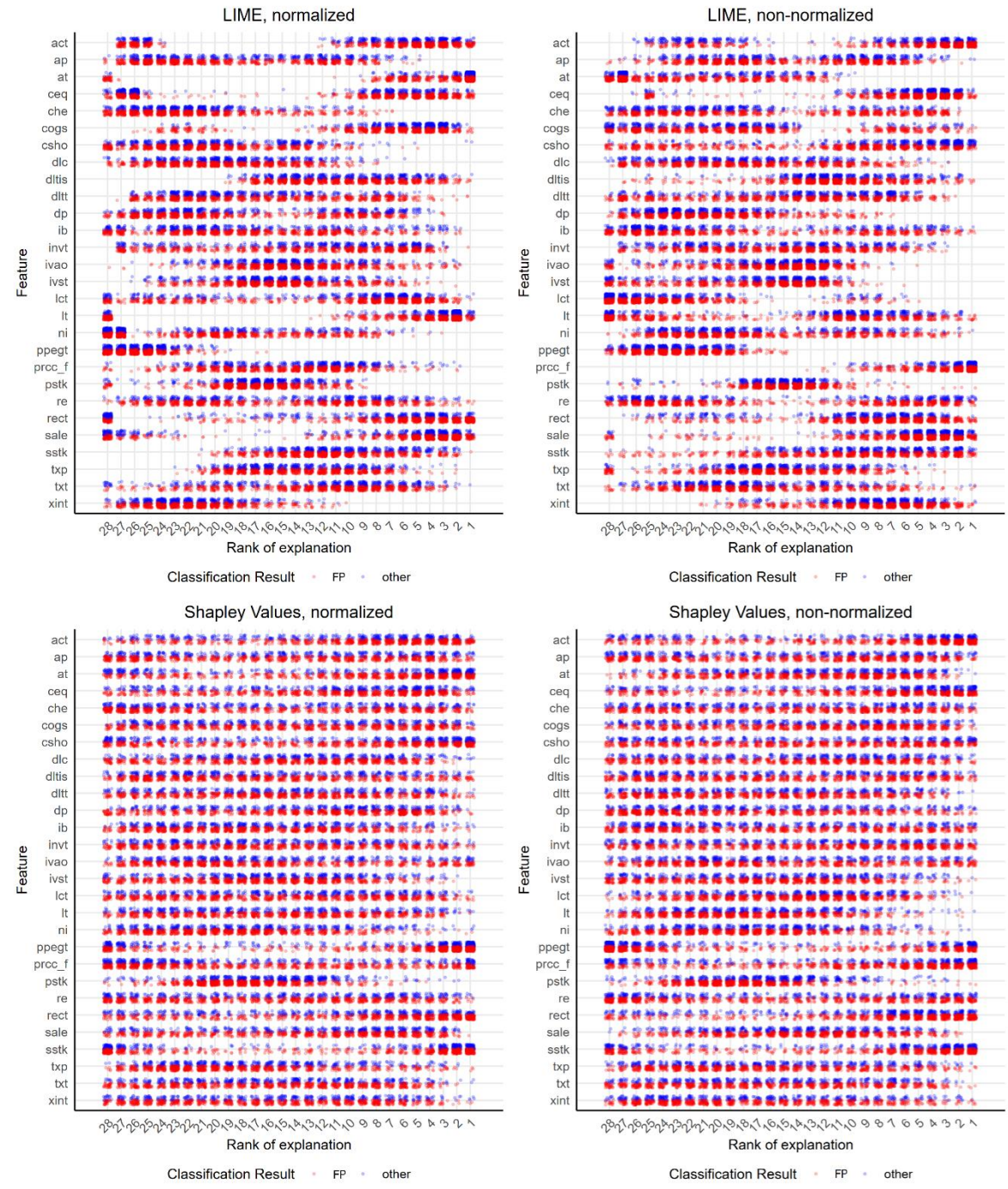
positives, as the required processing time would be disproportionate. For the analysis of false positives, I randomly selected 50 false positives for each test year and each of the four variants. The ranks of the explanations for the false positives are shown in red, while the ranks for the previously analyzed true positives and false negatives are combined as blue data points.

Figure D-9 highlights a notable aspect: The differing distribution of ranks between LIME and Shapley Values. The panels for Shapley Values show a very broad distribution of ranks across nearly all features. In contrast, the ranks for LIME explanations tend to exhibit certain structures in their distributions, with some features generally tending toward higher or lower ranks overall. Additionally, it can be observed that these differing distributions apply to both the actual misstatements, shown in blue, and the false positives, shown in red. This means that for the ranks of explanations provided by Shapley Values, both false positives and actual misstatement cases are broadly dispersed with almost no discernible patterns. In contrast, for LIME, the features tend to exhibit a more consistent rank distribution across observations, regardless of the classification outcomes. This observation of the differently pronounced distributions of ranks is similarly reflected in Figure D-8 with regard to the true positives and false negatives. This leads to further qualification of the seemingly well-explained cases by LIME (normalized) as shown in Figure D-8. The features Sales/Turnover (Net) (sale), Receivables, Total (rect), and Common/Ordinary Equity, Total (ceq), assigned to the misstatement types “Misstated revenue” (rev), “Misstated accounts receivable” (rec), and “Misstated reserve account” (reserve), consistently achieve high ranks not only for actual misstatements but also explicitly for false positive predictions. This indicates that while these explanations may align with the selected types of misstatements, they do so not because of a

unique connection to these cases but because the explanations also commonly appear for false positives. Based on these closer examinations of the categories and their related features, an important consideration arises for potential user groups. Although the explanations do not directly indicate whether particularly low or high values of the features drive the classification, an aggregated view of LIME explanations suggests a concerning trend. If features such as Sales/Turnover (sale), Receivables, Total (rect), and Common/Ordinary Equity, Total (ceq) drive the classification as misstatements regardless of the actual presence of misstatements, the models may be biased toward classifying companies with, for instance, high revenues, correspondingly higher receivables, or better equity positions as potential misstatements. While this cannot be conclusively proven, such a tendency could have significant implications, particularly for investors. If they rely on these models, they may risk excluding well-performing companies from their portfolios, potentially leading to adverse outcomes.

A final aspect that also limits the reliability of the local explanations is the similarity of the explanations. While I have only compared two cases explicitly across both model variants and the two explanation approaches in section D.5.4, one of these cases seemed to be relatively well explained in all four instances. However, this does not align with the distributions observed in Figure D-8 and Figure D-9. If the explanations provided by LIME and Shapley Values followed more similar distributions, this might suggest more reliable explanations. In conclusion, the reliability of local explanations for identifying manipulated areas in financial statements must be assessed very critically for the cases and approaches analyzed here. Even for the initially more promising explanations for certain types of manipulations, particularly through LIME, the risk of potential model biases must be considered.

Figure D-9: Distribution of features' ranks for false positive predictions



The figure illustrates the explanation ranks of all features and provides an overall overview of the explanations' distributions. The explanation ranks of false positive predictions are contrasted in red from the other analyzed classification results, here covering observations with actual misstatements in form of true positive and false negative predictions.

E. Conclusion

1. Summary of the Main Findings and Contributions

The starting point of this research lies in the field of machine-learning-based financial statement fraud detection models. This area of research is characterized by classification results that have not yet reached a level of cost-efficient performance that would enable widespread practical application (*Beneish & Vorst*, 2022). For a long time, the primary focus has been on improving the classification performance of these models. However, there appear to be inherent technical limitations, partly due to the rarity of fraud cases available for training. In contrast, the focus of machine learning research has increasingly shifted toward interpretability. Initial exploratory approaches have been presented in the context of financial statement fraud detection (*Craja et al.*, 2020; *Zhang*, C. *et al.*, 2022). However, it has not yet been examined whether explanations provided by interpretable machine learning methods can actually offer useful insights in the classification process.

Building on this, I first examined the conceptual framework to determine whether there is a fundamental demand for interpretable results – particularly local explanations – among potential user groups. I contribute to this by analyzing the legal and organizational conditions within which the primary user groups – auditors, enforcement authorities, and investors – operate. The analysis indicates a fundamentally high demand for reliable systems capable of providing local explanations for individual classification decisions. While this demand appears to be lowest for investors, it could be particularly valuable for enforcement authorities and auditors. For these groups, such tools could facilitate more efficient resource allocation and enable the earlier and more effective detection of fraud cases. In the context of

black-box models, legal certainty and the necessary transparency for auditors represent a significant barrier. This challenge could potentially be addressed through reliable local explanations.

After establishing the fundamental need and potential significance of reliable local explanations for model predictions, I analyzed, using selected and established approaches of financial statement fraud detection models (*Bao et al.*, 2020), the extent to which local explanations can genuinely provide targeted insights or be leveraged for human plausibility assessments with appropriate expertise. For this purpose, I trained cost-sensitive models using both normalized and non-normalized data. Additionally, considering potential temporal changes in the financial data itself as well as in the nature of manipulations, I employed two different training approaches: One using a fixed number of periods prior to the test year (rolling window) and the other using all available historical data before the respective test years (rolling origin retrain). I demonstrated that the classification performance in the rolling window approach did not systematically lag behind the performance achieved when incorporating all available data. This finding can be particularly significant for scenarios requiring computationally intensive analyses – especially in the context of interpretable machine learning, where computational efficiency plays a critical role.

There is currently relatively little guidance on evaluating interpretability. To the best of my knowledge, this research is the first to conduct a comprehensive analysis of local explanations for financial statement fraud detection models. For this purpose, I used LIME and Shapley Values, two established approaches capable of generating local explanations independently of the underlying model. In addition to illustrating these explanations through

selected examples, I performed a matching between the type of misstatement and the dataset features that could be subject to manipulation or at least fall within the relevant area. If the local explanations for these features in a given fraud case contributed significantly to the classification, user groups could derive targeted insights for further investigations. The analysis of different explanation approaches and types of fraud yielded highly constrained results. First, LIME and Shapley Values tended to produce differing explanations. Second, fraud types where explanations appeared promising must be reconsidered in light of the additional analysis of False Positives. In cases of misstated observations, the associated variables did not exclusively drive decisions in the correct direction. Instead, similar behavior was observed in the absence of misstatements. This points, on the one hand, to potential biases in the underlying models. On the other hand, the explanations exhibited such broad variability that it is difficult to view them as genuinely targeted and useful. This variability further undermines their reliability. While there is clearly a need for interpretable local predictions in this application, my findings contribute to the understanding that individual local explanations, in particular, must be approached with great caution. Although they are increasingly presented in research as illustrative examples, this analysis – based on the selected models and explanation approaches – demonstrates that the explanations in this specific context are (as yet) unable to provide reliable indications.

2. Implications for Business Practice

The implications for business practice must be differentiated based on both the conceptual and analytical findings and according to the potential user groups.

For auditors, regulatory requirements impose a high obligation for transparent and well-documented processes. Justifications must be comprehensible so that third parties can understand the decisions made and the resulting audit process. However, the high degree of reliability required for local explanations to justify specific decisions in the audit program could not be established using the analyzed approaches in this study. From a conceptual perspective, the auditing profession should continue working toward ensuring that the legally sound application of machine-learning-based systems is appropriately covered by auditing standards. In practical application, however, particular caution must be exercised when using exemplary explanations derived from interpretable machine learning approaches. Only through testing on a larger scale can meaningful conclusions be drawn about the actual accuracy and reliability of explanations provided by black-box models.

For enforcement authorities, which select companies to audit based on concrete or abstract risks – at least in Germany – the regulatory framework is more conducive to the application of such models for risk identification in financial statements. In this context, the requirement for a detailed and transparent justification of decisions is less stringent than in the case of auditors. Nevertheless, the same caution applies to the potential use of local explanations: they should only be adopted after extensive and successful testing. Relying on seemingly good explanations based solely on isolated examples would be insufficient and potentially misleading.

For investors, an additional aspect comes into play. The analyzed approaches were unable to provide reliable explanations for identifying manipulations within financial statements. However, certain patterns in the explanations – particularly when examining false

positives – suggest that the models may contain biases. If these biases tend to classify well-performing companies as misstated, this could result in their exclusion from investment portfolios. Consequently, investors could face significant opportunity costs in the form of forgone profits.

Overall, the findings show that, at least with regard to the approach analyzed here, locally generated explanations – when selected as isolated examples – can create the impression of being good and plausible. However, when viewed as a whole, models and the explanations derived from them may still be biased, leading to a deceptive sense of reliability if only exemplary explanations are highlighted. *Gu*, Y. *et al.* (2024, p. 9) succinctly put it, "It's Not Intelligence; It's Functionality". While the increasing use of machine-learning models will undoubtedly lead to significant changes, their application must always focus on functionality and actual performance. These models must therefore be critically questioned and evaluated in light of their true capabilities.

3. Limitations

As in most cases of research, the findings must be viewed in light of existing limitations. These limitations apply both to the conceptual part, which analyzes the environment in which potential user groups operate and their general need for locally interpretable model explanations in order to implement models in a cost-efficient way, as well as to the application of selected local explanation approaches to the trained models.

Findings of the conceptual section of this research are limited to accounting and information systems literature as well as relevant standards and legislation. Concerning the legal level and here considered ISA, which are directly or indirectly applicable in most

jurisdictions, only high-level EU legislation and exemplary German implementation of enforcement have been included.

The practical implementation of the explanation approaches is subject to the following limitations. The foundation of the analyzed model is based on a literature-recognized approach for identifying misstatements. Specifically, the RUSBoost algorithm and a theory-driven feature selection process were used for model training (*Bao et al.*, 2020). However, explanations for models trained with other algorithms were not analyzed. Furthermore, the explanations are limited to the two approaches, LIME and Shapley Values. To assess the quality of the explanations, I used the types of misstatements from the AAER dataset by *Dechow et al.* (2011). These types are restricted to categories of misstated accounts, with each category potentially containing a varying number of assignable financial items from the financial statements. For this reason, and because a fraud case can involve a varying number of manipulated areas, it is not possible to determine a directly comparable score across all cases. Instead, the evaluation was primarily conducted separately for each type of misstatement, particularly focusing on the distribution of the ranks of the assigned explanations by the individual features. In particular, the features incorporated into the training process are limited to publicly available financial data items derived from disclosed financial statements from the US. These do explicitly not include internal financial data as, e.g., a firm's data of its general ledger. And last, from a technical perspective, there may also be limitations due to the available computational capacity.

4. Avenues for Future Research

While this research indicates significant limitations regarding the potential for interpretability of predictions in the application area of machine learning-based financial statement fraud detection, it also offers several avenues for future research to build on both the conceptual and applied analytical findings.

In addition to the conceptual analysis of the legal and organizational conditions of potential user groups, I encourage research on an interview basis and survey-based covering multiple perspectives. I further encourage broadening research on enforcement, particularly outside the EU. Regarding the analysis of the applied explanation approaches, I encourage expanding future research to include, first, additional models trained with different algorithms, and second, further explanation approaches beyond LIME and Shapley Values. With regard to the evaluation of explanations, measures of similarity between different explanations could provide further insights into their reliability (*Gwinner et al.*, 2024). Lastly, the human component in the evaluation of explanations should not be underestimated. For the evaluation of design science research, which explicitly focuses on systems addressing practical real-world problems, potential users should be involved in future research through a naturalistic evaluation to assess the quality and usefulness of the generated explanations (*Doshi-Velez & Kim*, 2017; *Sutton et al.*, 2021; *Venable et al.*, 2016). One approach to further enhancing the user-friendliness of these explanations in general could be the integration of LLM-generated, i.e. by Large Language Models, narrative explanations (*Martens et al.*, 2025).

Overall, this research operates within a rapidly evolving environment – one that is driven both by technological advancements and regulatory developments. From a regulatory

perspective, reporting requirements are increasingly expanding to include ESG reporting, which is expected to assume a comparable role to financial reporting in the future. Accordingly, similar questions arise in this still largely separate area regarding the potential use of AI for efficient and effective ESG assurance (*Li et al.*, 2024). Significant potential and new research approaches may emerge as financial and ESG information become increasingly integrated, creating new opportunities for in-depth research. Furthermore, the use of large language models remains in its early stages. These models could basically also hold potential for identifying anomalies, among other applications, but their practical integration into financial statement analysis is still developing (*Gu*, H. *et al.*, 2024).

In light of these changes, another area remains largely unaddressed, having only been considered in isolated cases: How to respond, "When a Machine Is Listening", as the increasing use of machine learning in financial statement analysis is itself anticipated and countered. This perspective opens up further critical questions about the evolving dynamics between machine learning-based detection systems and the strategies used to evade them (*Cao et al.*, 2023). This challenge is further compounded by the fact that such models are often black-box models, which are not inherently explainable. Therefore, the functionality and the explanations of these models, considered in their entirety, will become even more critical in the future. Relying solely on individual exemplary explanations without accounting for the overall functionality risks leading to misleading conclusions.

Or in the words of Heinrich Heine: Hitting the mark once does not make one a marksman (*Leonhardt*, 2015, p. 107, translation by author).

References

Academic and Professional References

Abbasi, A., Albrecht, C., Vance, A., & Hansen, J. (2012). MetaFraud: A Meta-Learning Framework for Detecting Financial Fraud. *MIS Quarterly*, *36*(4), 1293–1327.

Aboud, A., & Robinson, B. (2022). Fraudulent financial reporting and data analytics: an explanatory study from Ireland. *Accounting Research Journal*, *35*(1), 21–36.

Adadi, A., & Berrada, M. (2018). Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, *6*, 52138–52160.

Akerlof, G. A. (1970). The Market for "Lemons": Quality Uncertainty and the Market Mechanism. *The Quarterly Journal of Economics*, *84*(3), 488–500.

Albrecht, W. S., Albrecht, C. O., Albrecht, C. C., & Zimbelman, M. F. (2016). Fraud Examination (5th ed.), Cengage Learning.

Albrecht, W. S., Howe, K. R., & Romney, M. B. (1984). Deterring fraud: the internal auditor's perspective.

Alden, M. E., Bryan, D. M., Lessley, B. J., & Tripathy, A. (2012). Detection of Financial Statement Fraud Using Evolutionary Algorithms. *Journal of Emerging Technologies in Accounting*, *9*(1), 71–94.

Alfaro, E., Gámez, M., & García, N. (2013). adabag: An R Package for Classification with Boosting and Bagging. *Journal of Statistical Software*, *54*(2), 1–35.

Altman, E. I. (1968). Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. *The Journal of Finance*, *23*(4), 589–609.

American Institute of Certified Public Accountants, & Chartered Professional Accountants of Canada. (2020). The Data-Driven Audit: How Automation and AI are Changing the Audit and the Role of the Auditor, accessed 30.08.2022: https://www.aicpa.org/content/dam/aicpa/interestareas/frc/assuranceadvisoryservices/downloadabledocuments/the-data-driven-audit.pdf.

Anderson, C. (2008). The end of theory: The data deluge makes the scientific method obsolete. *Wired Magazine*, *16*(7), 16-07.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115.

Association of Certified Fraud Examiners. (2024). Occupational Fraud 2024: A Report to the Nations, accessed 22.06.2024: https://www.acfe.com/-/media/files/acfe/pdfs/rttn/2024/2024-report-to-the-nations.pdf.

Bai, B., Yen, J., & Yang, X. (2008). False financial statements: Characteristics of China's listed companies and cart detecting approach. *International Journal of Information Technology and Decision Making*, *7*(2), 339–359.

Baldwin, D., & Yadav, S. B. (1995). The process of research investigations in artificial intelligence-a unified view. *IEEE Transactions on Systems, Man, and Cybernetics*, *25*(5), 852–861.

Bao, Y., Ke, B., Li, B., Yu, Y. J., & Zhang, J. (2020). Detecting Accounting Fraud in Publicly Traded U.S. Firms Using a Machine Learning Approach. *Journal of Accounting Research*, *58*(1), 199-235.

Bayer, S., Gimpel, H., & Markgraf, M. (2022). The role of domain expertise in trusting and following explainable AI decision support systems. *Journal of Decision Systems*, *32*(1), 110–138.

Beasley, M. S. (1996). An Empirical Analysis of the Relation between the Board of Director Composition and Financial Statement Fraud. *The Accounting Review*, *71*(4), 443–465.

Bedué, P., & Fritzsche, A. (2022). Can we trust AI? An empirical investigation of trust requirements and guide to successful AI adoption. *Journal of Enterprise Information Management*, *35*(2), 530–549.

Bender, R., Fröndhoff, B., & Nagel, L.-M. (2022, August 29). Wirtschaftsprüfer verzweifelt gesucht: Adler braucht einen neuen Wirtschaftsprüfer, findet aber keinen. Die Zeit drängt. In einem Bittbrief ködert der Konzern die Prüfer nun mit Sonderrechten. *Handelsblatt*, (166), p. 30.

Beneish, M. D. (1997). Detecting GAAP violation: implications for assessing earnings management among firms with extreme financial performance. *Journal of Accounting and Public Policy*, *16*(3), 271–309.

Beneish, M. D. (1999a). The Detection of Earnings Manipulation. *Financial Analysts Journal*, *55*(5), 24–36.

Beneish, M. D. (1999b). Incentives and Penalties Related to Earnings Overstatements that Violate GAAP. *The Accounting Review*, *74*(4), 425–457.

Beneish, M. D., & Vorst, P. (2022). The Cost of Fraud Prediction Errors. *The Accounting Review*, *97*(6), 91–121.

Benford, F. (1938). The Law of Anomalous Numbers. *Proceedings of the American Philosophical Society*, *78*(4), 551–572.

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192–213.

Bertomeu, J., Cheynel, E., Floyd, E., & Pan, W. (2021). Using machine learning to detect misstatements. *Review of Accounting Studies*, *26*(2), 468–519.

Bhattacharya, I., & Mickovic, A. (2024). Accounting fraud detection using contextual language learning. *International Journal of Accounting Information Systems*, *53*, 100682.

Biondi, Y., Bloomfield, R. J., Glover, J. C., Jamal, K., Ohlson, J. A., Penman, S. H., Tsujiyama, E., & Wilks, T. J. (2011). A Perspective on the Joint IASB/FASB Exposure Draft on Accounting for Leases. *Accounting Horizons*, *25*(4), 861–871.

Bird, A., Karolyi, S. A., & Ruchti, T. G. (2019). Understanding the "numbers game". *Journal of Accounting and Economics*, *68*(2–3), 101242.

Blocher, E., & Cooper, J. C. (1988). A Study of Auditors' Analytical Review Performance. *Auditing: A Journal of Practice & Theory*, *7*(2), 1–28.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. K. (1987). Occam's Razor. *Information Processing Letters*, *24*, 377–380.

Bonrath, A., & Eulerich, M. (2024). Internal auditing's role in preventing and detecting fraud: An empirical analysis. *International Journal of Auditing*, *28*(4), 615–631.

Booker, A., Chiu, V., Groff, N., & Richardson, V. J. (2024). AIS research opportunities utilizing Machine Learning: From a Meta-Theory of accounting literature. *International Journal of Accounting Information Systems*, *52*, 100661.

Boritz, J. E., & No, W. G. (2020). How Significant are the Differences in Financial Data Provided by Key Data Sources? A Comparison of XBRL, Compustat, Yahoo! Finance, and Google Finance. *Journal of Information Systems*, *34*(3), 47–75.

Box, G. E. P. (1976). Science and Statistics. *Journal of the American Statistical Association*, *71*(356), 791–799.

Bravidor, M., Förster, G., & Weißenberger, B. E. (2020). Berufsstand 4.0: Wirtschaftsprüfer und Steuerberater zwischen Berufsstand 4.0: Wirtschaftsprüfer und Steuerberater zwischen Fachexpertise, IT und Datenanalyse Fachexpertise, IT und Datenanalyse. *Die Wirtschaftsprüfung*, *73*(5), 287–294.

Brazel, J. F., Jones, K. L., Thayer, J., & Warne, R. C. (2015). Understanding investor perceptions of financial statement fraud and their use of red flags: evidence from the field. *Review of Accounting Studies*, *20*(4), 1373–1406.

Brown, N. C., Crowley, R. M., & Elliott, W. B. (2020). What Are You Saying? Using topic to Detect Financial Misreporting. *Journal of Accounting Research*, *58*(1), 237-291.

Cai, S., & Xie, Z. (2024). Explainable fraud detection of financial statement data driven by two-layer knowledge graph. *Expert Systems with Applications*, *246*, 123126.

Canadian Public Accountability Board. (2021). Technology in the audit, accessed 24.08.2022: https://cpab-ccrc.ca/docs/default-source/thought-leadership-publications/2021-technology-audit-en.pdf?sfvrsn=f29b51ce_14.

Cao, S., Jiang, W., Yang, B., & Zhang, A. L. (2023). How to Talk When a Machine Is Listening: Corporate Disclosure in the Age of AI. *The Review of Financial Studies*, *36*(9), 3603–3642.

Carnes, R. R., Christensen, D. M., & Madsen, P. E. (2023). Externalities of Financial Statement Fraud on the Incoming Accounting Labor Force. *Journal of Accounting Research*, *61*(5), 1531–1589.

Castelvecchi, D. (2016). Can we open the black box of AI? *Nature*, *538*(7623), 20–23.

Cecchini, M., Aytug, H., Koehler, G. J., & Pathak, P. (2010a). Detecting Management Fraud in Public Companies. *Management Science*, *56*(7), 1146–1160.

Cecchini, M., Aytug, H., Koehler, G. J., & Pathak, P. (2010b). Making words work: Using financial text as a predictor of financial events. *Decision Support Systems*, *50*(1), 164–175.

Chakrabarty, B., Moulton, P. C., Pugachev, L., & Wang, X. (2024). Catch me if you can: In search of accuracy, scope, and ease of fraud prediction. *Review of Accounting Studies*, *forthcoming*.

Chen, Y.-J., Liou, W.-C., Chen, Y.-M., & Wu, J.-H. (2019). Fraud detection for financial statements of business groups. *International Journal of Accounting Information Systems*, *32*, 1–23.

Chen, Y.-J., Wu, C.-H., Chen, Y.-M., Li, H.-Y., & Chen, H.-K. (2017). Enhancement of fraud detection for narratives in annual reports. *International Journal of Accounting Information Systems*, *26*, 32–45.

Cheynel, E., Cianciaruso, D., & Zhou, F. S. (2024). Fraud Power Laws. *Journal of Accounting Research*, *62*(3), 833–876.

Cho, S [Soohyun], Vasarhelyi, M. A., Sun, T., & Zhang, C. (2020). Learning from Machine Learning in Accounting and Assurance. *Journal of Emerging Technologies in Accounting*, *17*(1), 1–10.

Choi, J. H [Jung Ho], & Gipper, B. (2024). Fraudulent financial reporting and the consequences for employees. *Journal of Accounting and Economics*, *forthcoming*, 101673.

Choudhary, P., Merkley, K., & Schipper, K. (2021). Immaterial Error Corrections and Financial Reporting Reliability. *Contemporary Accounting Research*, *38*(4), 2423–2460.

Clark, T. E. (2004). Can out-of-sample forecast comparisons help prevent overfitting? *Journal of Forecasting*, *23*(2), 115–139.

Commerford, B. P., Dennis, S. A., Joe, J. R., & Ulla, J. W. (2022). Man Versus Machine: Complex Estimates and Auditor Reliance on Artificial Intelligence. *Journal of Accounting Research*, *60*(1), 171–201.

Craja, P., Kim, A., & Lessmann, S. (2020). Deep learning for detecting financial statement fraud. *Decision Support Systems*, *139*, 113421.

Cressey, D. R. (1953). Other People's Money: *A Study in the Social Pschology of Embezzlement.*

Daroca, F. P., & Holder, W. W. (1985). The Use of Analytical Procedures in Review and Audit Engagements. *Auditing: A Journal of Practice & Theory*, *4*(2), 80–92.

Dechow, P. M., Ge, W., Larson, C. R., & Sloan, R. G. (2011). Predicting Material Accounting Misstatements. *Contemporary Accounting Research*, *28*(1), 17–82.

Dechow, P. M., Sloan, R. G., & Sweeney, A. P. (1995). Detecting Earnings Management. *The Accounting Review*, *70*(2), 193–225.

Dickey, G., Blanke, S., & Seaton, L. (2019). Machine Learning in Auditing. *CPA Journal*, *89*(6), 16–21.

Dietterich, T. (1995). Overfitting and undercomputing in machine learning. *ACM Computing Surveys*, *27*(3), 326–327.

Dikmen, B., & Kücükkocaoglu, G. (2010). The detection of earnings manipulation: the three-phase cutting plane algorithm using mathematical programming. *Journal of Forecasting*, *29*, 442-466.

Dikmen, M., & Burns, C. (2022). The effects of domain knowledge on trust in explainable AI and task performance: A case of peer-to-peer lending. *International Journal of Human-Computer Studies*, *162*, 102792.

Ding, K., Peng, X., & Wang, Y. (2019). A Machine Learning-Based Peer Selection Method with Financial Ratios. *Accounting Horizons*, *33*(3), 75–87.

Dopuch, N., Holthausen, R. W., & Leftwich, R. W. (1987). Predicting Audit Qualifications with Financial and Market Variables. *The Accounting Review*, *62*(3), 431–454.

Dorminey, J., Fleming, A. S., Kranacher, M.-J., & Riley, R. A. (2012). The Evolution of Fraud Theory. *Issues in Accounting Education*, *27*(2), 555–579.

Dorminey, J. W., Fleming, A. S., Kranacher, M.-J., & Riley, R. A. (2010). Beyond the Fraud Triangle. *CPA Journal*, *80*(12), 16–23.

Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *ArXiv Preprint ArXiv:1702.08608*.

Drnevich, P. L., Mahoney, J. T., & Schendel, D. (2020). Has Strategic Management Research Lost Its Way? *Strategic Management Review*, *1*(1), 35–73.

Du, H., & Nehmer, R. A. (2024). Journal of Emerging Technologies in Accounting (JETA): Twenty Years of Growth and Innovation. *Journal of Emerging Technologies in Accounting*, *21*(1), 1–7.

Duan, B., Hu, D., & Lu, H. (2024). Video-Based Deception Detection and Financial Fraud. *SSRN Electronic Journal*, 4964419.

Dutta, I., Dutta, S., & Raahemi, B. (2017). Detecting financial restatements using data mining techniques. *Expert Systems with Applications*, *90*, 374–393.

Dyck, A., Morse, A., & Zingales, L. (2010). Who Blows the Whistle on Corporate Fraud? *The Journal of Finance*, *65*(6), 2213–2253.

Eckstein, C. (2004). The measurement and recognition of intangible assets: then and now. *Accounting Forum*, *28*(2), 139–158.

Elshandidy, T., Eldaly, M. K., & Abdel-Kader, M. (2021). Independent oversight of the auditing profession: A review of the literature. *International Journal of Auditing*, *25*(2), 373–407.

Epstein, B. J., & Ramamoorti, S. (2016). Today's Fraud Risk Models Lack Personality: Auditing with 'Dark Triad' Individuals in the Executive Ranks. *CPA Journal*, *86*(3), 14–21.

European Securities and Markets Authority. (2020). Guidelines on enforcement of financial information, accessed 16.08.2022: https://www.esma.europa.eu/sites/default/files/library/esma32-50-218_guidelines_on_enforcement_of_financial_information_de.pdf.

Ewert, R., & Wagenhofer, A. (2019). Effects of Increasing Enforcement on Financial Reporting Quality and Audit Quality. *Journal of Accounting Research*, *57*(1), 121–168.

Fahrenwaldt, M., & Nohl, S. (2022). Maschinelles Lernen in Risikomodellen: Wie soll Maschinelles Lernen in Risikomodellen reguliert werden? BaFin und Deutsche Bundesbank haben die Unternehmen dazu befragt. Die Ergebnisse der Konsultation liegen nun vor. Der Austausch geht weiter. *BaFin Journal*, (Februar), 14–16.

Fang, V. W., Huang, A. H., & Karpoff, J. M. (2016). Short Selling and Earnings Management: A Controlled Experiment. *The Journal of Finance*, *71*(3), 1251–1294.

Fanning, K. M., & Cogger, K. O. (1998). Neural network detection of management fraud using published financial data. *International Journal of Intelligent Systems in Accounting, Finance and Management*, *7*(1), 21–41.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874.

Federal Financial Supervisory Authority. (2022). Maschinelles Lernen in Risikomodellen – Charakteristika und aufsichtliche Schwerpunkte: Antworten auf das Konsultationspapier, accessed 12.08.2022: https://www.bafin.de/SharedDocs/Downloads/DE/Konsultation/2021/dl_kon_11_21_Ergebnisse_machinelles_Lernen_Risikomodelle.pdf?__blob=publicationFile&v=1.

Feliciano, C., & Quick, R. (2022). Innovative Information Technology in Auditing: Auditors' Perceptions of Future Importance and Current Auditor Expertise. *Accounting in Europe*, *19*(2), 311–331.

Feroz, E. H., Kwon, T. M., Pastena, V. S., & Park, K. (2000). The efficacy of red flags in predicting the SEC's targets: an artificial neural networks approach. *International Journal of Intelligent Systems in Accounting, Finance and Management*, *9*(3), 145–157.

Feroz, E. H., Park, K., & Pastena, V. S. (1991). The Financial and Market Effects of the SEC's Accounting and Auditing Enforcement Releases. *Journal of Accounting Research*, *29*(Supplement), 107–142.

Ferri, C., Hernández-Orallo, J., & Modroiu, E.-R. (2009). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, *30*(1), 27–38.

Financial Reporting Enforcement Panel. (2018). Tätigkeitsbericht 2017.

Fotoh, L. E., & Lorentzon, J. I. (2021). The Impact of Digitalization on Future Audits. *Journal of Emerging Technologies in Accounting*, *18*(2), 77–97.

Fotoh, L. E., & Lorentzon, J. I. (2023). Audit digitalization and its consequences on the audit expectation gap: A critical perspective. *Accounting Horizons*, 37(1), 43–69.

Freund, Y., & Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. *Icml*, *96*, 148–156.

Fukas, P., Rebstadt, J., Menzel, L., & Thomas, O. (2022). Towards Explainable Artificial Intelligence in Financial Fraud Detection: Using Shapley Additive Explanations to Explore Feature Importance. In X. Franch, G. Poels, F. Gailly, & M. Snoeck (Eds.), *Advanced Information Systems EngineeringVol.* 13295, pp. 109–126, Springer International Publishing.

Fülbier, R. U., & Sellhorn, T. (2023). Understanding and improving the language of business: How accounting and corporate reporting research can better serve business and society. *Journal of Business Economics*, *93*(6), 1089–1124.

Gaganis, C. (2009). Classification techniques for the identification of falsified financial statements: a comparative analysis. *Intelligent Systems in Accounting, Finance and Management*, *16*(3), 207–229.

Gaganis, C., Pasiouras, F., & Doumpos, M [Michael] (2007). Probabilistic neural networks for the identification of qualified audit opinions. *Expert Systems with Applications*, *32*(1), 114–124.

Geerts, G. L. (2011). A design science research methodology and its application to accounting information systems research. *International Journal of Accounting Information Systems*, *12*(2), 142–151.

Gepp, A., Kumar, K., & Bhattacharya, S. (2021). Lifting the numbers game: identifying key input variables and a best-performing model to detect financial statement fraud. *Accounting & Finance*, *61*(3), 4601–4638.

Gepp, A., Kumar, K., & Bhattacharya, S. (2024). Taking the hunch out of the crunch: A framework to improve variable selection in models to detect financial statement fraud. *Accounting & Finance*, *64*(2), 1569–1588.

Gepp, A., Linnenluecke, M. K., O'Neill, T. J., & Smith, T. (2018). Big data techniques in auditing research and practice: Current trends and future opportunities. *Journal of Accounting Literature*, *40*, 102–115.

Gerety, M., & Lehn, K. (1997). The causes and consequences of accounting fraud. *Managerial and Decision Economics*, *18*(7-8), 587–599.

Ghosh, P., Ocampo, J., Harris, L., Simpson, E., Krueger, J., & Vaidhyanathan, J. (1998, May 5). Enron Corporation, accessed 19.07.2023: https://de.scribd.com/doc/66581069/Cornell-Research-Report-on-Enron-1998.

Gipper, B., Leuz, C., & Maffett, M. (2020). Public Oversight and Reporting Credibility: Evidence from the PCAOB Audit Inspection Regime. *The Review of Financial Studies*, *33*(10), 4532–4579.

Giroux, G. A. (2006). Earnings magic and the unbalance sheet: *The search for financial reality*, Wiley.

Glancy, F. H., & Yadav, S. B. (2011). A computational model for financial reporting fraud detection. *Decision Support Systems*, *50*(3), 595–601.

Glikson, E., & Woolley, A. W. (2020). Human Trust in Artificial Intelligence: Review of Empirical Research. *Academy of Management Annals*, *14*(2), 627–660.

Goel, S., Gangolly, J., Faerman, S. R., & Uzuner, O. (2010). Can linguistic predictors detect fraudulent financial filings? *Journal of Emerging Technologies in Accounting*, *7*(1), 25–46.

Goel, S., & Uzuner, O. (2016). Do Sentiments Matter in Fraud Detection? Estimating Semantic Orientation of Annual Reports. *Intelligent Systems in Accounting, Finance and Management*, *23*(3), 215–239.

Goelzer, D. L. (2020). Audit Oversight and Effectiveness. *CPA Journal*, *90/91*(12/1), 50–55.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. *Adaptive computation and machine learning*, The MIT Press.

Green, B. P., & Choi, J. H [Jae Hwa] (1997). Assessing the Risk of Management Fraud Through Neural Network Technology. *Auditing: A Journal of Practice & Theory*, *16*(1), 14–28.

Gregor, S., & Hevner, A. R. (2013). Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*, *37*(2), 337–355.

Griffiths, I. (1986). Creative Accounting, Waterstone & Co Limited.

Gu, H., Schreyer, M., Moffitt, K., & Vasarhelyi, M. (2024). Artificial intelligence co-piloted auditing. *International Journal of Accounting Information Systems*, *54*, 100698.

Gu, Y., Huang, Q., & Vasarhelyi, M. A. (2024). It's Not Intelligence; It's Functionality! *Journal of Emerging Technologies in Accounting*, *21*(2), 9–18.

Guidotti, R. (2021). Evaluating local explanation methods on ground truth. *Artificial Intelligence*, *291*, 103428.

Gunning, D. (2017). Explainable Artificial Intelligence (XAI). *Defense Advanced Research Projects Agency* (DARPA), accessed 21.08.2022: https://nsarchive.gwu.edu/sites/default/files/documents/5794867/National-Security-Archive-David-Gunning-DARPA.pdf.

Gwinner, F., Tomitza, C., & Winkelmann, A. (2024). Comparing expert systems and their explainability through similarity. *Decision Support Systems*, *182*, 114248.

Hajek, P. (2019). Interpretable Fuzzy Rule-Based Systems for Detecting Financial Statement Fraud. In J. MacIntyre, I. Maglogiannis, L. Iliadis, & E. Pimenidis (Eds.), *Artificial Intelligence Applications and Innovations*, pp. 425–436, Springer International Publishing.

Hajek, P., & Henriques, R. (2017). Mining corporate annual reports for intelligent detection of financial statement fraud - A comparative study of machine learning methods. *Knowledge-Based Systems*, *128*, 139–152.

Hanenberg, L., & Kostjutschenkow, S. (2021). Die neue Bilanzkontrolle. *BaFin Journal*, (Dezember), 14–17.

Hartmann, M., & Weißenberger, B. E. (2024). Information overload research in accounting: a systematic review of the literature. *Management Review Quarterly*, *74*(3), 1619–1667.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2017). The Elements of Statistical Learning: *Data Mining, Inference, annd Prediction* (2nd ed.), Springer US.

Healy, P. M. (1985). The Effect of Bonus Schemes on Accounting Decisions. *Journal of Accounting and Economics*, *7*(1-3), 85–107.

Hennes, K. M., Leone, A. J., & Miller, B. P. (2008). The Importance of Distinguishing Errors from Irregularities in Restatement Research: The Case of Restatements and CEO/CFO Turnover. *The Accounting Review*, *83*(6), 1487–1519.

Hermanson, D. R., & Wolfe, D. T. (2024). The Fraud Diamond: A 20-Year Retrospective. *CPA Journal*, *94*(3/4), 16–21.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105.

Hevner, A. R., Parsons, J., Brendel, A. B., Lukyanenko, R., Tiefenbeck, V., Tremblay, M. C., & vom Brocke, J. (2024). Transparency in design science research. *Decision Support Systems*, *182*, 114236.

Hey, T., Tansley, S., & Tolle, K. (2009). The Fourth Paradigm: *Data-Intensive Scientific Discovery*, Microsoft Research.

Hobson, J. L., Mayew, W. J., & Venkatachalam, M. (2012). Analyzing Speech to Detect Financial Misreporting. *Journal of Accounting Research*, *50*(2), 349–392.

Höglund, H. (2012). Detecting earnings management with neural networks. *Expert Systems with Applications*, *39*(10), 9564–9570.

Höglund, H. (2013). Fuzzy linear regression-based detection of earnings management. *Expert Systems with Applications*, *40*(15), 6166–6172.

Hollinger, R. C., & Clark, J. P. (1983). Theft by Employees, Lexington Books.

Hoogs, B. K., Kiehl, T. R., LaComb, C. A., & Senturk, D. (2007). A genetic algorithm approach to detecting temporal patterns indicative of financial statement fraud. *Intelligent Systems in Accounting, Finance and Management*, *15*(1–2), 41–56.

Huang, A. H., Kraft, P., & Wang, S. (2023). The Usefulness of Credit Ratings for Accounting Fraud Prediction. *The Accounting Review*, *98*(7), 347–376.

Huang, L., Abrahams, A., & Ractham, P. (2022). Enhanced financial fraud detection using cost-sensitive cascade forest with missing value imputation. *Intelligent Systems in Accounting, Finance and Management*, *29*(3), 133–155.

Huang, S.-M., Wang, T., Yen, J.-C., Lee, C.-B., Wang, Y.-C., & Yang, Y.-T. (2022). The Use of Geographic Information in Audit Data Analytics for Evidence Gathering: A Design Science Approach. *Journal of Information Systems*, *36*(3), 115–128.

Huang, S.-Y., Tsaih, R.-H., & Yu, F. (2014). Topological pattern discovery and feature extraction for fraudulent financial reporting. *Expert Systems with Applications*, *41*(9), 4360–4372.

Humpherys, S. L., Moffitt, K. C., Burns, M. B., Burgoon, J. K., & Felix, W. F. (2011). Identification of fraudulent financial statements using linguistic credibility analysis. *Decision Support Systems*, *50*(3), 585–594.

International Auditing and Assurance Standards Board. (2020). Fraud and Going Concern in an Audit of Financial Statements: Exploring the Differences Between Public Perceptions About the Role of the Auditor and the Auditor's Responsibilities in a Financial Statement Audit, accessed 14.07.2024: https://www.iaasb.org/publications/fraud-and-going-concern-audit-financial-statements.

International Auditing and Assurance Standards Board. (2024). Proposed International Standard on Auditing 240 (Revised): The Auditor's Responsibilities Relating to Fraud in an Audit of Financial Statements and Proposed Conforming and Consequential Amendments to Other ISAs, accessed 30.06.2024: https://ifacweb.blob.core.windows.net/publicfiles/2024-02/IAASB-Exposure-Draft-Proposed-ISA-240-Revised-Fraud.pdf.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning, Springer US.

Jan, C. (2018). An Effective Financial Statements Fraud Detection Model for the Sustainable Development of Financial Markets: Evidence from Taiwan. *Sustainability*, *10*(2), 513.

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685–695.

Johnstone, K. M. (2000). Client-Acceptance Decisions: Simultaneous Effects of Client Business Risk, Audit Risk, Auditor Business Risk, and Risk Adaptation. *Auditing: A Journal of Practice & Theory*, *19*(1), 1–25.

Johnstone, K. M., & Bedard, J. C. (2003). Risk Management in Client Acceptance Decisions. *The Accounting Review*, *78*(4), 1003–1025.

Johnstone, K. M., & Bedard, J. C. (2004). Audit Firm Portfolio Management Decisions. *Journal of Accounting Research*, *42*(4), 659–690.

Jones, J. J. (1991). Earnings Management During Import Relief Investigations. *Journal of Accounting Research*, *29*(2), 193–228.

Jones, K. L., Krishnan, G. V., & Melendrez, K. D. (2008). Do Models of Discretionary Accruals Detect Actual Cases of Fraudulent and Restated Earnings? An Empirical Analysis. *Contemporary Accounting Research*, *25*(2), 499–531.

Kaminski, K. A., Sterling Wetzel, T., & Guan, L. (2004). Can financial ratios detect fraudulent financial reporting? *Managerial Auditing Journal*, *19*(1), 15–28.

Karpoff, J. M., Koester, A., Lee, D. S., & Martin, G. S. (2017). Proxies and Databases in Financial Misconduct Research. *The Accounting Review*, *92*(6), 129–163.

Karpoff, J. M., Lee, D. S., & Martin, G. S. (2008). The Cost to Firms of Cooking the Books. *Journal of Financial and Quantitative Analysis*, *43*(3), 581–611.

Karpoff, J. M., & Lou, X. (2010). Short Sellers and Financial Misconduct. *The Journal of Finance*, *65*(5), 1879–1913.

Ke, B. (2024). Accounting research for the digital age. *The British Accounting Review*, *forthcoming*, 101443.

Kelton, A. S., & Murthy, U. S. (2023). Reimagining design science and behavioral science AIS research through a business activity lens. *International Journal of Accounting Information Systems*, *50*, 100623.

Kim, B., Khanna, R., & Koyejo, O. O. (2016). Examples are not enough, learn to criticize! criticism for interpretability. *Advances in Neural Information Processing Systems*, *29*.

Kim, Y. J., Baik, B., & Cho, S [Sungzoon] (2016). Detecting financial misstatements with fraud intention using multi-class cost-sensitive learning. *Expert Systems with Applications*, *62*, 32–43.

Kirkos, E., Spathis, C., & Manolopoulos, Y. (2007). Data Mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, *32*(4), 995–1003.

Kirkos, E., Spathis, C., Nanopoulos, A., & Manolopoulos, Y. (2007). Identifying Qualified Auditors' Opinions: A Data Mining Approach. *Journal of Emerging Technologies in Accounting*, *4*(1), 183–197.

Knudsen, D.-R. (2020). Elusive boundaries, power relations, and knowledge production: A systematic review of the literature on digitalization in accounting. *International Journal of Accounting Information Systems*, *36*, 100441.

Kogan, A., Mayhew, B. W., & Vasarhelyi, M. A. (2019). Audit Data Analytics Research—An Application of Design Science Methodology. *Accounting Horizons*, *33*(3), 69–73.

Koh, H. C., & Woo, E.-S. (1998). The expectation gap in auditing. *Managerial Auditing Journal*, *13*(3), 147–154.

Krieger, F., Drews, P., & Velte, P. (2021). Explaining the (non-) adoption of advanced data analytics in auditing: A process theory. *International Journal of Accounting Information Systems*, *41*, 100511.

Krishnan, G. V., Sun, L., Wang, Q., & Yang, R. (2013). Client Risk Management: A Pecking Order Analysis of Auditor Response to Upward Earnings Management Risk. *Auditing: A Journal of Practice & Theory*, *32*(2), 147–169.

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, *28*(5), 1–26.

Kuhn, M., & Johnson, K. (2013). Applied Predictive Modeling, Springer New York.

Kumar, S., Marrone, M., Liu, Q., & Pandey, N. (2020). Twenty years of the International Journal of Accounting Information Systems: A bibliometric analysis. *International Journal of Accounting Information Systems*, *39*, 100488.

Kumarakulasinghe, N. B., Blomberg, T., Liu, J., Saraiva Leao, A., & Papapetrou, P. (2020). Evaluating Local Interpretable Model-Agnostic Explanations on Clinical Machine Learning Classification Models. In *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 7–12, IEEE.

Kureljusic, M., & Karger, E. (2024). Forecasting in financial accounting with artificial intelligence – A systematic literature review and future research agenda. *Journal of Applied Accounting Research*, *25*(1), 81–104.

Langenbucher, K., Leuz, C., Krahnen, J. P., & Pelizzon, L. (2020). What are the wider supervisory implications of the Wirecard case? *SAFE White Paper, No. 74*, Leibniz Institute for Financial Research SAFE.

Larcker, D. F., & Zakolyukina, A. A. (2012). Detecting Deceptive Discussions in Conference Calls. *Journal of Accounting Research*, *50*(2), 495–540.

Lazer, D., Kennedy, R., King, G., & Vespignani, A. (2014). The parable of Google Flu: traps in big data analysis. *Science*, *343*(6176), 1203–1205.

Lee, T. A., Ingram, R. W., & Howard, T. P. (1999). The Difference between Earnings and Operating Cash Flow as an Indicator of Financial Reporting Fraud *Contemporary Accounting Research*, *16*(4), 749–786.

Leonhardt, R. (2015). Klassische Literatur als Inspiration für Manager, Gabler Verlag.

Levitt, A. (1998). The 'Numbers Game'. *CPA Journal*, *68*(12), 14–19.

Li, N., Kim, M., Dai, J., & Vasarhelyi, M. A. (2024). Using Artificial Intelligence in ESG Assurance. *Journal of Emerging Technologies in Accounting*, *21*(2), 83–99.

Lin, C.-C., Chiu, A.-A., Huang, S. Y., & Yen, D. C. (2015). Detecting the financial statement fraud: The analysis of the differences between data mining techniques and experts' judgments. *Knowledge-Based Systems*, *89*, 459–470.

Lin, J. W., Hwang, M. I., & Becker, J. D. (2003). A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, *18*(8), 657–665.

Lin, K., & Gao, Y. (2022). Model interpretability of financial fraud detection by group SHAP. *Expert Systems with Applications*, *210*, 118354.

Ling, C. X., & Sheng, V. S. (2008). Cost-Sensitive Learning and the Class Imbalance Problem. *Encyclopedia of Machine Learning*, 231–235.

Liou, F.-M. (2008). Fraudulent financial reporting detection and business failure prediction models: a comparison. *Managerial Auditing Journal*, *23*(7), 650–662.

Liu, C., Low, A., Masulis, R. W., & Le Zhang (2020). Monitoring the Monitor: Distracted Institutional Investors and Board Governance. *The Review of Financial Studies*, *33*(10), 4489–4531.

Liu, M. (2022). Assessing Human Information Processing in Lending Decisions: A Machine Learning Approach. *Journal of Accounting Research*, *60*(2), 607–651.

Loebbecke, J. K., Eining, M. M., & Willingham, J. J. (1989). Auditors' Experience with Material Irregularities: Frequency, Nature, and Detectability. *Auditing: A Journal of Practice & Theory*, *9*(1), 1–28.

Löhlein, L. (2016). From peer review to PCAOB inspections: Regulating for audit quality in the U.S. *Journal of Accounting Literature*, *36*(1), 28–47.

Lokanan, M., & Sharma, S. (2024). The use of machine learning algorithms to predict financial statement fraud. *The British Accounting Review*, *56*, 101441.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251–266.

Marten, K.-U., Föhr, T. L., & McIntosh, S. (2022). KI-basierte Datenanalysen und risikoorientierter Prüfungsansatz. *Die Wirtschaftsprüfung*, *75*(16), 898–907.

Marten, K.-U., & Harder, R. (2019). Digitalisierung in der Abschlussprüfung. *Die Wirtschaftsprüfung*, *72*(14), 761–769.

Martens, D., Hinns, J., Dams, C., Vergouwen, M., & Evgeniou, T. (2025). Tell me a story! Narrative-driven XAI with Large Language Models. *Decision Support Systems*, *191*, 114402.

Massa, M., Zhang, B., & Zhang, H. (2015). The Invisible Hand of Short Selling: Does Short Selling Discipline Earnings Management? *The Review of Financial Studies*, *28*(6), 1701–1736.

McCarthy, J., Minsky, M., Rochester, N., & Shannon, C. (1955, August 31). A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, accessed 27.07.2023: https://web.archive.org/web/20080930164306/http://www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.html.

McCrum, D. (2020). Wirecard: the timeline, accessed 30.08.2022: https://www.ft.com/content/284fb1ad-ddc0-45df-a075-0709b36868db.

McKee, T. E. (2009). A Meta-Learning Approach to Predicting Financial Statement Fraud. *Journal of Emerging Technologies in Accounting*, *6*(1), 5–26.

Meisenbacher, S., Turowski, M., Phipps, K., Rätz, M., Müller, D., Hagenmeyer, V., & Mikut, R. (2022). Review of automated time series forecasting pipelines. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *12*(6), Article e1475.

Mercado, J. E., Rupp, M. A., Chen, J. Y. C., Barnes, M. J., Barber, D., & Procci, K. (2016). Intelligent Agent Transparency in Human-Agent Teaming for Multi-UxV Management. *Human Factors*, *58*(3), 401–415.

Mercer, M. (2004). How Do Investors Assess the Credibility of Management Disclosures? *Accounting Horizons*, *18*(3), 185–196.

Messalas, A., Kanellopoulos, Y., & Makris, C. (2019). Model-Agnostic Interpretability with Shapley Values. In *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38.

Molnar, C. (2022). Interpretable Machine Learning (2nd ed.).

Molnar, C., Casalicchio, G., & Bischl, B. (2018). iml: An R package for Interpretable Machine Learning. *Journal of Open Source Software*, *3*(27), 786.

Morales, J., Gendron, Y., & Guénin-Paracini, H. (2014). The construction of the risky individual and vigilant organization: A genealogy of the fraud triangle. *Accounting, Organizations and Society*, *39*(3), 170–194.

Morales-Díaz, J., & Zamora-Ramírez, C. (2018). The Impact of IFRS 16 on Key Financial Ratios: A New Methodological Approach. *Accounting in Europe*, *15*(1), 105–133.

Morris, G. D. (2009). How a group of business students sold Enron a year before the collapse. *Financial History*, *94*(Spring/Summer), 12–15.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, *116*(44), 22071–22080.

Murphy, K. P. (2012). Machine learning: a probabilistic perspective, MIT press.

Ogut, H., Aktas, R., Alp, A., & Doganay, M. M. (2009). Prediction of financial information manipulation by using support vector machine and probabilistic neural network. *Expert Systems with Applications*, *36*(3), 5419–5423.

Pai, P.-F., Hsu, M.-F., & Wang, M.-C. (2011). A support vector machine-based model for detecting top management fraud. *Knowledge-Based Systems*, *24*(2), 314–321.

Papík, M., & Papíková, L. (2020). Detection models for unintentional financial restatements. *Journal of Business Economics and Management*, *21*(1), 64–86.

Papík, M., & Papíková, L. (2022). Detecting accounting fraud in companies reporting under US GAAP through data mining. *International Journal of Accounting Information Systems*, *45*, 100559.

Paulhus, D. L., & Williams, K. M. (2002). The Dark Triad of personality: Narcissism, Machiavellianism, and psychopathy. *Journal of Research in Personality*, *36*, 556–563.

Peffers, K., Rothenberger, M. A., Tuunanen, T., & Vaezi, R. (2012). Design Science Research Evaluation. In K. Peffers, M. A. Rothenberger, & B. Kuechler (Eds.), *Design Science Research in Information Systems. Advances in Theory and Practice*, pp. 398–410, Springer.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77.

Perols, J. (2011). Financial statement fraud detection: An analysis of statistical and machine learning algorithms. *Auditing: A Journal of Practice & Theory*, *30*(2), 19–50.

Perols, J. L., Bowen, R. M., Zimmermann, C., & Samba, B. (2017). Finding Needles in a Haystack: Using Data Analytics to Improve Fraud Prediction. *The Accounting Review*, *92*(2), 221–245.

Persons, O. S. (1995). Using Financial Statement Data To Identify Factors Associated With Fraudulent Financial Reporting. *Journal of Applied Business Research*, *11*(3), 38–46.

Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. *International Journal of Machine Learning Technology*, *2*(1), 37–63.

PriceWaterhouseCoopers. (2022). PwC's Global Economic Crime and Fraud Survey 2022, accessed 19.07.2023: https://www.pwc.com/gx/en/forensics/gecsm-2022/pdf/PwC%E2%80%99s-Global-Economic-Crime-and-Fraud-Survey-2022.pdf.

PriceWaterhouseCoopers. (2024). PwC's Global Economic Crime Survey 2024, accessed 14.07.2024: https://www.pwc.com/gx/en/services/forensics/economic-crime-survey.html.

Purda, L., & Skillicorn, D. (2015). Accounting Variables, Deception, and a Bag of Words: Assessing the Tools of Fraud Detection. *Contemporary Accounting Research*, *32*(3), 1193–1223.

Ragothaman, S., Carpenter, J., & Buttars, T. (1995). Using Rule Induction for Knowledge Acquisition: An Expert Systems Approach to Evaluating Material Errors and Irregularities. *Expert Systems with Applications*, *9*(4), 483–490.

Ragothaman, S., & Lavin, A. (2008). Restatements Due to Improper Revenue Recognition: A Neural Networks Perspective. *Journal of Emerging Technologies in Accounting*, *5*(1), 129–142.

Rapp, D. J., & Pampel, J. (2021). Zur Akzeptanz künstlicher Intelligenz in der Abschlussprüfung. *Die Wirtschaftsprüfung*, *74*(11), 678–689.

Ravisankar, P., Ravi, V., Raghava Rao, G., & Bose, I. (2011). Detection of financial statement fraud and feature selection using data mining techniques. *Decision Support Systems*, *50*(2), 491–500.

Rezaee, Z., & Riley, R. A. (2009). Financial Statement Fraud: *Prevention and Detection* (2nd ed.), John Wiley & Sons.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016a). "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In B. Krishnapuram, M. Shah, A. Smola, C. Aggarwal, D. Shen, & R. Rastogi (Eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, ACM.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016b). Model-Agnostic Interpretability of Machine Learning. In *ICML Workshop on Human Interpretability in Machine Learning (WHI 2016),* New York.

Rudin, C. (2019). Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence*, *1*(5), 206–215.

Ruhnke, K., & Schmidt, M. (2014). The audit expectation gap: existence, causes, and the impact of changes. *Accounting and Business Research*, *44*(5), 572–601.

Russell, S., & Norvig, P. (2021). Artificial Intelligence, Global Edition: A Modern Approach, Pearson.

Samek, W., & Müller, K.-R. (2019). Towards Explainable Artificial Intelligence. In W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, K.-R. Müller, R. Goebel, Y. Tanaka, W. Wahlster, & J. Siekmann (Eds.), *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (pp. 5–22), Springer.

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*, 210–229.

Schneider, M., & Brühl, R. (2023). Disentangling the black box around CEO and financial information-based accounting fraud detection: machine learning-based evidence from publicly listed U.S. firms. *Journal of Business Economics*, *93*(9), 1591–1628.

Seidenstein, T., Marten, K.-U., Donaldson, G., Föhr, T. L., Reichelt, V., & Jakoby, L. B. (2024). Innovation in Audit and Assurance: A Global Study of Disruptive Technologies. *Journal of Emerging Technologies in Accounting*, *21*(1), 129–146.

Seiffert, C., Khoshgoftaar, T. M., van Hulse, J., & Napolitano, A. (2010). RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part a: Systems and Humans*, *40*(1), 185–197.

Sellhorn, T. (2020). Machine Learning und empirische Rechnungslegungsforschung: Einige Erkenntnisse und offene Fragen. *Schmalenbachs Zeitschrift Für Betriebswirtschaftliche Forschung*, *72*(1), 49–69.

Shapley, L. S. (1953). A Value for n-Person Games. In *Contributions to the Theory of Games (AM-28), Volume II* (pp. 307–318), 17. Princeton University Press.

Shimshack, J. P., & Ward, M. B. (2005). Regulator reputation, enforcement, and environmental compliance. *Journal of Environmental Economics and Management*, *50*(3), 519–540.

Shin, D. (2021). The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable AI. *International Journal of Human-Computer Studies*, *146*, 102551.

Silberzahn, R., & Uhlmann, E. L. (2015). Many hands make tight work. *Nature*, *526*(7572), 189–191.

Simon, H. A. (1996). The Sciences of the Artificial (3rd ed.), MIT press.

Singh, D., & Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, *97*, 105524.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, *45*(4), 427–437.

Song, X.-P., Hu, Z.-H., Du, J.-G., & Sheng, Z.-H. (2014). Application of Machine Learning Methods to Risk Assessment of Financial Statement Fraud: Evidence from China. *Journal of Forecasting*, *33*(8), 611–626.

Spathis, C. T. (2002). Detecting false financial statements using published data: some evidence from Greece. *Managerial Auditing Journal*, *17*(4), 179–191.

Spathis, C. T., Doumpos, M [M.], & Zopounidis, C. (2002). Detecting falsified financial statements: a comparative study using multicriteria analysis and multivariate statistical techniques. *European Accounting Review*, *11*(3), 509–535.

Sujon, K. M., Hassan, R. B., Towshi, Z. T., Othman, M. A., Samad, M. A., & Choi, K. (2024). When to Use Standardization and Normalization: Empirical Evidence From Machine Learning Models and XAI. *IEEE Access*, *12*, 135300–135314.

Summers, S. L., & Sweeney, J. T. (1998). Fraudulently Misstated Financial Statements and Insider Trading: An Empirical Analysis. *The Accounting Review*, *73*(1), 131–146.

Sundararajan, M., & Najmi, A. (2020). The Many Shapley Values for Model Explanation. In *Proceedings of the 37th International Conference on Machine Learning,* Online.

Sutherland, E. H. (1941). Crime and Business. *The Annals of the American Academy of Political and Social Science*, *217*(Sep.), 112–118.

Sutton, S. G., Arnold, V., Collier, P., & Leech, S. A. (2021). Leveraging the synergies between design science and behavioral science research methods. *International Journal of Accounting Information Systems*, *43*, 100536.

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, *16*(4), 437–450.

Tatusch, M., Klassen, G., Bravidor, M., & Conrad, S. (2020). Predicting Erroneous Financial Statements Using a Density-Based Clustering Approach. In *Proceedings of the 4th International Conference on Business and Information Management*, pp. 89–94, ACM, New York, NY, USA. Rome Italy, 03.08.2020 – 05.08.2020.

Thomas, O., Bruckner, A., Leimkühler, M., Remark, F., & Thomas, K. (2021). Konzeption, Implementierung und Einführung von KI-Systemen in der Wirtschaftsprüfung. *Die Wirtschaftsprüfung*, *70*(9), 551–561.

Throckmorton, C. S., Mayew, W. J., Venkatachalam, M., & Collins, L. M. (2015). Financial fraud detection using vocal, linguistic and financial cues. *Decision Support Systems*, *74*, 78–87.

Tsai, C.-F., & Chiou, Y.-J. (2009). Earnings management prediction: A pilot study of combining neural networks and decision trees. *Expert Systems with Applications*, *36*(3), 7183–7191.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, *59*(236), 433–460.

Venable, J., Pries-Heje, J., & Baskerville, R. (2016). FEDS: a Framework for Evaluation in Design Science Research. *European Journal of Information Systems*, *25*(1), 77–89.

Vilone, G., & Longo, L. (2021). Notions of explainability and evaluation approaches for explainable artificial intelligence. *Information Fusion*, *76*, 89–106.

Vitali, S., & Giuliani, M. (2024). Emerging digital technologies and auditing firms: Opportunities and challenges. *International Journal of Accounting Information Systems*, *53*, 100676.

Vladu, A. B., Amat, O., & Cuzdriorean, D. D. (2017). Truthfulness in Accounting: How to Discriminate Accounting Manipulators from Non-manipulators. *Journal of Business Ethics*, *140*(4), 633–648.

vom Brocke, J., Hevner, A., & Maedche, A. (2020). Introduction to Design Science Research. In J. vom Brocke, A. Hevner, & A. Maedche (Eds.), *Design Science Research. Cases* (pp. 1–16), Springer Nature Switzerland.

Weißenberger, B. E., Förster, G., Bravidor, M., & Wesser, M. B. (2019). Wohin führt die Digitalisierung? *Die Wirtschaftsprüfung*, *72*(20), 1118–1124.

Wells, J. T. (1997). Occupational fraud and abuse, Obsidian Publishing.

Winter, R. (2008). Design science research in Europe. *European Journal of Information Systems*, *17*(5), 470–475.

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a Standard Process Model for Data Mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 29–39.

Wolfe, D. T., & Hermanson, D. R. (2004). The fraud diamond: Considering the four elements of fraud. *CPA Journal*, *74*(12), 38–42.

Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, *415*, 295–316.

Zahn, M. von, Feuerriegel, S., & Kuehl, N. (2022). The Cost of Fairness in AI: Evidence from E-Commerce. *Business & Information Systems Engineering*, *64*(3), 335–348.

Zhang, C., Cho, S [Soohyun], & Vasarhelyi, M. (2022). Explainable Artificial Intelligence (XAI) in auditing. *International Journal of Accounting Information Systems*, *46*, 100572.

Zhang, G., Atasoy, H., & Vasarhelyi, M. A. (2022). Continuous monitoring with machine learning and interactive data visualization: An application to a healthcare payroll process. *International Journal of Accounting Information Systems*, *46*, 100570.

Zhang, Y [Yi], Hu, A., Wang, J., & Zhang, Y [Yaojie] (2022). Detection of fraud statement based on word vector: Evidence from financial companies in China. *Finance Research Letters*, *46*, 102477.

Legal Sources

APAReG (German Auditor Oversight Reform Act). Gesetz zur Umsetzung der aufsichts- und berufsrechtlichen Regelungen der Richtlinie 2014/56/EU sowie zur Ausführung der entsprechenden Vorgaben der Verordnung (EU) Nr. 537/2014 im Hinblick auf die Abschlussprüfung bei Unternehmen von öffentlichem Interesse (Abschlussprüferaufsichtsreformgesetz - APAReG) vom 31.03.2016, BGBl. 2016 I Nr. 14.

Directive 2004/109/EC of the European Parliament and of the Council of 15 December 2004 on the harmonisation of transparency requirements in relation to information about issuers whose securities are admitted to trading on a regulated market and amending Directive 2001/34/EC, Official Journal of the European L 390, 31.12.2004, pp. 38–57.

Directive 2014/56/EU of the European Parliament and the Council of 16 April 2014 amending Directive 2006/43/EC on statutory audits of annual accounts and consolidated accounts, Official Journal of the European L 158, 27.05.2014, pp. 196–226.

International Auditing and Assurance Standards Board. ISA 200 (2009). International Standard on Auditing 200 "Overall Objectives of the Independent Auditor and the Conduct of an Audit in Accordance with International Standards on Auditing".

International Auditing and Assurance Standards Board. ISA 230 (2009). International Standard on Auditing 230 "Audit Documentation".

International Auditing and Assurance Standards Board. ISA 240 (2009). International Standard on Auditing 240 "Fraud in an Audit of Financial Statements".

International Auditing and Assurance Standards Board. ISA 300 (2009). International Standard on Auditing 300 "Planning an Audit of Financial Statements".

International Auditing and Assurance Standards Board. ISA 315 (Rev.) (2013). International Standard on Auditing 315 (Revised) "Identifying and Assessing the Risks of Material Misstatement through Understanding the Entity and Its Environment".

International Auditing and Assurance Standards Board. ISA 520 (2009). International Standard on Auditing 520 "Analytical Procedures".

International Auditing and Assurance Standards Board. ISQC 1 (2009). International Standards on Quality Control 1 "Quality Control for Firms that Perform Audits and Reviews of Financial Statements, and Other Assurance and Related Services Engagements".

Institut der Wirtschaftsprüfer in Deutschland e. V. IDW PH 9.330.3 (2010). IDW Prüfungshinweis 9.330.3: Einsatz von Datenanalysen im Rahmen der Abschlussprüfung (German Audit Practice Note).

Institut der Wirtschaftsprüfer in Deutschland e. V. IDW QS 1 (2017). IDW Qualitätssicherungsstandard 1 "Anforderungen an die Qualitätssicherung in der Wirtschaftsprüferpraxis" (German Standard on Quality Control).

Regulation (EU) No 1095/2010 of the European Parliament and of the Council of 24 November 2010 establishing a European Supervisory Authority (European Securities and Markets Authority), amending Decision No 716/2009/EC and repealing Commission Decision 2009/77/EC, Official Journal of the European L 331, 15.12.2010, pp. 84–119.

Regulation (EU) No 537/2014 of the European Parliament and the Council of 16 April 2014 on specific requirements regarding statutory audit of public-interest entities and repealing Commission Decision 2005/909/EC, Official Journal of the European L 158, 27.05.2014, pp. 77–112.

SOX. Sarbanes-Oxley Act of 2002, Public Law No. 107–204, 116 Stat. 745 (2002).

WPO (German Public Accountants Act). Wirtschaftsprüferordnung in der Fassung der Bekanntmachung vom 5. November 1975 (BGBl. I S. 2803), das zuletzt durch Artikel 35 des Gesetzes vom 23. Oktober 2024 (BGBl. 2024 I Nr. 323) geändert worden ist, BGBl. 2024 I S. 323.

Appendix

Appendix A: Code – Data Preprocessing

Setup

```
## load packages
library(tidyverse)
library(readxl)
```

Data Import

Read AAER Excel Sheets

```
setwd(filepath_aaer)

## set modified function to read excel file
read_excel_allsheets <- function(filename, tibble = TRUE) {
    sheets <- readxl::excel_sheets(filename)
    x <- lapply(sheets, function(X) readxl::read_excel(filename, sheet = X))
    if(!tibble) x <- lapply(x, as.data.frame)
    names(x) <- sheets
    x
}

## import AAER data
df_aaer_all <- read_excel_allsheets("DGLS_20211231_D.xlsx")

## annual sheet and select for variables of interest
df_aaer_ann <- df_aaer_all$ann %>%
  select("YEARA", "GVKEY", "P_AAER", "UNDERSTATEMENT") %>%
  add_column(MISSTATEMENT = 1, .before = "UNDERSTATEMENT") %>%
  filter(!is.na(GVKEY)) %>% # only observations with GVKEY
  rename(FYEAR = YEARA) %>%
  rename_with(~ tolower(gsub(".", "_", .x, fixed = TRUE)))

## detail sheet and select for variables of interest
df_aaer_detail <- df_aaer_all$detail %>%
  filter(ANNUAL == 1) %>% # only with misstated annual data
  filter(!is.na(GVKEY)) %>% # only observations with GVKEY
  select("GVKEY", "REV", "REC", "COGS", "INV", "RESERVE", "DEBT", "MKT_SEC",
         "INC_EXP_SE", "ASSET", "PAY", "LIAB", "REASON", "EXPLANATION") %>%
  rename(C_REV = REV, C_REC = REC, C_COGS = COGS, C_INV = INV,
         C_RESERVE = RESERVE, C_DEBT = DEBT, C_MKT_SEC = MKT_SEC,
         C_INC_EXP_SE = INC_EXP_SE, C_ASSET = ASSET, C_PAY = PAY,
         C_LIAB = LIAB) %>%
  rename_with(~ tolower(gsub(".", "_", .x, fixed = TRUE)))

## remove temporary objects
rm(read_excel_allsheets)
```

AAER | Match detail & ann

```
## match annual ("ann") and detail ("detail") sheets on gvkey
df_aaer <- left_join(df_aaer_ann, df_aaer_detail, by = "gvkey")

df_aaer <- df_aaer %>%
  filter(!is.na(explanation))

## remove temporary objects
rm(df_aaer_ann, df_aaer_detail, df_aaer_all)
```

Import Compustat Data

```
setwd(filepath_Compustat)

## import Compustat data
df_Compustat <- read.csv("[redacted].csv")
```

Preprocess Compustat Data

```
## select variables of interest
df_Compustat <- df_Compustat %>%
  select(fyear, gvkey, sich, act, ap, at, ceq, che, cogs, csho, dlc, dltis,
         dltt, dp, ib, invt, ivao, ivst, lct, lt, ni, ppegt, pstk, re, rect,
         sale, sstk, txp, txt, xint, prcc_f)
```

Match Data

Join AAER data on Compustat data

```
## join aaer and Compustat data on gvkey and fyear
df_data_all <- left_join(df_Compustat, df_aaer, by = c("gvkey", "fyear"))

## remove temporary objects
rm(df_Compustat, df_aaer)
```

Reformat misstatement variables and reorder columns

```
## set misstatement, understatement & c_xxx to 0
df_data_all[c("misstatement", "understatement", "c_rev", "c_rec", "c_cogs",
              "c_inv", "c_reserve", "c_debt", "c_mkt_sec", "c_inc_exp_se",
              "c_asset", "c_pay", "c_liab"
              )][is.na(df_data_all[c("misstatement",
              "understatement", "c_rev", "c_rec", "c_cogs", "c_inv",
              "c_reserve", "c_debt", "c_mkt_sec", "c_inc_exp_se", "c_asset",
              "c_pay", "c_liab")])] <- 0

## reorder columns
col_order <- c("fyear", "gvkey", "sich", "p_aaer", "misstatement",
```

```
                "understatement", "act", "ap", "at", "ceq", "che", "cogs",
                "csho", "dlc", "dltis", "dltt", "dp", "ib", "invt", "ivao",
                "ivst", "lct", "lt", "ni", "ppegt", "pstk", "re", "rect",
                "sale", "sstk", "txp", "txt", "xint", "prcc_f","c_rev",
                "c_rec", "c_cogs", "c_inv", "c_reserve", "c_debt",
                "c_mkt_sec", "c_inc_exp_se", "c_asset", "c_pay", "c_liab",
                "reason", "explanation")

df_data_all <- df_data_all[, col_order]

## remove temporary objects
rm(col_order)
```

Financial Ratios for Benchmark Model

Additional variables for a benchmark model (Dechow et al. 2010) are calculated below as it has been done in previous research. Calculation is adapted based on SAS coding from Bao et al. 2020 referring to Dechow et al. 2011, Cecchini et al. 2010, Beneish 1999 and Summers/Sweeney 1998. - 11 ratios (Dechow et al. 2011) - 3 ratios (Cecchini et al. 2010) - depreciation index (based on Beneish 1999) - retained earnings over total assets (based on Summers/Sweeney 1998) - EBIT (based on Summers/Sweeney 1998)

Compute lagged variables

```
## set lagged variables
df_data_all <- df_data_all %>%
  mutate(
    lag_gvkey = lag(gvkey),
    lag_fyear = lag(fyear),
    lag_at = lag(at)
  )

## reorder columns
col_order2 <- c("fyear", "lag_fyear", "gvkey", "lag_gvkey", "sich", "p_aaer",
                "misstatement", "understatement", "act", "ap", "at",
                "lag_at", "ceq", "che", "cogs", "csho", "dlc", "dltis",
                "dltt", "dp", "ib", "invt", "ivao", "ivst", "lct", "lt",
                "ni", "ppegt", "pstk", "re", "rect", "sale", "sstk", "txp",
                "txt", "xint", "prcc_f","c_rev", "c_rec", "c_cogs", "c_inv",
                "c_reserve", "c_debt", "c_mkt_sec", "c_inc_exp_se",
                "c_asset", "c_pay", "c_liab", "reason", "explanation")

df_data_all <- df_data_all[, col_order2]

## remove temporary objects
rm(col_order2)

## handling missing values of lag_gvkey
df_data_all <- df_data_all %>%
  group_by(gvkey) %>%
  mutate(
    lag_fyear = ifelse(lag(gvkey) != gvkey, NA, lag_fyear),
    lag_at = ifelse(lag(gvkey) != gvkey, NA, lag_at)
```

```
  ) %>%
  ungroup()
```

dch_wc = changes in working capital accruals

```
## working capital
df_data_all <- df_data_all %>%
  mutate(
    txp = ifelse(is.na(txp), 0, txp),
    wc = (act - che) - (lct - dlc - txp),
    lag_wc = lag(wc)
  )

## changes in working capital accruals
df_data_all <- df_data_all %>%
  mutate(
    ch_wc = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                   wc - lag_wc, NA)
  )

## changes in working capital accruals as a percentage of total assets
df_data_all <- df_data_all %>%
  mutate(
    dch_wc = ch_wc * 2 / (at + lag_at)
  )
```

ch_rsst = changes in RSST_accruals

```
## compute nco
df_data_all <- df_data_all %>%
  mutate(
    ivao = ifelse(is.na(ivao), 0, ivao),
    nco = (at - act - ivao) - (lt - lct - dltt),
    lag_nco = lag(nco)
  )

## compute ch_nco
df_data_all <- df_data_all %>%
  mutate(
    ch_nco = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                    nco - lag_nco, NA)
  )

## compute fin and lag_fin
df_data_all <- df_data_all %>%
  mutate(
    ivst = ifelse(is.na(ivst), 0, ivst),
    pstk = ifelse(is.na(pstk), 0, pstk),
    fin = (ivst + ivao) - (dltt + dlc + pstk),
    lag_fin = lag(fin)
  )

## compute ch_fin
```

```
df_data_all <- df_data_all %>%
  mutate(
    ch_fin = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                    fin - lag_fin, NA)
  )

## compute ch_rsst
df_data_all <- df_data_all %>%
  mutate(
    ch_rsst = (ch_wc + ch_nco + ch_fin) * 2 / (at + lag_at)
  )
```

dch_rec = changes in receivables

```
## compute lag_rect
df_data_all <- df_data_all %>%
  arrange(gvkey, fyear) %>%
  mutate(
    lag_rect = lag(rect)
  )

## compute ch_rec
df_data_all <- df_data_all %>%
  mutate(
    ch_rec = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                    rect - lag_rect, NA)
  )

## compute dch_rec
df_data_all <- df_data_all %>%
  mutate(
    dch_rec = ch_rec * 2 / (at + lag_at)
  )
```

dch_inv = changes in inventories

```
## compute ch_inv and dch_inv
df_data_all <- df_data_all %>%
  arrange(gvkey, fyear) %>%
  mutate(
    lag_invt = lag(invt),
    ch_inv = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                    invt - lag_invt, NA),
    dch_inv = ch_inv * 2 / (at + lag_at)
  )
```

soft_assets = percentage of soft assets

Bao et al. list ppegt (Legacy CST Item Number 7) in their dataset but ppent (Legacy CST Item Number 8) in their sas coding. I.e., for

subsequent computation of financial ratios they are actually using the net instead of the gross item, which is included in their raw data items. In footnote 12 they highlight the differing usage of Dechow et al. and Cecchini et al. but conclude to treat both equivalently. To be consistent, I only use the ppegt item as a raw data item and for subsequent computation of soft assets and the depreciation index.

```
## compute soft_assets
df_data_all <- df_data_all %>%
  mutate(
    soft_assets = (at - ppegt - che) / at
  )
```

ch_cs = percentage change in cash sales

```
## compute cs and ch_cs
df_data_all <- df_data_all %>%
  mutate(
    lag_rect = lag(rect),
    cs = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                sale - (rect - lag_rect), NA),
    lag_cs = lag(cs),
    ch_cs = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                   (cs - lag_cs) / lag_cs, NA)
  )
```

ch_cm = change in cash margin

```
## compute cm and ch_cm
df_data_all <- df_data_all %>%
  arrange(gvkey, fyear) %>%
  mutate(
    lag_ap = lag(ap),
    lag_invt = lag(invt),
    lag_rect = lag(rect),
    cmm = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                 (cogs-(invt-lag_invt)+(ap-lag_ap))/(sale-(rect-lag_rect)),
                 NA),
    cm = 1 - cmm,
    lag_cm = lag(cm),
    ch_cm = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                   (cm-lag_cm)/lag_cm, NA)
  )

df_data_all <- df_data_all %>%
  mutate(
    ch_cm = ifelse(is.na(cogs) | is.na(sale), NA, ch_cm)
  )
```

ch_roa = change in return on assets

```
## compute roa and ch_roa
df_data_all <- df_data_all %>%
  mutate(
    roa = ni*2/(at+lag_at),
    lag_roa = lag(roa),
    ch_roa = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                    roa - lag_roa, NA)
  )
```

ch_ib = changes in free cash flow

```
## compute ch_ib
df_data_all <- df_data_all %>%
  mutate(
    lag_ib = lag(ib),
    ch_ib = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                   (ib-lag_ib)*2/(at+lag_at), NA)
  )
```

issue = actual issuance

```
## compute actual issuance dummy
df_data_all <- df_data_all %>%
  mutate(
    issue = case_when(
      sstk > 0 ~ 1,
      dltis > 0 ~ 1,
      is.na(sstk) & is.na(dltis) ~ NA_real_,
      TRUE ~ 0
    )
  )
```

bm = book-to-market

```
## compute bm
df_data_all <- df_data_all %>%
  mutate(
    bm = ceq/(prcc_f*csho)
  )
```

Cecchini et al. 2010

dpi = depreciation index (based on Beneish 1999)

```
## compute dpi
df_data_all <- df_data_all %>%
  arrange(gvkey, fyear) %>%
```

```
  mutate(
    lag_dp = lag(dp),
    lag_ppegt = lag(ppegt),
    dpi = ifelse(lag_gvkey == gvkey & lag_fyear == fyear - 1,
                 (lag_dp/(lag_dp + lag_ppegt))/(dp/(dp + ppegt)), NA)
  )
```

reoa = retained earnings over assets (based on Summers/Sweeney 1998)

```
## compute reoa
df_data_all <- df_data_all %>%
  mutate(
    reoa = re / at
  )
```

EBIT = earnings before interest and tax (based on Summers/Sweeney 1998)

```
## compute EBIT
df_data_all <- df_data_all %>%
  mutate(
    EBIT = (ni + xint + txt)/at
  )
```

Remove all temporary variables

```
df_data_all <- df_data_all %>%
  select(-c(lag_fyear, lag_gvkey, lag_at, wc, lag_wc, ch_wc, nco, lag_nco,
            ch_nco, fin, lag_fin, ch_fin, lag_rect, ch_rec, ch_inv, lag_invt,
            cs, lag_cs, lag_ap, cmm, cm, roa, lag_roa, lag_ib, lag_dp,
            lag_ppegt, lag_cm))
```

Reformat variables' classes and reorder columns

Usually, variables which should be formatted as factors need to be manipulated manually. If the variables' values consist out of numbers, the type will be automatically set as integer (*int*) or numeric (*num*). In case of other characters, the type will be set as character (*chr*), i.e. string/textual data.

```
## change classes to factor
df_data_all <- df_data_all %>%
  mutate_each_(funs(factor(.)),
               c("misstatement", "understatement", "issue")) %>%
  mutate_each_(funs(as.integer), vars("fyear", "gvkey", "p_aaer"))

## reorder columns
col_order <- c("fyear", "gvkey", "sich", "p_aaer", "misstatement",
```

```
                "understatement", "act", "ap", "at", "ceq", "che", "cogs",
                "csho", "dlc", "dltis", "dltt", "dp", "ib", "invt", "ivao",
                "ivst", "lct", "lt", "ni", "ppegt", "pstk", "re", "rect",
                "sale", "sstk", "txp", "txt", "xint", "prcc_f", "dch_wc",
                "ch_rsst", "dch_rec", "dch_inv", "soft_assets", "ch_cs",
                "ch_cm", "ch_roa", "ch_ib", "issue", "bm", "dpi", "reoa",
                "EBIT", "c_rev", "c_rec", "c_cogs", "c_inv", "c_reserve",
                "c_debt", "c_mkt_sec", "c_inc_exp_se", "c_asset", "c_pay",
                "c_liab", "reason", "explanation")

df_data_all <- df_data_all[, col_order]

## remove temporary objects
rm(col_order)
```

Filter for missing values

Filter for at (assets, total) should eliminate incomplete data. This dataset is temporaryly stored for descriptive purposes while all firms (espec. regarding peculiarities of certain industries with e.g. specific missing variables) are maintained.

```
## remove observations with at = na (Bao et al. 2020)
df_data_all <- df_data_all %>%
  filter((!is.na(at)))
```

Filter for missing values within the raw financial data items eliminates further incomplete data. This dataset is finally used for building the model.

```
## remove observations with na (Bao et al. 2020)
df_data <- df_data_all %>%
  filter_at(vars(act, ap, at, ceq, che, cogs, csho, dlc, dltis, dltt, dp, ib,
                 invt, ivao, ivst, lct, lt, ni, ppegt, pstk, re, rect, sale,
                 sstk, txp, txt, xint, prcc_f),all_vars(!is.na(.)))

rm(df_data_all)
```

Replace special values

Some subsequent computations with financial ratio features cannot handle special values (NaN, (-)Inf) and have to be eliminated.

```
## eliminate special values
df_data <- df_data %>%
  mutate(across(c(dch_wc, ch_rsst, dch_rec, dch_inv, soft_assets, ch_cs,
                  ch_cm, ch_roa, ch_ib, issue, bm, dpi, reoa, EBIT),
                ~ ifelse(. %in% c("NaN", "Inf", "-Inf"), NA, .)))
```

Filter for Finance, Insurance and Real Estate (SIC = 6???)

Remove all remaining observations of firms' SIC starting with 6,
i.e. from the Finance, Insurance and Real Estate Industry.

```
## remove observations with SIC = 6????
df_data <- df_data %>%
  filter(!sich %in% (6000:6999))
```

Normalize raw financial data items in separate dataframe

```
model_var_norm <- c("act","ap","at","ceq","che","cogs","csho","dlc","dltis",
                    "dltt","dp","ib","invt","ivao","ivst","lct","lt","ni",
                    "ppegt","pstk","re","rect","sale","sstk","txp","txt",
                    "xint","prcc_f")

df_data_norm <- df_data %>%
  rowwise() %>%
  mutate(norm_vec = sqrt(sum(act^2,ap^2,at^2,ceq^2,che^2,cogs^2,csho^2,dlc^2,
                             dltis^2,dltt^2,dp^2,ib^2,invt^2,ivao^2,ivst^2,
                             lct^2,lt^2,ni^2,ppegt^2,pstk^2,re^2,rect^2,sale^2,
                             sstk^2,txp^2,txt^2,xint^2,prcc_f^2)))

df_data_norm <- df_data_norm %>%
  rowwise %>%
  mutate(across(all_of(model_var_norm), ~ . / norm_vec)) %>%
  select(-norm_vec)
```

Store dataframes as RDA files

```
setwd(filepath_preprocess)

## save raw and normalized dataframes
save(df_data, file = "df_data.Rda")
save(df_data_norm, file = "df_data_norm.Rda")

## remove temporary objects
rm(df_data, df_data_norm, model_var_norm)
```

Appendix B: Code – Descriptive Statistics

Setup

```
## load packages
library(tidyverse)
library(knitr)
library(kableExtra)
```

Import data

```
setwd(filepath_preprocess)

## import raw and normalized dataframes
load(file = "df_data.rda")
load(file = "df_data_norm.rda")
```

Descriptive Tables

Table: Firm Years & Fraud Frequency over Time (excluding SIC = 6???)

```
## create descriptive table with firm years and fraud frequencies
df_data_distribution <- df_data %>%
  select(fyear, sich, misstatement) %>%
  group_by(fyear) %>%
  summarise(
    num_total = n(),
    num_misstate = sum(misstatement == 1),
    percentage_misstate = round((num_misstate / num_total)*100, digits = 4)
  )
```

Table: Number of Firms and Frequency of Fraud over Years

```
## create summarized bottom row
total_row <- data.frame(
  fyear = "Total",
  num_total = sum(df_data_distribution$num_total),
  num_misstate = sum(df_data_distribution$num_misstate),
  percentage_misstate = round(mean(df_data_distribution$percentage_misstate),
                              digits = 4)
)

df_data_distribution <- df_data_distribution %>%
  mutate(fyear = as.character(fyear))

df_data_distribution <- add_row(df_data_distribution, total_row)

df_data_distribution <- df_data_distribution %>%
  rename(c("Year" = fyear,
```

```
            "Number of Firms" = num_total,
            "Number of Fraud Firms" = num_misstate,
            "Percentage of Fraud Firms" = percentage_misstate))

## table which illustrates the number of firm and fraud years with the
## corresponding percentages (with and without finance industries)
table_distribution <- kable(
  df_data_distribution,
  format = "html",
  col.names = c("Year",
                "Number of Firms",
                "Number of Fraud Firms",
                "Percentage of Fraud Firms")
  ) %>%
  row_spec(42, bold = TRUE) %>%
  kable_styling(bootstrap_options  = "striped", full_width = FALSE)

## remove temporary objects
rm(total_row)
```

Filter time range

```
## filter for time range between 1990 (Bao et al. 2020)
## and 2019 (last detected misstatement by AAERs until 2021)
df_data <- df_data %>%
  filter(fyear %in% (1990:2019))

df_data_norm <- df_data_norm %>%
  filter(fyear %in% (1990:2019))
```

Accounts affected by Fraud: 1990 - 2019

```
## select variables of interest and filter for fraud cases
df_data_fraud <- df_data %>%
  select(fyear, gvkey, sich, p_aaer, misstatement, understatement, c_rev,
         c_rec, c_cogs, c_inv, c_reserve, c_debt, c_mkt_sec, c_inc_exp_se,
         c_asset, c_pay, c_liab) %>%
  filter(misstatement == 1) %>%
  mutate(acc_affected = rowSums(select(., c_rev, c_rec, c_cogs, c_inv,
                                       c_reserve, c_debt, c_mkt_sec,
                                       c_inc_exp_se, c_asset, c_pay, c_liab),
                                na.rm = TRUE))


## number of Fraudulent Firm Years (1990-2019)
fraud_firmyears_19902019 <- as.numeric(nrow(df_data_fraud))


## count affected accounts per Fraud Case
df_data_fraud_no <- df_data_fraud %>%
  group_by(acc_affected) %>%
  count() %>%
  arrange(acc_affected) %>%
```

```
  rename(c("Number of Accounts Affected" = acc_affected,
           "Frequency" = n)) %>%
  mutate(Percentage = (round((Frequency / fraud_firmyears_19902019),
                             digits = 4)))

## create table
table_fraud_no <- kable(
  df_data_fraud_no,
  format = "html",
  col.names = c("Number of Accounts Affected", "Frequency", "Percentage")
  ) %>%
  kable_styling(bootstrap_options  = "striped", full_width = FALSE)

## count affected accounts overall
df_data_fraud_acc <- df_data_fraud %>%
  summarise(
    "Misstated revenue" = sum(c_rev),
    "Misstated accounts receivable" = sum(c_rec),
    "Misstated cost of goods sold" = sum(c_cogs),
    "Misstated inventory" = sum(c_inv),
    "Misstated reserve account" = sum(c_reserve),
    "Misstated allowance for bad debt" = sum(c_debt),
    "Misstated marketable securities" = sum(c_mkt_sec),
    "Misstatement of other expense/shareholder equity account" =
      sum(c_inc_exp_se),
    "Capitalized costs as assets" = sum(c_asset),
    "Misstated payables" = sum(c_pay),
    "Misstated liabilities" = sum(c_liab)
  ) %>%
  t() %>%
  as.data.frame() %>%
  rename(Frequency = V1) %>%
  rownames_to_column("Account Affected") %>%
  arrange(desc(Frequency)) %>%
  mutate(Percentage = (round((Frequency / fraud_firmyears_19902019),
                             digits = 4)))

table_fraud_acc <- kable(
  df_data_fraud_acc,
  format = "html",
  col.names = c("Account Affected", "Frequency", "Percentage")
  ) %>%
  kable_styling(bootstrap_options  = "striped", full_width = FALSE)

## remove temporary objects
rm(df_data_fraud, fraud_firmyears_19902019)
```

Store Dataframes as RDA files and Tables

```
setwd(filepath_descriptive)

## save preprocessed dataframes
save(df_data, file = "df_data.Rda")
save(df_data_norm, file = "df_data_norm.Rda")
save(df_data_fraud_acc, file = "df_data_fraud_acc.Rda")
save(df_data_fraud_no, file = "df_data_fraud_no.Rda")
```

```
## save csv files for desriptive purposes
write.csv2(df_data_distribution, file = "df_data_distribution.csv")
write.csv2(df_data_fraud_acc, file = "df_data_fraud_acc.csv")
write.csv2(df_data_fraud_no, file = "df_data_fraud_no.csv")

## remove temporary objects
rm(df_data, df_data_norm, df_data_fraud_acc, df_data_fraud_no,
   df_data_distribution, table_distribution, table_fraud_acc, table_fraud_no)
```

Appendix C: Code – Model Training

Setup

```
## basic packages
library(tidyverse)
library(knitr)
library(rmarkdown)
theme_set(theme_classic())

## data processing
library(data.table)

## training and validation
library(caret)
library(adabag)

## performance evaluation
library(pROC)

## visualization
library(svglite)
```

Import data

```
setwd(filepath_descriptive)

## import preprocessed dataframes
load(file = "df_data.rda")
load(file = "df_data_norm.rda")
```

Train benchmark models

Below, two logistic regression model comparable to the benchmark models by Bao et al. (2020) are trained based on: 1) 14 financial ratios logit 2) 28 raw financial items (normalized) logit

The models are trained (1990-2001) only for one test period (2003) to check if the in previous research demonstrated superior performance of more complex models also applies for the models trained belwo. Both models are subsequently compared to the RUSBoost model.

```
## model formulas
log_ratio_formula <- misstatement ~ dch_wc + ch_rsst + dch_rec + dch_inv +
                                     soft_assets + ch_cs + ch_cm + ch_roa +
                                     ch_ib + issue + bm + dpi + reoa + EBIT
log_norm_formula <- misstatement ~ act + ap + at + ceq + che + cogs + csho +
                                   dlc + dltis + dltt + dp + ib + invt +
                                   ivao + ivst + lct + lt + ni + ppegt +
                                   pstk + re + rect + sale + sstk + txp +
                                   txt + xint + prcc_f
```

```
## training periods
data_train_ratio_03 <- df_data %>%
  filter(between(fyear, 1990, 2001))
data_train_norm_03 <- df_data_norm %>%
  filter(between(fyear, 1990, 2001))

## test period
data_test_ratio_03 <- df_data %>%
  filter(fyear == 2003)
data_test_norm_03 <- df_data_norm %>%
  filter(fyear == 2003)

## logistic regression models (logit)
model_log_ratio_03 <- glm(log_ratio_formula,
                          family = "binomial",
                          data = data_train_ratio_03)
model_log_norm_03 <- glm(log_norm_formula,
                         family = "binomial",
                         data = data_train_norm_03)

## save logit models
saveRDS(model_log_norm_03, "model_log_norm_03.rds")
save(model_log_norm_03, file = "model_log_norm_03.RData")
saveRDS(model_log_ratio_03, "model_log_ratio_03.rds")
save(model_log_ratio_03, file = "model_log_ratio_03.RData")

## predictor
model_log_ratio_03_pred <- predict(model_log_ratio_03,
                             newdata = data_test_ratio_03,
                             type = "response")
model_log_norm_03_pred <- predict(model_log_norm_03,
                             newdata = data_test_norm_03,
                             type = "response")

## performance AUC
model_log_ratio_03_auc <- roc(response = data_test_ratio_03$misstatement,
                              predictor = model_log_ratio_03_pred,
                              metric = "auc")
model_log_norm_03_auc <- roc(response = data_test_norm_03$misstatement,
                             predictor = model_log_norm_03_pred,
                             metric = "auc")

## remove temporary objects
rm(model_log_ratio_03, model_log_norm_03, model_log_ratio_03_pred,
   model_log_norm_03_pred, df_data_norm, data_test_ratio_03,
   data_test_norm_03)
```

Train RUSBoost models

Hyperparameter Tuning

Data split

For purposes of hyperparameter tuning I use the following training and validation periods: - Training Period: 1990-1999 - Validation Period: 2000-2001

This step is conducted within the `trainControl`-function using the method `timeslice`.

Sampling

RUSBoost is manuallay adapted incorporating the `AdaBoost.M1`-algorithm and the random downsampling (also in `trainControl`). I use this manual approach rather than a predefined function (e.g. `rus()`from the `embc`-package because of compatibility issues, which would otherwise arise in combination with further packages).

Hyperparameters

The `AdaBoost.M1`-algorithm only requires tuning of three parameters (based on `adabag`-package). Depending on the software and package, these parameters may vary slightly. Here, the following parameters can be tuned: - `mfinal`: Number of trees - `maxdepth`: Maximum Tree Depth - `coeflearn`: Coefficient Type

I conduct a grid search, which varies the number of trees from 100:3,000 (steps of 100), the maximum depth of trees from 1 to 10, and all three available learning coefficients (Breiman, Freund, and Zhu).

Below the package `doParallel` and its main function is used for parallel computing of similar processes. In all places where this made sense, I have parallelized the computing processes in order to save computing time.

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)

## select training data for normalized and non-normalized finanical items
data_train_03 <- df_data %>%
     filter(fyear >= 1990 & fyear <= 2001) %>%
     mutate(misstatement = ifelse(misstatement == 0,"No","Yes")) %>%
     arrange(fyear)

data_train_norm_03 <- df_data_norm %>%
     filter(fyear >= 1990 & fyear <= 2001) %>%
     mutate(misstatement = ifelse(misstatement == 0,"No","Yes")) %>%
     arrange(fyear)

## formula
ada_formula <- misstatement ~ act + ap + at + ceq + che + cogs + csho +
                              dlc + dltis + dltt + dp + ib + invt + ivao +
                              ivst + lct + lt + ni + ppegt + pstk + re +
                              rect + sale + sstk + txp + txt + xint + prcc_f

## grid search
ada_grid <- expand.grid(mfinal = c(1:30)*100,
                        maxdepth = c(1:10),
                            coeflearn = c("Breiman", "Freund", "Zhu"))

## further training parameters (timeslices, sampling, and metrics)
```

```
## set training data
obs_train <- data_train_03 %>%
  filter(fyear >= 1990 & fyear <= 1999) %>%
  nrow()

## set validation data
obs_valid <- data_train_03 %>%
  filter(fyear >= 2000 & fyear <= 2001) %>%
  nrow()

## assign train and validation periods, set timeslices and random undersampling
ada_ctrl <- trainControl(method = "timeslice",
                         initialWindow = obs_train,
                               horizon = obs_valid,
                               fixedWindow = FALSE,
                         sampling = "down",
                         classProbs = TRUE,
                               summaryFunction = twoClassSummary)

## train model (non-normalized items)
model_ada_down_valid <- caret::train(ada_formula,
                                     data = data_train_03,
                                     method = "AdaBoost.M1",
                                     metric = "ROC",
                                     tuneGrid = ada_grid,
                                     trControl = ada_ctrl)

## train models (normalized items)
model_ada_down_norm_valid <- caret::train(ada_formula,
                                          data = data_train_norm_03,
                                          method = "AdaBoost.M1",
                                          metric = "ROC",
                                          tuneGrid = ada_grid,
                                          trControl = ada_ctrl)

## stop parallel computing
stopCluster(cl)

## remove temporary objects
rm(obs_train, obs_valid, ada_grid, ada_ctrl, cl)
```

Visualization

The following code visualizes the results of the hyperparameter tuing in two different ways. Based on these results the hyperparameters for the subsequent model trainings are selected.

```
## visualization 1 of hyperparameter tuning's grid search
trellis.par.set(caretTheme())
plot(model_ada_down_valid)

ggplot(model_ada_down_valid)

plot(model_ada_down_valid, metric = "Accuracy", plotType = "level",
     scales = list(x = list(rot = 90)))
```

```
valided_plot <- ggplot(model_ada_down_valid)


## visualization 2 of hyperparameter tuning's grid search
valided_plot2 <- ggplot(model_ada_down_valid,
                        metric = "ROC",
                        plotType = "level",
                        scales = list(x = list(rot = 90))) +
                        scale_fill_gradient2(low="magenta",
                                             high="cyan",
                                             mid="white",
                                             midpoint=0.77) +
                        scale_x_discrete(breaks = c(0,
                                                    1000,
                                                    2000,
                                                    3000),
                                         labels = c("0",
                                                    "1000",
                                                    "2000",
                                                    "3000"))


setwd("C:/Users/Loesse/sciebo/5-Projekt/30_PAPER3/70_CODE/30_training")

## save validation model
saveRDS(model_ada_down_valid, "model_ada_down_valided.rds")
save(model_ada_down_valid, file = "model_ada_down_valided.RData")

## save validation plots
ggsave(file="valided_plot_wide.svg", plot=valided_plot, width=15, height=8)
ggsave(file="valided_plot_2.svg", plot=valided_plot2, width=10, height=8)

## print validation plots
print(valided_plot)
print(valided_plot2)
```

Hyperparater selection

Number of trees:

Around the size of 1000 the performance is still quite volatile. The performance seems to stabilize, especially for deeper trees, in the range between 2000 and 3000 trees. Thus, I set the size of the ensemble to 3,000 trees: `mfinal = 3000`.

Maximum depth of trees:

Except for one outlier (`maxdepth = 6` and `coefleran = "Breiman"`), deeper trees seem to be superior through all types of coefficients. Especially, the deepest trees with a depth of 10 in the case of the learning coefficients "Freund" and "Zhu" achieves a superior performance. Thus, I set the maximum depth of the trees to 10: `maxdepth = 10`.

Coefficient type:

Among the types of learning coefficients, "Freund" achieves the most consistent results for deep trees. Thus, I set the learning coefficient to "Freund": `coeflearn = "Freund"`.

Costs and Threshold

Due to different misclassification costs, depending on the type of error, I determine a cost-efficient threshold for subsequent predictions. Here, the calculation of costs cannot be completely objective. The paper addresses multiple perspectives, especially comprising auditors, investors and regulators/enforcement authorities. Their individual misclassification costs vary, however, the literature is unanimous that false negatives are significantly more costly than false positives. Beneish (1997, 1999) considers a cost ratio of 20:1 to 30:1 to be appropriate for investors. Cecchini et al. (2010) differentiate less between the addressees and determines a ratio of 50:1 as adequate. I use cost ratios from 10:1 to 100:1 to calculate an abstract cost measure. The range covers the previously mentioned ratios and includes a reserve for more severe cost ratios.

To determine the optimal threshold, I use costs calculated for unused data from 2002.

Predictor based on 2002 for cost-efficient Cutoff Optimization

The selection of cost efficient thresholds under the previously described assumptions, is conducted using the data from 2002, which is neither part of the training data nor of the test data. The naming of the R objects only refers to "03" since this is the first period on which the thresholds are later applied. Therefore, in a first step, a predictor is required. I manually adjusted the computation of the following two chunks of code, the predictor and costs were calculated for the 1) normalized data and for the 2) non-normalized data. Thus, the code shown refers to the objects of the non-normalized version.

```
## select data from 2002 as a basis to select cost efficient thresholds
data_cutoff_03 <- df_data %>%
  filter(fyear == 2002)

## build a predictor based on unused data from 2002
model_ada_down_valid_pred <- predict(model_ada_down_valid,
                             newdata = data_cutoff_03,
                             type = "prob")

model_ada_down_valid_pred <- model_ada_down_valid_pred[["Yes"]]

## compute performance using auc
model_ada_down_valid_auc <- roc(response = data_cutoff_03$misstatement,
                        predictor= model_ada_down_valid_pred,
                        metric = "auc")

## plot auc
plot_model_ada_down_valid_auc <- plot(model_ada_down_valid_auc,
     xlim=c(1,0),
```

```
     ylim=c(0,1),
     col="blue",
     print.auc=TRUE,
     auc.polygon=TRUE,
     auc.polygon.col="gray",
     max.auc.polygon=TRUE)

print(plot_model_ada_down_valid_auc)

## remove temporary objects
rm(model_ada_down_valid_pred)
```

Comparision of different Cost-Ratios for varying Cutoff Thresholds

As described above, the effects of different thresholds are computed for 1) normalized and 2) non-normalized data. This adjustment has been done manually after the normalized version had been processed. Therefore, the code for the non-normalized version is shown below.

```
## select actual misstatements
model_ada_down_valid_act <- data_cutoff_03$misstatement

## empty dataframe to be filled
cutoff_optimization <- data.frame(Cost_Ratio = numeric(0),
                                  Threshold = numeric(0),
                                  Total_Cost = numeric(0),
                                  TPR = numeric(0),
                                  FPR = numeric(0))


## loop for computing metrics for different cutoffs
for (threshold in seq(0, 1, by = 0.01)) {
  for (cost_fn_factor in seq(10, 100, by = 10)) {

    cost_fp <- 1
    cost_fn <- cost_fn_factor

  ## compute labels based on threshold
  predicted_labels <- as.factor(ifelse(model_ada_down_valid_pred >= threshold,
                                       1, 0))

  ## compute tp, fp, tn, fn
  tp <- sum(predicted_labels == 1 & model_ada_down_valid_act == 1)
  fp <- sum(predicted_labels == 1 & model_ada_down_valid_act == 0)
  tn <- sum(predicted_labels == 0 & model_ada_down_valid_act == 0)
  fn <- sum(predicted_labels == 0 & model_ada_down_valid_act == 1)

  ## compuate total misclassification costs
  total_cost <- (cost_fp * fp) + (cost_fn * fn)

  ## compute tpr, fpr
  tpr <- tp / (tp + fn)
  fpr <- fp / (fp + tn)

  ## append results to dataframe
```

```
  cutoff_optimization <- bind_rows(cutoff_optimization,
                                   data.frame(Cost_Ratio = cost_fn_factor,
                                              Threshold = threshold,
                                              Total_Cost = total_cost,
                                              TPR = tpr,
                                              FPR = fpr))
  }
}

## tabulate optimal cutoffs based on cost ratio and minimal total costs
cutoff_optimals <- cutoff_optimization %>%
  group_by(Cost_Ratio) %>%
  summarize(Min_Total_Cost = min(Total_Cost),
            Associated_Threshold = Threshold[which.min(Total_Cost)]) %>%
  mutate(Cost_Ratio = c("10:1", "20:1", "30:1", "40:1", "50:1", "60:1",
                        "70:1", "80:1", "90:1", "100:1")) %>%
  mutate(Cost_Ratio = as.factor(Cost_Ratio))

cutoff_optimals$Cost_Ratio <- factor(cutoff_optimals$Cost_Ratio,
                                     levels = c("100:1", "90:1", "80:1",
                                                "70:1", "60:1", "50:1",
                                                "40:1", "30:1", "20:1",
                                                "10:1"))

## mutate tabular cutoff data
cutoff_optimization <- cutoff_optimization %>%
  mutate(Cost_Ratio = paste(as.character(Cost_Ratio),":1"), collapse= NULL) %>%
  mutate(Cost_Ratio = gsub(" ", "", Cost_Ratio)) %>%
  mutate(Cost_Ratio = as.factor(Cost_Ratio))

cutoff_optimization$Cost_Ratio <- factor(cutoff_optimization$Cost_Ratio,
                                         levels = c("100:1", "90:1", "80:1",
                                                    "70:1", "60:1", "50:1",
                                                    "40:1", "30:1", "20:1",
                                                    "10:1"))


## figure total costs with different cost ratios for varying thresholds
plot_total_costs <- ggplot(cutoff_optimization, aes(x = Threshold)) +
  geom_line(aes(y = Total_Cost, color = Cost_Ratio), size = 1) +
  geom_point(data = cutoff_optimals, aes(x = Associated_Threshold,
                                         y = Min_Total_Cost,
                                         color = Cost_Ratio),
             size = 3, shape = 19) +
  labs(x = "Cutoff Threshold", y = "Cost Units") +
  labs(color = "Cost Ratio") +
  theme_minimal() +
  theme(axis.line = element_line(color = "black"))

print(cutoff_optimals)
print(plot_total_costs)


## extract optimal cutoff for a assumed cost ratio of 50:1
cutoff_optimal <- cutoff_optimals %>%
  filter(Cost_Ratio == "50:1") %>%
  pull(Associated_Threshold)
```

```
## save figure with costs for varying thresholds and table with optimal cutoffs
ggsave(file="plot_total_costs.svg", plot=plot_total_costs, width=12, height=8)
write.csv2(cutoff_optimals, file = "cutoff_optimals.csv")


## remove temporary objects
rm(threshold, tp, tn, fp, fn, tpr, fpr, total_cost, cost, cost_fn,
   cost_fn_factor, cost_fp, cutoff_optimization, cutoff_optimals,
   plot_total_costs)
```

Train models 2003 - 2019

Based on the hyperparameter optimization above, I train multiple classification models for every single test year from 2003 to 2019 below. I train models varying the two aspects: 1. Features: normalized vs. non-normalized financial data items (normalized version indicated by suffix "_norm")
2. Training period: rolling 10 years window or all available preceeding years (rolling 10 years indicated by suffix "_fix")

With regard to the training period: The two years before the test year are excluded for both options. Thus, regarding e.g. the test year 2005, in the case of a rolling 10 years window the training data covers data from 1993-2002 and in the case of all available preceeding years, the training data covers data from 1990-2002.

Therefore, for each year of test data 4 different models are trained: 1. normalized & 10 years rolling window 2. normalized & all available preceeding years 3. non-normalized & 10 years rolling window 4. non-normalized & all available preceeding years

Set formula, hyperparameter and sampling approach

The formula sets the target variable "misstatement" and incorporates 28 financial data items as features. Hyperparameters are set as described above, while the sampling procedure is set as random undersampling.

```
## formula
ada_formula <- misstatement ~ act + ap + at + ceq + che + cogs + csho +
                              dlc + dltis + dltt + dp + ib + invt + ivao +
                              ivst + lct + lt + ni + ppegt + pstk + re +
                              rect + sale + sstk + txp + txt + xint + prcc_f

## hyperparameters
ada_grid_test <- expand.grid(mfinal = 3000,
                             maxdepth = 10,
                                 coeflearn = "Freund")

## sampling
ada_ctrl_test <- trainControl(method = "none",
                              sampling = "down",
                              classProbs = TRUE,
                                  summaryFunction = twoClassSummary)
```

Train Models: Option 1

Features: normalized
Training period: 10 years rolling window

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## loop for each test year
for (testyear in seq(2003, 2019, by = 1)) {

  ## select training data
  data_train <- df_data_norm %>%
    filter(between(fyear, testyear-11, testyear-2)) %>%
    mutate(misstatement = ifelse(misstatement == 0,"No","Yes"))


  ## train model
  model <- caret::train(ada_formula,
                        data = data_train,
                        method = "AdaBoost.M1",
                        metric = "ROC",
                        tuneGrid = ada_grid_test,
                        trControl = ada_ctrl_test)

  ## rename model
  model_name <- paste0("model_rusboost_norm_fix_",
                       substr(as.character(testyear),
                              nchar(as.character(testyear)) - 1,
                              nchar(as.character(testyear))))
  assign(model_name, model)

  ## save trained models
  save(model, file = paste0(model_name, ".RData"))

  ## remove temporary objects
  rm(model, model_name, data_train)

}

## stop parallel computing
stopCluster(cl)
```

Train Models: Option 2

Features: normalized
Training period: all available preceeding years

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## loop for each test year
for (testyear in seq(2003, 2019, by = 1)) {

  ## select training data
  data_train <- df_data_norm %>%
    filter(between(fyear, 1990, testyear-2)) %>%
    mutate(misstatement = ifelse(misstatement == 0,"No","Yes"))


  ## train model
  model <- caret::train(ada_formula,
                        data = data_train,
                        method = "AdaBoost.M1",
                        metric = "ROC",
                        tuneGrid = ada_grid_test,
                        trControl = ada_ctrl_test)

  ## rename model
  model_name <- paste0("model_rusboost_norm_",
                       substr(as.character(testyear),
                              nchar(as.character(testyear)) - 1,
                              nchar(as.character(testyear))))
  assign(model_name, model)

  ## save trained models
  save(model, file = paste0(model_name, ".RData"))

  ## remove temporary objects
  rm(model, model_name, data_train)

}

## stop parallel computing
stopCluster(cl)
```

Train Models: Option 3

Features: non-normalized
Training period: 10 years rolling window

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## loop for each test year
for (testyear in seq(2003, 2019, by = 1)) {
```

```
  ## select training data
  data_train <- df_data %>%
    filter(between(fyear, testyear-11, testyear-2)) %>%
    mutate(misstatement = ifelse(misstatement == 0,"No","Yes"))


  ## train model
  model <- caret::train(ada_formula,
                        data = data_train,
                        method = "AdaBoost.M1",
                        metric = "ROC",
                        tuneGrid = ada_grid_test,
                        trControl = ada_ctrl_test)

  ## rename model
  model_name <- paste0("model_rusboost_fix_",
                       substr(as.character(testyear),
                              nchar(as.character(testyear)) - 1,
                              nchar(as.character(testyear))))
  assign(model_name, model)

  ## save trained models
  save(model, file = paste0(model_name, ".RData"))

  ## remove temporary objects
  rm(model, model_name, data_train)

}

## stop parallel computing
stopCluster(cl)
```

Train Models: Option 4

Features: non-normalized
Training period: all available preceeding years

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## loop for each test year
for (testyear in seq(2003, 2019, by = 1)) {

  ## select training data
  data_train <- df_data %>%
    filter(between(fyear, 1990, testyear-2)) %>%
    mutate(misstatement = ifelse(misstatement == 0,"No","Yes"))


  ## train model
  model <- caret::train(ada_formula,
                        data = data_train,
                        method = "AdaBoost.M1",
```

```
                        metric = "ROC",
                        tuneGrid = ada_grid_test,
                        trControl = ada_ctrl_test)

  ## rename model
  model_name <- paste0("model_rusboost_",
                       substr(as.character(testyear),
                              nchar(as.character(testyear)) - 1,
                              nchar(as.character(testyear))))
  assign(model_name, model)

  ## save trained models
  save(model, file = paste0(model_name, ".RData"))

  ## remove temporary objects
  rm(model, model_name, data_train)

}

## stop parallel computing
stopCluster(cl)
```

Classification Performance

RUSBoost

Below the classification performance for all 4 versions and for each individual test years within the period between 2003 and 2019 is calculated. To cover as much information as possible the following performance measures are included: - AUC - Accuracy - Sensitivity - Precision - Specificity - False Positive Rate

To compute the measures for all 4 versions, the following code has been run 4 times with manually adjusting for the options of normalized vs. non-normalized features and the type of training period. The manual adjustments are commented at the corresponding code lines below.

```
## dataframe for performance measures over testing periods
result_performance <- data.frame()

## loop for each test year
for(year in 2003:2019) {

  ## test data for each year
  ## manually set to df_data (non-normalized) or df_data_norm (normalized)
  test_data <- df_data %>%
    filter(fyear == year)

  ## select model for the test year
  ## manually adjust to 4 versions (suffixes "_norm" & "_fix")
  model_name <- paste0("model_rusboost_", substr(year, 3, 4))
  temp_env <- new.env()
  load(model_name, envir = temp_env)
  model <- temp_env[[ls(temp_env)[1]]]
```

```
## compute predictions for test data
predictions <- predict(model, newdata = test_data, type = "prob")[,2]

## compute ROC and AUC
roc_obj <- roc(test_data$misstatement, predictions)
auc_value <- auc(roc_obj)

## performance measures based on cost-efficient threshold
## manually set to: for normalized data 0.51 & non-normalized data 0.53
pred_class <- ifelse(predictions >= 0.53, 1, 0)

## append predictions and classification results to test_data
test_data <- test_data %>%
  add_column(prediction = as.factor(pred_class))

test_data <- test_data %>%
  mutate(
    classification_result = case_when(
      misstatement == 1 & prediction == 1 ~ "tp",
      misstatement == 1 & prediction == 0 ~ "fn",
      misstatement == 0 & prediction == 1 ~ "fp",
      misstatement == 0 & prediction == 0 ~ "tn")
        )

## save test_data for each year
## manually adjust to 4 versions (suffixes "_norm" & "_fix")
test_data_name <- paste0("test_data_",
                         substr(as.character(year),
                                nchar(as.character(year)) - 1,
                                nchar(as.character(year))))
assign(test_data_name, test_data)

## save test data including predictions and classification results
save(test_data, file = paste0(test_data_name, ".RData"))

## compute confusion matrix
cm <- confusionMatrix(table(pred_class, test_data$misstatement), positive = "1")

accuracy <- cm$overall["Accuracy"]
sensitivity <- cm$byClass["Sensitivity"]
specificity <- cm$byClass["Specificity"]
precision <- cm$byClass["Pos Pred Value"]
false_positive_rate <- 1 - specificity

## store performance values in temporary dataframe
temp_df <- data.frame(
  "Year" = year,
  "AUC" = auc_value,
  "Accuracy" = accuracy,
  "Sensitivity" = sensitivity,
  "Precision" = precision,
  "Specificity" = specificity,
  "False Positive Rate" = false_positive_rate
)

## append new column to the results dataframe
result_performance <- rbind(result_performance, temp_df)

## remove temporary objects
```

```
  rm(test_data, test_data_name, list = test_data_name)

}

## adjust performance dataframe
results_performance <- result_performance %>%
  gather(key = "Metric", value = "Value", -Year) %>%
  spread(key = "Year", value = "Value")

## arrange metrics and print results table
ordered_metrics <- c("AUC", "Accuracy", "Sensitivity", "Precision",
                     "Specificity", "False.Positive.Rate")
results_performance <- results_performance[match(ordered_metrics,
                                                 results_performance$Metric), ]

## count observations per year for aggregation of performance metrics
observations_per_year <- df_data %>%
  group_by(fyear) %>%
  filter(between(fyear, 2003, 2019)) %>%
  summarise(count = n()) %>%
  t()

## save results as dataframes
## manually adjust to 4 versions (suffixes "_norm" & "_fix")
write.csv2(observations_per_year, file = "observations_per_year.csv")
write.csv2(results_performance, file = "results_performance_pos.csv")

## remove temporary objects
rm(ordered_metrics, observations_per_year, results_performance,
   result_performance, accuracy, false_positive_rate, precision,
   sensitivity, specificity, year, model, model_name, pred_class, predictions,
   roc_obj, auc_value, cm, temp_df)
```

Comparison | RUSBoost and Logit

To compare the RUSBoost model I subsequently illustrate the performance of the two benchmark models mentioned above and the performance of the RUSBoost model. This is exemplary illustrated for the performance of the test year 2003.

```
## load RUSBoost and test data
data_test_03 <- df_data %>%
  filter(fyear == 2003)
model_rusboost_03 <- readRDS("model_rusboost_03.rds")

## predictor
model_rusboost_03_pred <- predict(model_rusboost_03,
                                  newdata = data_test_03,
                                  type = "prob")
model_rusboost_03_pred <- model_rusboost_03_pred[["Yes"]]

## performance AUC
model_rusboost_03_auc <- roc(response = data_test_03$misstatement,
                             predictor= model_rusboost_03_pred,
                             metric = "auc")
```

```
## plot with all three ROC
svg("figure_classification_performance.svg")

plot(model_log_ratio_03_auc,
     xlim = c(1, 0),
     ylim = c(0, 1),
     max.auc.polygon = TRUE,
     col = "magenta")
lines(model_log_norm_03_auc,
      col = "blue")
lines(model_rusboost_03_auc,
      col = "green")
legend("bottomright", legend=c(paste("Logit Ratios (AUC = ",
                                     round(auc(model_log_ratio_03_auc),
                                           digits = 3), ")", sep=""),
                               paste("Logit Raw Items (AUC = ",
                                     round(auc(model_log_norm_03_auc),
                                           digits = 3), ")", sep=""),
                               paste("RUSBoost Raw Items (AUC = ",
                                     round(auc(model_rusboost_03_auc),
                                           digits = 3), ")", sep="")),
       col = c("magenta", "blue", "green"),
       lty = 1,
       lwd = 2,
       cex = 0.8,
       text.width = NULL)

dev.off()
```

Appendix D: Code – Model Explanations

Setup

```
## basic packages
library(tidyverse)
library(knitr)
library(rmarkdown)

## data processing
library(data.table)
library(Matrix)

## training and validation
library(glmnet)
library(gower)

## explanations
library(iml)

## visualizations
library(gridExtra)
library(grid)
library(ggridges)
library(ggthemes)
library(patchwork)
theme_set(theme_minimal())
```

Import data

```
setwd(filepath_descriptive)

## import preprocessed dataframes
load("df_data.RDa")
load("df_data_norm.RDa")
```

Set target and features

Below, features and the target are assigned, as well as names for additional columns for features explanation weights are defined. Subsequently, the prefix 'w_' refers to LIME explanations and 's_' to Shapley Values.

```
## assign features and target variables used in model training
features <- c("act", "ap", "at", "ceq", "che", "cogs", "csho", "dlc",
              "dltis", "dltt", "dp", "ib", "invt", "ivao", "ivst", "lct",
              "lt", "ni", "ppegt", "pstk", "re", "rect", "sale", "sstk",
              "txp", "txt", "xint", "prcc_f")

target <- c("misstatement")

## names for additional columns of features' weights
```

```
features_weights <- paste0("w_", features)
features_weights_shapley <- paste0("s_", features)
```

Create Predictors

Below, predictor objects are computed. Predictor objects hold the classification model and its underlying training data. These are required for subsequent computations of local explanations. Predictors are computed for each test year individually and corresponding to previous steps for both, normalized and non-normalized data. The code below is run twice, manually adjusted for the normalized and non-normalized variants. Technically, the 'Predictor' function of the 'iml' package is applied.

```
## loop for creating predictor objects for each test year (2003-2019)
## manually adjusted for both, normalized and non-normalized, data

for(testyear in 2003:2019) {

  setwd(filepath_training)

  ## select model for the test year
  ## [for normalized version manually set to "model_rusboost_norm_fix_"]
  model_name <- paste0("model_rusboost_fix_",
                       substr(testyear, 3, 4),
                       ".RData")
  temp_env <- new.env()
  load(model_name, envir = temp_env)
  model <- temp_env[[ls(temp_env)[1]]]

  ## select corresponding training data
  ## [for normalized version manually set to "df_data_norm"]
  data_train <- df_data %>%
    filter(between(fyear, testyear-11, testyear-2)) %>%
    select(target, features) # %>%

  ## create predictor for each test data year
  predictor <- Predictor$new(model,
                             data = data_train[features],
                             y = data_train$misstatement)

  ## rename predictor
  ## [for normalized version manually set to "predictor_norm_fix_"]
  predictor_name <- paste0("predictor_fix_",
                       substr(as.character(testyear),
                              nchar(as.character(testyear)) - 1,
                              nchar(as.character(testyear))))
  assign(predictor_name, predictor)

  ## save trained predictors
  setwd(filepath_evaluation)
  save(predictor, file = paste0(predictor_name, ".RData"))

  ## remove temporary objects
  rm(model, model_name, data_train, predictor, predictor_name, temp_env)
```

```
}
```

LIME

Following, LIME and Shapley Values are computed. The calculations are carried out in blocks, separately for LIME and Shapley Values. For LIME the 'LocalModel' function of the 'iml' package is applied. First, all true positives and false negatives are calculated. Subsequently, 50 false positives per test year are drawn separately at random, which are then also explained with LIME and Shapley values.

For the numerically largest group, the true negatives, no explanations are calculated, as this group of classification results - non-fraudulent observations where no risk was classified - is of the least interest.

LIME | True Positives & False Negatives

Below, explanations are computed for true positive (tp) and false negative (fn) predictions based on LIME. The code is run four times with manually setting for "tp" or "fn" and for both variants of normalized and non-normalized data.

```
## start parallel computing
 library(doParallel)
 cl <- makePSOCKcluster(8)
 registerDoParallel(cl)


## manually adjusted test year due to long computation time
for(testyear in 2003:2019) {

  ## import predictor
  setwd(filepath_evaluation)

  ## [for normalized version manually set to "predictor_norm_fix_"]
  predictor_name <- paste0("predictor_fix_",
                           substr(testyear, 3, 4),
                           ".RData")
  temp_env <- new.env()
  load(predictor_name, envir = temp_env)
  predictor <- temp_env[[ls(temp_env)[1]]]

  ## import test year data
  setwd(filepath_training)

  ## [for normalized version manually set to "test_data_norm_fix_"]
  data_name <- paste0("test_data_fix_",
                      substr(testyear, 3, 4),
                      ".RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  test_data <- temp_env[[ls(temp_env)[1]]]

  ## add empty columns for subsequently calculated feature weights
```

```
test_data <- bind_cols(test_data,
                       setNames(data.frame(
                         matrix(NA, nrow = nrow(test_data),
                                ncol = length(features_weights))),
                         features_weights))
test_data <- test_data %>%
  mutate(across(all_of(features_weights), as.double))

## for efficient memory usage
rm(temp_env)
gc()


## measure computation time
start = Sys.time()


## loop for observations
## manually set for the type of classification result ("tp" and "fn")
for (i in 1:nrow(test_data)) {
  if (test_data$classification_result[i] == "fn") {

  ## specify x.interest (only features of single observation)
  lime_features <- test_data[i,] %>%
    select(all_of(features))

  ## compute explainer for individual observation
  lime_explain <- LocalModel$new(predictor,
                                 k = 28,
                                 x.interest = lime_features)

  ## filter and select features' weights
  lime_effects <- lime_explain$results %>%
    filter(.class == "Yes") %>%
    select(feature, effect) %>%
    mutate(feature = paste0("w_", feature)) %>%
    t() %>%
    as.data.frame() %>%
    setNames(.[1,]) %>%
    slice(-1) %>%
    mutate(across(everything(), as.numeric))

  ## for efficient memory usage
  rm(lime_explain, lime_features)
  gc()
  Sys.sleep(240)
  gc()


  ## insert features' weights
  common_columns <- intersect(names(test_data), names(lime_effects))

  for(column in common_columns) {
    test_data[i, column] <- lime_effects[[column]]
  }

  ## for efficient memory usage
  rm(column, common_columns, lime_effects)
  gc()
```

```
    Sys.sleep(240)
    gc()
  }
}


## meausre computation time
print( Sys.time() - start )


## stop parallel computing
stopCluster(cl)

## save filled data
## [for normalized version manually set to "test_data_norm_"]
## [manually setting "tp" or "fn"]
setwd(filepath_evaluation)
save(test_data, file = paste0("test_data_", substr(testyear, 3, 4),
                              "_filled_", "fn", ".RData"))

rm(test_data)
gc()

}
```

LIME | False Positives

Below, explanations are computed for false positives (fp) predictions based on LIME. Due to the high number and long computation time, 50 false positive classifications are randomly selected. The code is run twice with manually setting for both variants of normalized and non-normalized data.

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## manually due to long computation time
for(testyear in 2003:2019) {

  ## import predictor
  setwd(filepath_evaluation)

  ## [for normalized version manually set to "predictor_norm_fix_"]
  predictor_name <- paste0("predictor_fix_",
                           substr(testyear, 3, 4),
                           ".RData")
  temp_env <- new.env()
  load(predictor_name, envir = temp_env)
  predictor <- temp_env[[ls(temp_env)[1]]]

  ## import test year data
  setwd(filepath_training)
```

```
## [for normalized version manually set to "test_data_norm_fix_"]
data_name <- paste0("test_data_fix_",
                    substr(testyear, 3, 4),
                    ".RData")
temp_env <- new.env()
load(data_name, envir = temp_env)
test_data <- temp_env[[ls(temp_env)[1]]]

## add empty columns for subsequently calculated feature weights
test_data <- bind_cols(test_data,
                       setNames(data.frame(
                         matrix(NA, nrow = nrow(test_data),
                                ncol = length(features_weights))),
                         features_weights))
test_data <- test_data %>%
  mutate(across(all_of(features_weights), as.double))

## for efficient memory usage
rm(temp_env)
gc()


## measure computation time
start = Sys.time()


## select randomly 50 fp observations
fp_indices <- which(test_data$classification_result == "fp")
selected_indices <- sample(fp_indices, min(length(fp_indices), 50))


## loop for observations
for (i in selected_indices) {

  ## specify x.interest (only features of single observation)
  lime_features <- test_data[i,] %>%
    select(all_of(features))

  ## compute explainer for individual observation
  lime_explain <- LocalModel$new(predictor,
                                 k = 28,
                                 x.interest = lime_features)

  ## filter and select features' weights
  lime_effects <- lime_explain$results %>%
    filter(.class == "Yes") %>%
    select(feature, effect) %>%
    mutate(feature = paste0("w_", feature)) %>%
    t() %>%
    as.data.frame() %>%
    setNames(.[1,]) %>%
    slice(-1) %>%
    mutate(across(everything(), as.numeric))

  ## for efficient memory usage
  rm(lime_explain, lime_features)
  gc()
  Sys.sleep(30)
  gc()
```

```
    ## insert features' weights
    common_columns <- intersect(names(test_data), names(lime_effects))

    for(column in common_columns) {
      test_data[i, column] <- lime_effects[[column]]
    }

    ## for efficient memory usage
    rm(column, common_columns, lime_effects)
    gc()
    Sys.sleep(45)
    gc()
  }



## measure computation time
print( Sys.time() - start )

gc()
Sys.sleep(120)
gc()

## stop parallel computing
stopCluster(cl)

## save filled data
## [for normalized version manually set to "test_data_norm_"]
setwd(filepath_evaluation)
save(test_data, file = paste0("test_data_", substr(testyear, 3, 4),
                              "_filled_", "fp", ".RData"))

rm(test_data)
gc()
Sys.sleep(240)
gc()

}
```

LIME | Visualization

Individually performed for selected examples to be illustrated. Below, illustrated for fyear = 2014, GVKEY = 14304, AAER = 3931 (additionally conducted for fyear = 2010, GVKEY = 25405, AAER = 3840).

```
## select specific observation with individual LIME (non-normalized)
df_2014_14304_raw_lime <- test_data_14_filled_lime_tp.RData %>%
  filter(gvkey == 14304) %>%
  filter(fyear == 2014) %>%
  select(features_weights) %>%
  transpose() %>%
  rename(effect = V1) %>%
  mutate(feature = features) %>%
```

```
  select(feature, everything())

## select specific observation with individual LIME (normalized)
df_2014_14304_norm_lime <- test_data_norm_14_filled_lime_tp.RData %>%
  filter(gvkey == 14304) %>%
  filter(fyear == 2014) %>%
  select(features_weights) %>%
  transpose() %>%
  rename(effect = V1) %>%
  mutate(feature = features) %>%
  select(feature, everything())


## ggplot of explanation LIME, non-normalized
plot_2014_14304_raw_lime <- df_2014_14304_raw_lime %>%
  filter(.class == "Yes") %>%
  mutate(is_special = ifelse(feature == "rect", "highlight", "normal")) %>%
  ggplot(aes(x = reorder(feature, effect), y = effect, fill = is_special)) +
    facet_wrap(~ .class, ncol = 2) +
    geom_bar(stat = "identity", alpha = 0.8) +
    scale_fill_manual(values = c("highlight" = "orange",
                                 "normal" ="darkgrey")) +
    coord_flip() +
    labs(title = "LIME, non-normalized",
         x = "Feature",
         y = "Effect") +
    guides(fill = FALSE) +
    theme(
    plot.title = element_text(hjust = 0.5)
  ) +
  scale_y_continuous(limits = c(
    -max(abs(lime_explain$results$effect)),
    max(abs(lime_explain$results$effect))))

print(plot_2014_14304_raw_lime)


## ggplot of explanation LIME, normalized
plot_2014_14304_norm_lime <- df_2014_14304_norm_lime %>%
  filter(.class == "Yes") %>%
  mutate(is_special = ifelse(feature == "rect", "highlight", "normal")) %>%
  ggplot(aes(x = reorder(feature, effect), y = effect, fill = is_special)) +
    facet_wrap(~ .class, ncol = 2) +
    geom_bar(stat = "identity", alpha = 0.8) +
    scale_fill_manual(values = c("highlight" = "orange",
                                 "normal" ="darkgrey")) +
    coord_flip() +
    labs(title = "LIME, normalized",
         x = "Feature",
         y = "Effect") +
    guides(fill = FALSE) +
    theme(
    plot.title = element_text(hjust = 0.5)
  ) +
  scale_y_continuous(limits = c(
    -max(abs(lime_explain$results$effect)),
    max(abs(lime_explain$results$effect))))

print(plot_2014_14304_norm_lime)
```

```
## combine plots
plot_2014_14304_combined_lime <-
  plot_2014_14304_norm_lime +
  plot_2014_14304_raw_lime


## save illustrations
ggsave(file="plot_2014_14304_norm_lime.svg",
       plot = plot_2014_14304_norm_lime,
       width = 5, height = 5)

ggsave(file="plot_2014_14304_raw_lime.svg",
       plot = plot_2014_14304_raw_lime,
       width = 5, height = 5)

ggsave(file="plot_2014_14304_combined_lime.svg",
       plot = plot_2014_14304_combined_lime,
       width = 10, height = 5)
```

Shapley Values

Following, Shapley Values are computed. The calculations are carried out in blocks and apply the 'Shapley' function of the 'iml' package. First, all true positives and false negatives are calculated. Subsequently, 50 false positives per test year are drawn separately at random, which are then also explained with Shapley values.

For the numerically largest group, the true negatives, no explanations are calculated, as this group of classification results - non-fraudulent observations where no risk was classified - is of the least interest.

Shapley Values | True Positives & False Negatives

Below, explanations are computed for true positive (tp) and false negative (fn) predictions based on Shapley Values. The code is run four times with manually setting for "tp" or "fn" and for both variants of normalized and non-normalized data.

```
## start parallel computing
library(doParallel)
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## manually due to long computation time
for(testyear in 2003:2019) {

  ## import predictor
  setwd(filepath_evaluation)

  ## manually set
  ## [for normalized version manually set to "predictor_norm_fix_"]
  predictor_name <- paste0("predictor_fix_",
```

```
                          substr(testyear, 3, 4),
                          ".RData")
temp_env <- new.env()
load(predictor_name, envir = temp_env)
predictor <- temp_env[[ls(temp_env)[1]]]

## import test year data
setwd(filepath_training)

## [for normalized version manually set to "test_data_norm_fix_"]
data_name <- paste0("test_data_fix_",
                    substr(testyear, 3, 4),
                    ".RData")
temp_env <- new.env()
load(data_name, envir = temp_env)
test_data <- temp_env[[ls(temp_env)[1]]]

## add empty columns for subsequently calculated feature weights
test_data <- bind_cols(test_data,
                       setNames(data.frame(
                         matrix(NA, nrow = nrow(test_data),
                                ncol = length(features_weights_shapley))),
                         features_weights_shapley))
test_data <- test_data %>%
  mutate(across(all_of(features_weights_shapley), as.double))

## for efficient memory usage
rm(temp_env)
gc()


## measure computation time
start = Sys.time()


## loop for observations
## manually set for the type of classification result ("tp" and "fn")

for (i in 1:nrow(test_data)) {
  if (test_data$classification_result[i] == "tp") {

  ## specify x.interest (only features of single observation)
  shapley_features <- test_data[i,] %>%
    select(all_of(features))

  ## compute explainer for individual observation
  shapley_explain <- Shapley$new(predictor,
                                 x.interest = shapley_features)

  ## filter and select features' weights
  shapley_effects <- shapley_explain$results %>%
    filter(class == "Yes") %>%
    select(feature, phi) %>%
    mutate(feature = paste0("s_", feature)) %>%
    t() %>%
    as.data.frame() %>%
    setNames(.[1,]) %>%
    slice(-1) %>%
    mutate(across(everything(), as.numeric))
```

```
    ## for efficient memory usage
    rm(shapley_explain, shapley_features)
    gc()
    Sys.sleep(30)
    gc()

    ## insert features' weights
    common_columns <- intersect(names(test_data), names(shapley_effects))

    for(column in common_columns) {
      test_data[i, column] <- shapley_effects[[column]]
    }

    ## for efficient memory usage
    rm(column, common_columns, shapley_effects)
    gc()
    Sys.sleep(60)
    gc()

  }
}


## meausre computation time
print( Sys.time() - start )


## stop parallel computing
stopCluster(cl)

## save filled data
## [for normalized version manually set to "test_data_norm_"]
## [manually setting "tp" or "fn"]
setwd(filepath_evaluation)
save(test_data, file = paste0("test_data_", substr(testyear, 3, 4),
                              "_filled_shapley_", "tp", ".RData"))

rm(test_data, predictor)
gc()
Sys.sleep(240)
gc()

}
```

Shapley Values | False Positives

Below, explanations are computed for false positives (fp) predictions based on Shapley Values. Due to the high number and long computation time, 50 false positive classifications are randomly selected. The code is run twice with manually setting for both variants of normalized and non-normalized data.

```
## start parallel computing
library(doParallel)
```

```
cl <- makePSOCKcluster(8)
registerDoParallel(cl)


## manually due to long computation time
for(testyear in 2003:2019) {

  ## import predictor
  setwd(filepath_evaluation)

  ## [for normalized version manually set to "predictor_norm_fix_"]
  predictor_name <- paste0("predictor_fix_",
                           substr(testyear, 3, 4),
                           ".RData")
  temp_env <- new.env()
  load(predictor_name, envir = temp_env)
  predictor <- temp_env[[ls(temp_env)[1]]]

  ## import test year data
  setwd(filepath_training)

  ## [for normalized version manually set to "test_data_norm_fix_"]
  data_name <- paste0("test_data_fix_",
                      substr(testyear, 3, 4),
                      ".RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  test_data <- temp_env[[ls(temp_env)[1]]]

  ## add empty columns for subsequently calculated feature weights
  test_data <- bind_cols(test_data,
                         setNames(data.frame(
                           matrix(NA, nrow = nrow(test_data),
                                  ncol = length(features_weights_shapley))),
                           features_weights_shapley))
  test_data <- test_data %>%
    mutate(across(all_of(features_weights_shapley), as.double))

  ## for efficient memory usage
  rm(temp_env)
  gc()


  ## measure computation time
  start = Sys.time()


  ## select randomly 50 fp observations
  fp_indices <- which(test_data$classification_result == "fp")
  selected_indices <- sample(fp_indices, min(length(fp_indices), 50))


  ## loop for observations
  for (i in selected_indices) {

    ## specify x.interest (only features of single observation)
    shapley_features <- test_data[i,] %>%
      select(all_of(features))
```

```
    ## compute explainer for individual observation
    shapley_explain <- Shapley$new(predictor,
                                   x.interest = shapley_features)

    ## filter and select features' weights
    shapley_effects <- shapley_explain$results %>%
      filter(class == "Yes") %>%
      select(feature, phi) %>%
      mutate(feature = paste0("s_", feature)) %>%
      t() %>%
      as.data.frame() %>%
      setNames(.[1,]) %>%
      slice(-1) %>%
      mutate(across(everything(), as.numeric))

    ## for efficient memory usage
    rm(shapley_explain, shapley_features)
    gc()
    Sys.sleep(30)
    gc()

    ## insert features' weights
    common_columns <- intersect(names(test_data), names(shapley_effects))

    for(column in common_columns) {
      test_data[i, column] <- shapley_effects[[column]]
    }

    ## for efficient memory usage
    rm(column, common_columns, shapley_effects)
    gc()
    Sys.sleep(60)
    gc()

  }

## measure computation time
print( Sys.time() - start )


## stop parallel computing
stopCluster(cl)

## save filled data
## [for normalized version manually set to "test_data_norm_"]
setwd(filepath_evaluation)
save(test_data, file = paste0("test_data_", substr(testyear, 3, 4),
                              "_filled_shapley_", "fp", ".RData"))

rm(test_data, predictor)
gc()
Sys.sleep(240)
gc()

}
```

Shapley Values | Visualization

Individually performed for selected examples to be illustrated. Below, illustrated for fyear = 2014, GVKEY = 14304, AAER = 3931 (additionally conducted for fyear = 2010, GVKEY = 25405, AAER = 3840).

```
## select specific observation with individual shapley values (non-normalized)
df_2014_14304_raw_shapley <- test_data_14_filled_shapley_tp.RData %>%
  filter(gvkey == 14304) %>%
  filter(fyear == 2014) %>%
  filter(row_number() == 1) %>%
  select(features_weights_shapley) %>%
  transpose() %>%
  rename(effect = V1) %>%
  mutate(feature = features) %>%
  select(feature, everything())

## select specific observation with individual shapley values (normalized)
df_2014_14304_norm_shapley <- test_data_norm_14_filled_shapley_tp.RData %>%
  filter(gvkey == 14304) %>%
  filter(fyear == 2014) %>%
  filter(row_number() == 1) %>%
  select(features_weights_shapley) %>%
  transpose() %>%
  rename(effect = V1) %>%
  mutate(feature = features) %>%
  select(feature, everything())


## ggplot of explanation
plot_2014_14304_raw_shapley <- df_2014_14304_raw_shapley %>%
  mutate(is_special = ifelse(feature == "rect", "highlight", "normal")) %>%
  ggplot(aes(x = reorder(feature, effect), y = effect, fill = is_special)) +
    geom_bar(stat = "identity", alpha = 0.8) +
    scale_fill_manual(values = c("highlight" = "orange",
                                 "normal" ="darkgrey")) +
    coord_flip() +
    labs(title = "Shapley Values, non-normalized",
         x = "Feature",
         y = "Effect") +
    guides(fill = FALSE) +
    theme(
    plot.title = element_text(hjust = 0.5)
  ) +
  scale_y_continuous(limits = c(
    -max(abs(df_2014_14304_raw_shapley$effect)),
    max(abs(df_2014_14304_raw_shapley$effect))))

print(plot_2014_14304_raw_shapley)


## ggplot of explanation
plot_2014_14304_norm_shapley <- df_2014_14304_norm_shapley %>%
  mutate(is_special = ifelse(feature == "rect", "highlight", "normal")) %>%
  ggplot(aes(x = reorder(feature, effect), y = effect, fill = is_special)) +
    geom_bar(stat = "identity", alpha = 0.8) +
    scale_fill_manual(values = c("highlight" = "orange",
                                 "normal" ="darkgrey")) +
    coord_flip() +
```

```
    labs(title = "Shapley Values, normalized",
         x = "Feature",
         y = "Effect") +
    guides(fill = FALSE) +
    theme(
    plot.title = element_text(hjust = 0.5)
  ) +
  scale_y_continuous(limits = c(
    -max(abs(df_2014_14304_norm_shapley$effect)),
    max(abs(df_2014_14304_norm_shapley$effect))))

print(plot_2014_14304_norm_shapley)


plot_2014_14304_combined_shapley <-
  plot_2014_14304_norm_shapley +
  plot_2014_14304_raw_shapley


## save illustrations
ggsave(file="plot_2014_14304_raw_shapley.svg",
       plot = plot_2014_14304_raw_shapley,
       width = 5, height = 5)

ggsave(file="plot_2014_14304_norm_shapley.svg",
       plot = plot_2014_14304_norm_shapley,
       width = 5, height = 5)

ggsave(file="plot_2014_14304_combined_shapley.svg",
       plot = plot_2014_14304_combined_shapley,
       width = 10, height = 5)
```

Combined visualization of LIME and Shapley Values

```
## additional combined plot
plot_2014_14304_combined <-
  plot_2014_14304_combined_lime /
  plot_2014_14304_combined_shapley

## save illustration
ggsave(file="plot_2014_14304_combined.svg",
       plot = plot_2014_14304_combined,
       width = 10, height = 10)
```

Appendix E: Code – Model Interpretability Evaluation

Setup

```
## basic packages
library(tidyverse)
library(knitr)
library(rmarkdown)

## data processing
library(data.table)
library(Matrix)

## visualizations
library(patchwork)
```

Set target and features

```
variable_names <- c("fyear", "gvkey", "sich", "p_aaer", "misstatement",
                    "understatement", "act", "ap", "at", "ceq", "che", "cogs",
                    "csho", "dlc", "dltis", "dltt", "dp", "ib", "invt", "ivao",
                    "ivst", "lct", "lt", "ni", "ppegt", "pstk", "re", "rect",
                    "sale", "sstk", "txp", "txt", "xint", "prcc_f", "dch_wc",
                    "ch_rsst", "dch_rec", "dch_inv", "soft_assets", "ch_cs",
                    "ch_cm", "ch_roa", "ch_ib", "issue", "bm", "dpi", "reoa",
                    "EBIT", "c_rev", "c_rec", "c_cogs", "c_inv", "c_reserve",
                    "c_debt", "c_mkt_sec", "c_inc_exp_se", "c_asset", "c_pay",
                    "c_liab", "reason", "explanation", "prediction",
                    "classification_result")

features <- c("act", "ap", "at", "ceq", "che", "cogs", "csho", "dlc",
              "dltis", "dltt", "dp", "ib", "invt", "ivao", "ivst", "lct",
              "lt", "ni", "ppegt", "pstk", "re", "rect", "sale", "sstk",
              "txp", "txt", "xint", "prcc_f")

target <- c("misstatement")

fraud_cat <- c("c_rev", "c_rec", "c_cogs", "c_inv", "c_reserve", "c_debt",
               "c_mkt_sec", "c_inc_exp_se", "c_asset", "c_pay", "c_liab")

fraud_categories <- c("rev", "rec", "cogs", "inv", "reserve", "debt",
                      "mkt_sec", "inc_exp_se", "asset", "pay", "liab")

## names for additional columns of features' weights
features_weights_lime <- paste0("w_", features)
features_weights_shapley <- paste0("s_", features)
features_weights_all <- c(features_weights_lime, features_weights_shapley)

## add columns for score values per items depending on number of fraud categories affected and based on items' ranks
features_rank_lime <- paste0(features_weights_lime, "_rank")
features_rank_shapley <- paste0(features_weights_shapley, "_rank")
```

Combine data frames per year

Necessary because of separated computations of each class and type of explanation. Combines data frames for year e.g. based on: - test_data_03_filled_tp.RData - test_data_03_filled_fn.RData - test_data_03_filled_fp.RData

Normalized dataset

```
for(testyear in 2003:2019) {

  ## LIME

  setwd(filepath_evaluation)
  data_name <- paste0("test_data_norm_", substr(testyear, 3, 4),
                      "_filled_", "tp.RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  test_data <- temp_env[[ls(temp_env)[1]]]
  df_expl_raw <- test_data

  for(class_result in c("fn", "fp")) {

    ## import data per year data and result
    data_name <- paste0("test_data_norm_", substr(testyear, 3, 4),
                        "_filled_", class_result,".RData")
    temp_env <- new.env()
    load(data_name, envir = temp_env)
    test_data <- temp_env[[ls(temp_env)[1]]]

    common_columns <- intersect(names(df_expl_raw), names(test_data))

    ## weights lime to be filled
    for (i in 1:nrow(test_data)) {
      if(test_data$classification_result[i] == class_result) {
        for(column in common_columns) {
          df_expl_raw[i, column] <- test_data[[i, column]]
        }
      }
    }
  }


  ## Shapley

  ## Add columns for Shapley's weights
  df_expl_raw <- bind_cols(df_expl_raw, setNames(data.frame(
    matrix(NA, nrow = nrow(df_expl_raw),
           ncol = length(features_weights_shapley))),
    features_weights_shapley))
  df_expl_raw <- df_expl_raw %>%
    mutate(across(all_of(features_weights_shapley), as.double))

  for(class_result in c("tp", "fn", "fp")) {

    ## import data per year data and result
    data_name <- paste0("test_data_norm_", substr(testyear, 3, 4),
                        "_filled_shapley_", class_result,".RData")
```

```
    temp_env <- new.env()
    load(data_name, envir = temp_env)
    test_data <- temp_env[[ls(temp_env)[1]]]

    ## weights lime to be filled
    for (i in 1:nrow(test_data)) {
      if(test_data$classification_result[i] == class_result) {
        for(column in features_weights_shapley) {
          df_expl_raw[i, column] <- test_data[[i, column]]
        }
      }
    }
  }

  ## store filled data frame per year
  setwd(filepath_interpretation)
  save(df_expl_raw, file = paste0("df_expl_norm_",
                                  substr(testyear, 3, 4), ".RData"))
  rm(df_expl_raw)
  gc()

}
```

Non-normalized dataset

```
for(testyear in 2003:2019) {

  ## LIME

  setwd(filepath_evaluation)
  data_name <- paste0("test_data_raw_", substr(testyear, 3, 4),
                      "_filled_", "tp.RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  test_data <- temp_env[[ls(temp_env)[1]]]
  df_expl_raw <- test_data

  for(class_result in c("fn", "fp")) {

    ## import data per year data and result
    data_name <- paste0("test_data_raw_", substr(testyear, 3, 4),
                        "_filled_", class_result,".RData")
    temp_env <- new.env()
    load(data_name, envir = temp_env)
    test_data <- temp_env[[ls(temp_env)[1]]]

    common_columns <- intersect(names(df_expl_raw), names(test_data))

    ## weights lime to be filled
    for (i in 1:nrow(test_data)) {
      if(test_data$classification_result[i] == class_result) {
        for(column in common_columns) {
          df_expl_raw[i, column] <- test_data[[i, column]]
        }
      }
    }
  }
```

```
  ## Shapley

  ## Add columns for Shapley's weights
  df_expl_raw <- bind_cols(df_expl_raw,
                           setNames(data.frame(
                             matrix(NA, nrow = nrow(df_expl_raw),
                                    ncol = length(features_weights_shapley))),
                             features_weights_shapley))
  df_expl_raw <- df_expl_raw %>%
    mutate(across(all_of(features_weights_shapley), as.double))

  for(class_result in c("tp", "fn", "fp")) {

    ## import data per year data and result
    data_name <- paste0("test_data_raw_", substr(testyear, 3, 4),
                        "_filled_shapley_", class_result,".RData")
    temp_env <- new.env()
    load(data_name, envir = temp_env)
    test_data <- temp_env[[ls(temp_env)[1]]]

    ## weights lime to be filled
    for (i in 1:nrow(test_data)) {
      if(test_data$classification_result[i] == class_result) {
        for(column in features_weights_shapley) {
          df_expl_raw[i, column] <- test_data[[i, column]]
        }
      }
    }
  }

  ## store filled data frame per year
  setwd(filepath_interpretation)
  save(df_expl_raw, file = paste0("df_expl_raw_",
                                  substr(testyear, 3, 4),
                                  ".RData"))
  rm(df_expl_raw)
  gc()

}
```

Bind over years

Bind for normalized data sets

```
## import data frame of first year (2003)
setwd(filepath_interpretation)
load("df_expl_norm_03.RData")
df_expl_norm_total <- df_expl_raw

for(testyear in 2003:2019) {
  data_name <- paste0("df_expl_norm_", substr(testyear, 3, 4), ".RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  df_expl_norm <- temp_env[[ls(temp_env)[1]]]

  ## bind subsequent years on total data frame
```

```
  df_expl_norm_total <- rbind(df_expl_norm_total, df_expl_norm)

}

## save filled data
setwd(filepath_interpretation)
save(df_expl_norm_total, file = "df_expl_norm_total.RData")
```

Bind for non-normalized data sets

```
## import data frame of first year (2003)
setwd(filepath_interpretation)
load("df_expl_raw_03.RData")
df_expl_raw_total <- df_expl_raw

for(testyear in 2003:2019) {
  data_name <- paste0("df_expl_raw_", substr(testyear, 3, 4), ".RData")
  temp_env <- new.env()
  load(data_name, envir = temp_env)
  df_expl_raw <- temp_env[[ls(temp_env)[1]]]

  df_expl_raw_total <- rbind(df_expl_raw_total, df_expl_raw)

}

## save filled data
setwd(filepath_interpretation)
save(df_expl_raw_total, file = "df_expl_raw_total.RData")
```

Filter for analyzed observations and calculate ranks

Below, features ranks are calculated depending on their contribution to the classification, measured by LIME and Shapley Values. Ranks are assigned from 1 to 28. A rank of 1 is assigned to the feature with the highest LIME or Shapley Value for a classification as a misstatement (=1) and following ranks are assigned in a descending order.

```
## import total data frames
load("df_expl_raw_total.RData")
load("df_expl_norm_total.RData")

df_expl_raw <- df_expl_raw_total[apply(
  df_expl_raw_total[features_weights_all],
  1,
  function(x) any(!is.na(x))), ]
df_expl_norm <- df_expl_norm_total[apply(
  df_expl_norm_total[features_weights_all],
  1,
  function(x) any(!is.na(x))), ]

df_expl_raw$fraud_cat_count <- rowSums(df_expl_raw[fraud_cat], na.rm = TRUE)
df_expl_norm$fraud_cat_count <- rowSums(df_expl_norm[fraud_cat], na.rm = TRUE)
```

```r
## Function for calculating ranks per row for the features values
rank_values_for_features <- function(row, features) {
  values <- as.numeric(row[features])
  ranks <- rank(-values, ties.method = "first")
  rank_names <- paste(features, "rank", sep = "_")
  names(ranks) <- rank_names
  return(ranks)
}

## Ranks for raw & lime
df_ranks <- t(apply(df_expl_raw[features_weights_lime],
                    1,
                    rank_values_for_features,
                    features = features_weights_lime))
df_ranks <- as.data.frame(df_ranks)
df_expl_raw <- bind_cols(df_expl_raw, df_ranks)

## Ranks for raw & shapley
df_ranks <- t(apply(df_expl_raw[features_weights_shapley],
                    1,
                    rank_values_for_features,
                    features = features_weights_shapley))
df_ranks <- as.data.frame(df_ranks)
df_expl_raw <- bind_cols(df_expl_raw, df_ranks)

## Ranks for norm & lime
df_ranks <- t(apply(df_expl_norm[features_weights_lime],
                    1,
                    rank_values_for_features,
                    features = features_weights_lime))
df_ranks <- as.data.frame(df_ranks)
df_expl_norm <- bind_cols(df_expl_norm, df_ranks)

## Ranks for norm & shapley
df_ranks <- t(apply(df_expl_norm[features_weights_shapley],
                    1,
                    rank_values_for_features,
                    features = features_weights_shapley))
df_ranks <- as.data.frame(df_ranks)
df_expl_norm <- bind_cols(df_expl_norm, df_ranks)

## Remove ranks for observations without weights
df_expl_norm <- df_expl_norm %>%
  rowwise() %>%
  mutate(
    all_na_vars = all(is.na(c_across(all_of(features_weights_lime)))),
    across(all_of(features_rank_lime), ~ ifelse(all_na_vars, NA, .))
  ) %>%
  select(-all_na_vars) %>%
  ungroup()

df_expl_norm <- df_expl_norm %>%
  rowwise() %>%
  mutate(
    all_na_vars = all(is.na(c_across(all_of(features_weights_shapley)))),
    across(all_of(features_rank_shapley), ~ ifelse(all_na_vars, NA, .))
  ) %>%
```

```
  select(-all_na_vars) %>%
  ungroup()

df_expl_raw <- df_expl_raw %>%
  rowwise() %>%
  mutate(
    all_na_vars = all(is.na(c_across(all_of(features_weights_lime)))),
    across(all_of(features_rank_lime), ~ ifelse(all_na_vars, NA, .))
  ) %>%
  select(-all_na_vars) %>%
  ungroup()

df_expl_raw <- df_expl_raw %>%
  rowwise() %>%
  mutate(
    all_na_vars = all(is.na(c_across(all_of(features_weights_shapley)))),
    across(all_of(features_rank_shapley), ~ ifelse(all_na_vars, NA, .))
  ) %>%
  select(-all_na_vars) %>%
  ungroup()

## save filtered data
setwd(filepath_interpretation)
save(df_expl_raw, file = "df_expl_raw.RData")
save(df_expl_norm, file = "df_expl_norm.RData")

rm(df_ranks)
gc()
```

Descriptives

TP and FN according to number of fraud categories affected

Below the frequencies of true positive and false negative classifications are calculated for both model variants and grouped by the number of account categories affected by misstatements.

```
## calculate tp and fn frequencies for different classification results
df_norm_fraud_cat_count_tp <- df_expl_norm %>%
  filter(classification_result == "tp") %>%
  select(fraud_cat_count) %>%
  table()

df_norm_fraud_cat_count_fn <- df_expl_norm %>%
  filter(classification_result == "fn") %>%
  select(fraud_cat_count) %>%
  table()

df_raw_fraud_cat_count_tp <- df_expl_raw %>%
  filter(classification_result == "tp") %>%
  select(fraud_cat_count) %>%
  table()

df_raw_fraud_cat_count_fn <- df_expl_raw %>%
  filter(classification_result == "fn") %>%
  select(fraud_cat_count) %>%
```

```
  table()


## combine dataframe
df_fraud_cat_count_tpandfn <- rbind(df_norm_fraud_cat_count_tp,
                                    df_norm_fraud_cat_count_fn,
                                    df_raw_fraud_cat_count_tp,
                                    df_raw_fraud_cat_count_fn)

df_fraud_cat_count_tpandfn <- as.data.frame(t(df_fraud_cat_count_tpandfn[,]))


## save changes in form of additional columns
setwd(filepath_interpretation)
save(df_norm_fraud_cat_count,
     file = "df_norm_fraud_cat_count.RData")
```

TP and FN according to account categories affected by fraud

Below the frequencies of true positive and false negative classifications are calculated for both model variants and grouped by the account categories affected by misstatements.

```
## calculate the frequencies of accounts affected by misstatements
fraud_cat_sums <- df_expl_norm %>%
  filter(misstatement == 1) %>%
  select(all_of(fraud_cat)) %>%
  colSums()

## calculate tp and fn frequencies for different classification results
## normalized variant
df_norm_fraud_cat_tpandfn <- data.frame(
  category = character(),
  tp = integer(),
  fn = integer()
)

for (fraud_category in fraud_cat) {

  tp_norm <- df_expl_norm %>%
    filter(!!sym(fraud_category) == 1) %>%
    filter(classification_result == "tp") %>%
    nrow()

  fn_norm <- df_expl_norm %>%
    filter(!!sym(fraud_category) == 1) %>%
    filter(classification_result == "fn") %>%
    nrow()

  df_temp_results <- cbind(fraud_category,
                           tp_norm,
                           fn_norm)

  df_norm_fraud_cat_tpandfn <- rbind(df_norm_fraud_cat_tpandfn,
                                     df_temp_results)
  }
```

```
## non-normalized variant
df_raw_fraud_cat_tpandfn <- data.frame(
  category = character(),
  tp = integer(),
  fn = integer()
)

for (fraud_category in fraud_cat) {

  tp_raw <- df_expl_raw %>%
    filter(!!sym(fraud_category) == 1) %>%
    filter(classification_result == "tp") %>%
    nrow()

  fn_raw <- df_expl_raw %>%
    filter(!!sym(fraud_category) == 1) %>%
    filter(classification_result == "fn") %>%
    nrow()

  df_temp_results <- cbind(fraud_category,
                           tp_raw,
                           fn_raw)

  df_raw_fraud_cat_tpandfn <- rbind(df_raw_fraud_cat_tpandfn,
                                    df_temp_results)
  }


## save changes in form of additional columns
setwd(filepath_interpretation)
save(df_norm_fraud_cat_tpandfn,
     file = "df_norm_fraud_cat_tpandfn.RData")
save(df_raw_fraud_cat_tpandfn,
     file = "df_raw_fraud_cat_tpandfn.RData")
```

Match features and fraud categories

Matching fraud categories with related features according to Bao et al. 2020 p. 229. However, I chose to drop the mapping of Net Income (Loss) ("ni") and Receivables, Total ("re") to Misstated Revenues ("c_rev") and Misstated Costs of Goods Sold ("c_cogs") as these would not be able to provide valuable guidance for potential users.

```
## load processed data sets
load("df_expl_raw.RData")
load("df_expl_norm.RData")

## Match features to fraud categories
c_rev_items <- c("sale")
c_rec_items <- c("rect")
c_cogs_items <- c("cogs")
c_inv_items <- c("invt")
c_reserve_items <- c("ceq")
```

```
c_debt_items <- c()
c_mkt_sec_items <- c("ivst")
c_inc_exp_se_items <- c("ceq", "csho", "dp", "ib", "ni", "pstk", "re",
                        "sstk", "txt", "xint", "prcc_f")
c_asset_items <- c("act", "at", "che", "ivao", "ppegt")
c_pay_items <- c("ap")
c_liab_items <- c("dlc", "dltis", "dltt", "lct", "lt", "txp")


## Create dummy data frame
fraud_cat_dummies <- data.frame(matrix(
  0,
  nrow = length(fraud_cat),
  ncol = length(features)
  )
)


## create names for dummy columns
features_dummies <- paste0("d_", features)

rownames(fraud_cat_dummies) <- fraud_cat
colnames(fraud_cat_dummies) <- features_dummies

items_list <- list(
  c_rev_items,
  c_rec_items,
  c_cogs_items,
  c_inv_items,
  c_reserve_items,
  c_debt_items,
  c_mkt_sec_items,
  c_inc_exp_se_items,
  c_asset_items,
  c_pay_items,
  c_liab_items
)

## Fill dummy data frame
for (i in seq_along(fraud_cat)) {
  fraud_cat_dummies[i, features %in% items_list[[i]]] <- 1
}
```

Feature Dummies

Add dummy vector

Below, dummy columns for subsequent mapping between relevant features and fraud categories are added to the dataframes.

```
load("df_expl_norm.RData")
load("df_expl_raw.RData")


## add empty dummy columns
df_expl_norm_ranks <- bind_cols(
```

```
  df_expl_norm,
  setNames(
    data.frame(
      matrix(0,
             nrow = nrow(df_expl_norm),
             ncol = length(features_dummies))),
    features_dummies))

df_expl_raw_ranks <- bind_cols(
  df_expl_raw,
  setNames(
    data.frame(
      matrix(0,
             nrow = nrow(df_expl_raw),
             ncol = length(features_dummies))),
    features_dummies))

## save changes in form of additional (empty) dummy columns
setwd(filepath_interpretation)

save(df_expl_norm_ranks, file = "df_expl_norm_ranks.RData")
save(df_expl_raw_ranks, file = "df_expl_raw_ranks.RData")
```

Fill dummy columns

Below, the dummy columns are filled. The more categories are affected, the more corresponding feature dummies will be set to 1.

```
## loop for norm data
for(row in 1:nrow(df_expl_norm_ranks)){
  for(fraud_category in fraud_cat) {
    for(dummy in features_dummies) {
df_expl_norm_ranks[row,dummy] <- ifelse(
  df_expl_norm_ranks[row,fraud_category] == 1,
  df_expl_norm_ranks[row,dummy] +
    fraud_cat_dummies[fraud_category,dummy],
  df_expl_norm_ranks[row,dummy])
}
}
}

## replace sums > 1 with 1, so that actual dummy-columns result
df_expl_norm_ranks <- df_expl_norm_ranks %>%
  mutate(across(all_of(features_dummies), ~ if_else(. > 0, 1, 0)))


## loop for raw data
for(row in 1:nrow(df_expl_raw_ranks)){
  for(fraud_category in fraud_cat) {
    for(dummy in features_dummies) {
df_expl_raw_ranks[row,dummy] <- ifelse(
  df_expl_raw_ranks[row,fraud_category] == 1,
  df_expl_raw_ranks[row,dummy] +
    fraud_cat_dummies[fraud_category,dummy],
  df_expl_raw_ranks[row,dummy])
}
```

```
}
}

## replace sums > 1 with 1, so that actual dummy-columns result
df_expl_raw_ranks <- df_expl_raw_ranks %>%
  mutate(across(all_of(features_dummies), ~ if_else(. > 0, 1, 0)))


## save changes in form of columns
setwd(filepath_interpretation)
save(df_expl_norm_ranks, file = "df_expl_norm_ranks.RData")
save(df_expl_raw_ranks, file = "df_expl_raw_ranks.RData")
```

Apply dummy columns on features ranks

The code below keeps only ranks of features being related to the affected fraud category, i.e. dropping variables not being related to any fraud category.

```
## set names for rank columns for each feature after applying dummies ("ad")
ranks_after_dummy_lime <- paste0("w_", features, "_rank_ad")
ranks_after_dummy_shapley <-  paste0("s_", features, "_rank_ad")
ranks_after_dummy_all <-  c(ranks_after_dummy_lime, ranks_after_dummy_shapley)

## load data frames
load("df_expl_norm_ranks.RData")
load("df_expl_raw_ranks.RData")


## Add columns for final ranks
df_expl_norm_ranks <- bind_cols(
  df_expl_norm_ranks,
  setNames(data.frame(matrix(0,
                             nrow = nrow(df_expl_norm_ranks),
                             ncol = length(ranks_after_dummy_all))),
           ranks_after_dummy_all))

df_expl_norm_ranks <- df_expl_norm_ranks %>%
  mutate(across(all_of(ranks_after_dummy_all), as.double))


df_expl_raw_ranks <- bind_cols(
  df_expl_raw_ranks,
  setNames(data.frame(matrix(0,
                             nrow = nrow(df_expl_raw_ranks),
                             ncol = length(ranks_after_dummy_all))),
           ranks_after_dummy_all))

df_expl_raw_ranks <- df_expl_raw_ranks %>%
  mutate(across(all_of(ranks_after_dummy_all), as.double))


## Multiply scores with dummies (norm data)

df_expl_norm_ranks_ad <- df_expl_norm_ranks %>%
```

```
  mutate(w_act_rank_ad = w_act_rank * d_act) %>%
  mutate(w_ap_rank_ad = w_ap_rank * d_ap) %>%
  mutate(w_at_rank_ad = w_at_rank * d_at) %>%
  mutate(w_ceq_rank_ad = w_ceq_rank * d_ceq) %>%
  mutate(w_che_rank_ad = w_che_rank * d_che) %>%
  mutate(w_cogs_rank_ad = w_cogs_rank * d_cogs) %>%
  mutate(w_csho_rank_ad = w_csho_rank * d_csho) %>%
  mutate(w_dlc_rank_ad = w_dlc_rank * d_dlc) %>%
  mutate(w_dltis_rank_ad = w_dltis_rank * d_dltis) %>%
  mutate(w_dltt_rank_ad = w_dltt_rank * d_dltt) %>%
  mutate(w_dp_rank_ad = w_dp_rank * d_dp) %>%
  mutate(w_ib_rank_ad = w_ib_rank * d_ib) %>%
  mutate(w_invt_rank_ad = w_invt_rank * d_invt) %>%
  mutate(w_ivao_rank_ad = w_ivao_rank * d_ivao) %>%
  mutate(w_ivst_rank_ad = w_ivst_rank * d_ivst) %>%
  mutate(w_lct_rank_ad = w_lct_rank * d_lct) %>%
  mutate(w_lt_rank_ad = w_lt_rank * d_lt) %>%
  mutate(w_ni_rank_ad = w_ni_rank * d_ni) %>%
  mutate(w_ppegt_rank_ad = w_ppegt_rank * d_ppegt) %>%
  mutate(w_pstk_rank_ad = w_pstk_rank * d_pstk) %>%
  mutate(w_re_rank_ad = w_re_rank * d_re) %>%
  mutate(w_rect_rank_ad = w_rect_rank * d_rect) %>%
  mutate(w_sale_rank_ad = w_sale_rank * d_sale) %>%
  mutate(w_sstk_rank_ad = w_sstk_rank * d_sstk) %>%
  mutate(w_txp_rank_ad = w_txp_rank * d_txp) %>%
  mutate(w_txt_rank_ad = w_txt_rank * d_txt) %>%
  mutate(w_xint_rank_ad = w_xint_rank * d_xint) %>%
  mutate(w_prcc_f_rank_ad = w_prcc_f_rank * d_prcc_f)

df_expl_norm_ranks_ad <- df_expl_norm_ranks_ad %>%
  mutate(s_act_rank_ad = s_act_rank * d_act) %>%
  mutate(s_ap_rank_ad = s_ap_rank * d_ap) %>%
  mutate(s_at_rank_ad = s_at_rank * d_at) %>%
  mutate(s_ceq_rank_ad = s_ceq_rank * d_ceq) %>%
  mutate(s_che_rank_ad = s_che_rank * d_che) %>%
  mutate(s_cogs_rank_ad = s_cogs_rank * d_cogs) %>%
  mutate(s_csho_rank_ad = s_csho_rank * d_csho) %>%
  mutate(s_dlc_rank_ad = s_dlc_rank * d_dlc) %>%
  mutate(s_dltis_rank_ad = s_dltis_rank * d_dltis) %>%
  mutate(s_dltt_rank_ad = s_dltt_rank * d_dltt) %>%
  mutate(s_dp_rank_ad = s_dp_rank * d_dp) %>%
  mutate(s_ib_rank_ad = s_ib_rank * d_ib) %>%
  mutate(s_invt_rank_ad = s_invt_rank * d_invt) %>%
  mutate(s_ivao_rank_ad = s_ivao_rank * d_ivao) %>%
  mutate(s_ivst_rank_ad = s_ivst_rank * d_ivst) %>%
  mutate(s_lct_rank_ad = s_lct_rank * d_lct) %>%
  mutate(s_lt_rank_ad = s_lt_rank * d_lt) %>%
  mutate(s_ni_rank_ad = s_ni_rank * d_ni) %>%
  mutate(s_ppegt_rank_ad = s_ppegt_rank * d_ppegt) %>%
  mutate(s_pstk_rank_ad = s_pstk_rank * d_pstk) %>%
  mutate(s_re_rank_ad = s_re_rank * d_re) %>%
  mutate(s_rect_rank_ad = s_rect_rank * d_rect) %>%
  mutate(s_sale_rank_ad = s_sale_rank * d_sale) %>%
  mutate(s_sstk_rank_ad = s_sstk_rank * d_sstk) %>%
  mutate(s_txp_rank_ad = s_txp_rank * d_txp) %>%
  mutate(s_txt_rank_ad = s_txt_rank * d_txt) %>%
  mutate(s_xint_rank_ad = s_xint_rank * d_xint) %>%
  mutate(s_prcc_f_rank_ad = s_prcc_f_rank * d_prcc_f)

## Multiply scores with dummies (raw data)
```

```
df_expl_raw_ranks_ad <- df_expl_raw_ranks %>%
  mutate(w_act_rank_ad = w_act_rank * d_act) %>%
  mutate(w_ap_rank_ad = w_ap_rank * d_ap) %>%
  mutate(w_at_rank_ad = w_at_rank * d_at) %>%
  mutate(w_ceq_rank_ad = w_ceq_rank * d_ceq) %>%
  mutate(w_che_rank_ad = w_che_rank * d_che) %>%
  mutate(w_cogs_rank_ad = w_cogs_rank * d_cogs) %>%
  mutate(w_csho_rank_ad = w_csho_rank * d_csho) %>%
  mutate(w_dlc_rank_ad = w_dlc_rank * d_dlc) %>%
  mutate(w_dltis_rank_ad = w_dltis_rank * d_dltis) %>%
  mutate(w_dltt_rank_ad = w_dltt_rank * d_dltt) %>%
  mutate(w_dp_rank_ad = w_dp_rank * d_dp) %>%
  mutate(w_ib_rank_ad = w_ib_rank * d_ib) %>%
  mutate(w_invt_rank_ad = w_invt_rank * d_invt) %>%
  mutate(w_ivao_rank_ad = w_ivao_rank * d_ivao) %>%
  mutate(w_ivst_rank_ad = w_ivst_rank * d_ivst) %>%
  mutate(w_lct_rank_ad = w_lct_rank * d_lct) %>%
  mutate(w_lt_rank_ad = w_lt_rank * d_lt) %>%
  mutate(w_ni_rank_ad = w_ni_rank * d_ni) %>%
  mutate(w_ppegt_rank_ad = w_ppegt_rank * d_ppegt) %>%
  mutate(w_pstk_rank_ad = w_pstk_rank * d_pstk) %>%
  mutate(w_re_rank_ad = w_re_rank * d_re) %>%
  mutate(w_rect_rank_ad = w_rect_rank * d_rect) %>%
  mutate(w_sale_rank_ad = w_sale_rank * d_sale) %>%
  mutate(w_sstk_rank_ad = w_sstk_rank * d_sstk) %>%
  mutate(w_txp_rank_ad = w_txp_rank * d_txp) %>%
  mutate(w_txt_rank_ad = w_txt_rank * d_txt) %>%
  mutate(w_xint_rank_ad = w_xint_rank * d_xint) %>%
  mutate(w_prcc_f_rank_ad = w_prcc_f_rank * d_prcc_f)

df_expl_raw_ranks_ad <- df_expl_raw_ranks_ad %>%
  mutate(s_act_rank_ad = s_act_rank * d_act) %>%
  mutate(s_ap_rank_ad = s_ap_rank * d_ap) %>%
  mutate(s_at_rank_ad = s_at_rank * d_at) %>%
  mutate(s_ceq_rank_ad = s_ceq_rank * d_ceq) %>%
  mutate(s_che_rank_ad = s_che_rank * d_che) %>%
  mutate(s_cogs_rank_ad = s_cogs_rank * d_cogs) %>%
  mutate(s_csho_rank_ad = s_csho_rank * d_csho) %>%
  mutate(s_dlc_rank_ad = s_dlc_rank * d_dlc) %>%
  mutate(s_dltis_rank_ad = s_dltis_rank * d_dltis) %>%
  mutate(s_dltt_rank_ad = s_dltt_rank * d_dltt) %>%
  mutate(s_dp_rank_ad = s_dp_rank * d_dp) %>%
  mutate(s_ib_rank_ad = s_ib_rank * d_ib) %>%
  mutate(s_invt_rank_ad = s_invt_rank * d_invt) %>%
  mutate(s_ivao_rank_ad = s_ivao_rank * d_ivao) %>%
  mutate(s_ivst_rank_ad = s_ivst_rank * d_ivst) %>%
  mutate(s_lct_rank_ad = s_lct_rank * d_lct) %>%
  mutate(s_lt_rank_ad = s_lt_rank * d_lt) %>%
  mutate(s_ni_rank_ad = s_ni_rank * d_ni) %>%
  mutate(s_ppegt_rank_ad = s_ppegt_rank * d_ppegt) %>%
  mutate(s_pstk_rank_ad = s_pstk_rank * d_pstk) %>%
  mutate(s_re_rank_ad = s_re_rank * d_re) %>%
  mutate(s_rect_rank_ad = s_rect_rank * d_rect) %>%
  mutate(s_sale_rank_ad = s_sale_rank * d_sale) %>%
  mutate(s_sstk_rank_ad = s_sstk_rank * d_sstk) %>%
  mutate(s_txp_rank_ad = s_txp_rank * d_txp) %>%
  mutate(s_txt_rank_ad = s_txt_rank * d_txt) %>%
  mutate(s_xint_rank_ad = s_xint_rank * d_xint) %>%
  mutate(s_prcc_f_rank_ad = s_prcc_f_rank * d_prcc_f)
```

```
## save changes in form of additional columns
setwd(filepath_interpretation)
save(df_expl_norm_ranks_ad, file = "df_expl_norm_ranks_ad.RData")
save(df_expl_raw_ranks_ad, file = "df_expl_raw_ranks_ad.RData")
```

Calculate maximum ranks of related contributing features

Below, columns for maximum ranks of are added. For fraud categories with only one related feature, the value of this feature is adapted. For fraud categories with multiple related features the minimum value (i.e. the highest rank) is calculated.

```
## set names for maximum rank columns for each fraud category
max_ranks_lime <- paste0("w_max_rank_", fraud_categories)
max_ranks_shapley <- paste0("s_max_rank_", fraud_categories)
max_ranks_all <- c(max_ranks_lime, max_ranks_shapley)

## load data frames
load("df_expl_norm_ranks_ad.RData")
load("df_expl_raw_ranks_ad.RData")


## Add columns for final ranks
df_expl_norm_ranks_ad <- bind_cols(
  df_expl_norm_ranks_ad,
  setNames(data.frame(matrix(0,
                             nrow = nrow(df_expl_norm_ranks_ad),
                             ncol = length(max_ranks_all))),
           max_ranks_all))

df_expl_norm_ranks_ad <- df_expl_norm_ranks_ad %>%
  mutate(across(all_of(max_ranks_all), as.double))


df_expl_raw_ranks_ad <- bind_cols(
  df_expl_raw_ranks_ad,
  setNames(data.frame(matrix(0,
                             nrow = nrow(df_expl_raw_ranks_ad),
                             ncol = length(max_ranks_all))),
           max_ranks_all))

df_expl_raw_ranks_ad <- df_expl_raw_ranks_ad %>%
  mutate(across(all_of(max_ranks_all), as.double))


## select maximum feature ranks depending on fraud category (normalized data)

df_expl_norm_ranks_final <- df_expl_norm_ranks_ad %>%
  mutate(w_max_rank_rev = w_sale_rank_ad) %>%
  mutate(w_max_rank_rec = w_rect_rank_ad) %>%
  mutate(w_max_rank_cogs = w_cogs_rank_ad) %>%
  mutate(w_max_rank_inv = w_invt_rank_ad) %>%
  mutate(w_max_rank_reserve = w_ceq_rank_ad) %>%
  mutate(w_max_rank_debt = 0) %>%
  mutate(w_max_rank_mkt_sec = w_ivst_rank_ad) %>%
```

```
  mutate(w_max_rank_inc_exp_se = pmin(
    ifelse(w_ceq_rank_ad == 0, Inf, w_ceq_rank_ad),
    ifelse(w_csho_rank_ad == 0, Inf, w_csho_rank_ad),
    ifelse(w_dp_rank_ad == 0, Inf, w_dp_rank_ad),
    ifelse(w_ib_rank_ad == 0, Inf, w_ib_rank_ad),
    ifelse(w_ni_rank_ad == 0, Inf, w_ni_rank_ad),
    ifelse(w_pstk_rank_ad == 0, Inf, w_pstk_rank_ad),
    ifelse(w_re_rank_ad == 0, Inf, w_re_rank_ad),
    ifelse(w_sstk_rank_ad == 0, Inf, w_sstk_rank_ad),
    ifelse(w_txt_rank_ad == 0, Inf, w_txt_rank_ad),
    ifelse(w_xint_rank_ad == 0, Inf, w_xint_rank_ad),
    ifelse(w_prcc_f_rank_ad == 0, Inf, w_prcc_f_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(w_max_rank_asset = pmin(
    ifelse(w_act_rank_ad == 0, Inf, w_act_rank_ad),
    ifelse(w_at_rank_ad == 0, Inf, w_at_rank_ad),
    ifelse(w_che_rank_ad == 0, Inf, w_che_rank_ad),
    ifelse(w_ivao_rank_ad == 0, Inf, w_ivao_rank_ad),
    ifelse(w_ivst_rank_ad == 0, Inf, w_ivst_rank_ad),
    ifelse(w_ppegt_rank_ad == 0, Inf, w_ppegt_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(w_max_rank_pay = w_ap_rank_ad) %>%
  mutate(w_max_rank_liab = pmin(
    ifelse(w_dlc_rank_ad == 0, Inf, w_dlc_rank_ad),
    ifelse(w_dltis_rank_ad == 0, Inf, w_dltis_rank_ad),
    ifelse(w_dltt_rank_ad == 0, Inf, w_dltt_rank_ad),
    ifelse(w_lct_rank_ad == 0, Inf, w_lct_rank_ad),
    ifelse(w_lt_rank_ad == 0, Inf, w_lt_rank_ad),
    ifelse(w_txp_rank_ad == 0, Inf, w_txp_rank_ad),
    na.rm = TRUE)
    )

df_expl_norm_ranks_final <- df_expl_norm_ranks_final %>%
  mutate(s_max_rank_rev = s_sale_rank_ad) %>%
  mutate(s_max_rank_rec = s_rect_rank_ad) %>%
  mutate(s_max_rank_cogs = s_cogs_rank_ad) %>%
  mutate(s_max_rank_inv = s_invt_rank_ad) %>%
  mutate(s_max_rank_reserve = s_ceq_rank_ad) %>%
  mutate(s_max_rank_debt = 0) %>%
  mutate(s_max_rank_mkt_sec = s_ivst_rank_ad) %>%
  mutate(s_max_rank_inc_exp_se = pmin(
    ifelse(s_ceq_rank_ad == 0, Inf, s_ceq_rank_ad),
    ifelse(s_csho_rank_ad == 0, Inf, s_csho_rank_ad),
    ifelse(s_dp_rank_ad == 0, Inf, s_dp_rank_ad),
    ifelse(s_ib_rank_ad == 0, Inf, s_ib_rank_ad),
    ifelse(s_ni_rank_ad == 0, Inf, s_ni_rank_ad),
    ifelse(s_pstk_rank_ad == 0, Inf, s_pstk_rank_ad),
    ifelse(s_re_rank_ad == 0, Inf, s_re_rank_ad),
    ifelse(s_sstk_rank_ad == 0, Inf, s_sstk_rank_ad),
    ifelse(s_txt_rank_ad == 0, Inf, s_txt_rank_ad),
    ifelse(s_xint_rank_ad == 0, Inf, s_xint_rank_ad),
    ifelse(s_prcc_f_rank_ad == 0, Inf, s_prcc_f_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(s_max_rank_asset = pmin(
    ifelse(s_act_rank_ad == 0, Inf, s_act_rank_ad),
    ifelse(s_at_rank_ad == 0, Inf, s_at_rank_ad),
    ifelse(s_che_rank_ad == 0, Inf, s_che_rank_ad),
```

```
    ifelse(s_ivao_rank_ad == 0, Inf, s_ivao_rank_ad),
    ifelse(s_ivst_rank_ad == 0, Inf, s_ivst_rank_ad),
    ifelse(s_ppegt_rank_ad == 0, Inf, s_ppegt_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(s_max_rank_pay = s_ap_rank_ad) %>%
  mutate(s_max_rank_liab = pmin(
    ifelse(s_dlc_rank_ad == 0, Inf, s_dlc_rank_ad),
    ifelse(s_dltis_rank_ad == 0, Inf, s_dltis_rank_ad),
    ifelse(s_dltt_rank_ad == 0, Inf, s_dltt_rank_ad),
    ifelse(s_lct_rank_ad == 0, Inf, s_lct_rank_ad),
    ifelse(s_lt_rank_ad == 0, Inf, s_lt_rank_ad),
    ifelse(s_txp_rank_ad == 0, Inf, s_txp_rank_ad),
    na.rm = TRUE)
    )


## select maximum feature ranks depending on fraud category (non-normalized data)

df_expl_raw_ranks_final <- df_expl_raw_ranks_ad %>%
  mutate(w_max_rank_rev = w_sale_rank_ad) %>%
  mutate(w_max_rank_rec = w_rect_rank_ad) %>%
  mutate(w_max_rank_cogs = w_cogs_rank_ad) %>%
  mutate(w_max_rank_inv = w_invt_rank_ad) %>%
  mutate(w_max_rank_reserve = w_ceq_rank_ad) %>%
  mutate(w_max_rank_debt = 0) %>%
  mutate(w_max_rank_mkt_sec = w_ivst_rank_ad) %>%
  mutate(w_max_rank_inc_exp_se = pmin(
    ifelse(w_ceq_rank_ad == 0, Inf, w_ceq_rank_ad),
    ifelse(w_csho_rank_ad == 0, Inf, w_csho_rank_ad),
    ifelse(w_dp_rank_ad == 0, Inf, w_dp_rank_ad),
    ifelse(w_ib_rank_ad == 0, Inf, w_ib_rank_ad),
    ifelse(w_ni_rank_ad == 0, Inf, w_ni_rank_ad),
    ifelse(w_pstk_rank_ad == 0, Inf, w_pstk_rank_ad),
    ifelse(w_re_rank_ad == 0, Inf, w_re_rank_ad),
    ifelse(w_sstk_rank_ad == 0, Inf, w_sstk_rank_ad),
    ifelse(w_txt_rank_ad == 0, Inf, w_txt_rank_ad),
    ifelse(w_xint_rank_ad == 0, Inf, w_xint_rank_ad),
    ifelse(w_prcc_f_rank_ad == 0, Inf, w_prcc_f_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(w_max_rank_asset = pmin(
    ifelse(w_act_rank_ad == 0, Inf, w_act_rank_ad),
    ifelse(w_at_rank_ad == 0, Inf, w_at_rank_ad),
    ifelse(w_che_rank_ad == 0, Inf, w_che_rank_ad),
    ifelse(w_ivao_rank_ad == 0, Inf, w_ivao_rank_ad),
    ifelse(w_ivst_rank_ad == 0, Inf, w_ivst_rank_ad),
    ifelse(w_ppegt_rank_ad == 0, Inf, w_ppegt_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(w_max_rank_pay = w_ap_rank_ad) %>%
  mutate(w_max_rank_liab = pmin(
    ifelse(w_dlc_rank_ad == 0, Inf, w_dlc_rank_ad),
    ifelse(w_dltis_rank_ad == 0, Inf, w_dltis_rank_ad),
    ifelse(w_dltt_rank_ad == 0, Inf, w_dltt_rank_ad),
    ifelse(w_lct_rank_ad == 0, Inf, w_lct_rank_ad),
    ifelse(w_lt_rank_ad == 0, Inf, w_lt_rank_ad),
    ifelse(w_txp_rank_ad == 0, Inf, w_txp_rank_ad),
    na.rm = TRUE)
    )
```

```
df_expl_raw_ranks_final <- df_expl_raw_ranks_final %>%
  mutate(s_max_rank_rev = s_sale_rank_ad) %>%
  mutate(s_max_rank_rec = s_rect_rank_ad) %>%
  mutate(s_max_rank_cogs = s_cogs_rank_ad) %>%
  mutate(s_max_rank_inv = s_invt_rank_ad) %>%
  mutate(s_max_rank_reserve = s_ceq_rank_ad) %>%
  mutate(s_max_rank_debt = 0) %>%
  mutate(s_max_rank_mkt_sec = s_ivst_rank_ad) %>%
  mutate(s_max_rank_inc_exp_se = pmin(
    ifelse(s_ceq_rank_ad == 0, Inf, s_ceq_rank_ad),
    ifelse(s_csho_rank_ad == 0, Inf, s_csho_rank_ad),
    ifelse(s_dp_rank_ad == 0, Inf, s_dp_rank_ad),
    ifelse(s_ib_rank_ad == 0, Inf, s_ib_rank_ad),
    ifelse(s_ni_rank_ad == 0, Inf, s_ni_rank_ad),
    ifelse(s_pstk_rank_ad == 0, Inf, s_pstk_rank_ad),
    ifelse(s_re_rank_ad == 0, Inf, s_re_rank_ad),
    ifelse(s_sstk_rank_ad == 0, Inf, s_sstk_rank_ad),
    ifelse(s_txt_rank_ad == 0, Inf, s_txt_rank_ad),
    ifelse(s_xint_rank_ad == 0, Inf, s_xint_rank_ad),
    ifelse(s_prcc_f_rank_ad == 0, Inf, s_prcc_f_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(s_max_rank_asset = pmin(
    ifelse(s_act_rank_ad == 0, Inf, s_act_rank_ad),
    ifelse(s_at_rank_ad == 0, Inf, s_at_rank_ad),
    ifelse(s_che_rank_ad == 0, Inf, s_che_rank_ad),
    ifelse(s_ivao_rank_ad == 0, Inf, s_ivao_rank_ad),
    ifelse(s_ivst_rank_ad == 0, Inf, s_ivst_rank_ad),
    ifelse(s_ppegt_rank_ad == 0, Inf, s_ppegt_rank_ad),
    na.rm = TRUE)
    ) %>%
  mutate(s_max_rank_pay = s_ap_rank_ad) %>%
  mutate(s_max_rank_liab = pmin(
    ifelse(s_dlc_rank_ad == 0, Inf, s_dlc_rank_ad),
    ifelse(s_dltis_rank_ad == 0, Inf, s_dltis_rank_ad),
    ifelse(s_dltt_rank_ad == 0, Inf, s_dltt_rank_ad),
    ifelse(s_lct_rank_ad == 0, Inf, s_lct_rank_ad),
    ifelse(s_lt_rank_ad == 0, Inf, s_lt_rank_ad),
    ifelse(s_txp_rank_ad == 0, Inf, s_txp_rank_ad),
    na.rm = TRUE)
    )


## set all max_ranks which are not between 1 and 28 to NA
df_expl_norm_ranks_final <- df_expl_norm_ranks_final %>%
  mutate(across(
    .cols = max_ranks_all,
    .fns = ~ ifelse(. >= 1 & . <= 28, ., NA)
  ))

df_expl_raw_ranks_final <- df_expl_raw_ranks_final %>%
  mutate(across(
    .cols = max_ranks_all,
    .fns = ~ ifelse(. >= 1 & . <= 28, ., NA)
  ))


## save changes in form of additional columns
setwd(filepath_interpretation)
```

```
save(df_expl_norm_ranks_final, file = "df_expl_norm_ranks_final.RData")
save(df_expl_raw_ranks_final, file = "df_expl_raw_ranks_final.RData")
```

Visualization of explanations

Preprocessing of data frames for visualizations

Below, the data frames for the variants LIME vs Shapley Values and normalized vs. non-normalized are preprocessed. This includes transformations into a long data format.

```
## max ranks lime (normalized)
df_norm_lime_visualize <- df_expl_norm_ranks_final %>%
  filter(classification_result %in% c("tp", "fn")) %>%
  select(fyear, gvkey, classification_result,
         fraud_cat_count, any_of(max_ranks_lime))

## transform into long format
df_norm_lime_visualize_long <- df_norm_lime_visualize %>%
  pivot_longer(cols = starts_with("w_max_rank_"),
               names_to = "Category",
               values_to = "Rank") %>%
  drop_na()

## remove prefix of Category items
df_norm_lime_visualize_long <- df_norm_lime_visualize_long %>%
  mutate(Category = gsub("^w_max_rank_", "", Category)) %>%
  mutate(Category = as.factor(Category))

df_norm_lime_visualize_long$Category <- factor(
  df_norm_lime_visualize_long$Category,
  levels = c("rev", "rec", "cogs", "inv", "reserve", "mkt_sec", "pay",
             "asset", "liab",
             "inc_exp_se"))


## max ranks lime (non-normalized)
df_raw_lime_visualize <- df_expl_raw_ranks_final %>%
  filter(classification_result %in% c("tp", "fn")) %>%
  select(fyear, gvkey, classification_result,
         fraud_cat_count, any_of(max_ranks_lime))

## transform into long format
df_raw_lime_visualize_long <- df_raw_lime_visualize %>%
  pivot_longer(cols = starts_with("w_max_rank_"),
               names_to = "Category",
               values_to = "Rank") %>%
  drop_na()

## remove prefix of Category items
df_raw_lime_visualize_long <- df_raw_lime_visualize_long %>%
  mutate(Category = gsub("^w_max_rank_", "", Category))  %>%
  mutate(Category = as.factor(Category))

df_raw_lime_visualize_long$Category <- factor(
  df_raw_lime_visualize_long$Category,
```

```
  levels = c("rev", "rec", "cogs", "inv", "reserve", "mkt_sec", "pay",
             "asset", "liab",
             "inc_exp_se"))


## max ranks shapley values (normalized)
df_norm_shapley_visualize <- df_expl_norm_ranks_final %>%
  filter(classification_result %in% c("tp", "fn")) %>%
  select(fyear, gvkey, classification_result,
         fraud_cat_count, any_of(max_ranks_shapley))

## transform into long format
df_norm_shapley_visualize_long <- df_norm_shapley_visualize %>%
  pivot_longer(cols = starts_with("s_max_rank_"),
               names_to = "Category",
               values_to = "Rank") %>%
  drop_na()

## remove prefix of Category items
df_norm_shapley_visualize_long <- df_norm_shapley_visualize_long %>%
  mutate(Category = gsub("^s_max_rank_", "", Category)) %>%
  mutate(Category = as.factor(Category))

df_norm_shapley_visualize_long$Category <- factor(
  df_norm_shapley_visualize_long$Category,
  levels = c("rev", "rec", "cogs", "inv", "reserve", "mkt_sec", "pay",
             "asset", "liab",
             "inc_exp_se"))


## max ranks shapley values (non-normalized)
df_raw_shapley_visualize <- df_expl_raw_ranks_final %>%
  filter(classification_result %in% c("tp", "fn")) %>%
  select(fyear, gvkey, classification_result,
         fraud_cat_count, any_of(max_ranks_shapley))

## transform into long format
df_raw_shapley_visualize_long <- df_raw_shapley_visualize %>%
  pivot_longer(cols = starts_with("s_max_rank_"),
               names_to = "Category",
               values_to = "Rank") %>%
  drop_na()

## remove prefix of Category items
df_raw_shapley_visualize_long <- df_raw_shapley_visualize_long %>%
  mutate(Category = gsub("^s_max_rank_", "", Category)) %>%
  mutate(Category = as.factor(Category))

df_raw_shapley_visualize_long$Category <- factor(
  df_raw_shapley_visualize_long$Category,
  levels = c("rev", "rec", "cogs", "inv", "reserve", "mkt_sec", "pay",
             "asset", "liab",
             "inc_exp_se"))
```

Jitter Visualization

The code below was run four times manually adjusted for LIME vs Shapley Values and normalized vs. non-normalized variants. Here shown for LIME based on models trained on normalized data.

```
## create ggplot object
plot_expl_lime_norm <- ggplot(
  df_norm_lime_visualize_long,
  aes(x = Category, y = Rank, color = classification_result)) +
  geom_point(position = position_jitterdodge(
    jitter.width = 0.2,
    jitter.height = 0.05,
    dodge.width = 0.5),
    alpha = 0.3) +

  ## add vertical lines to separate between groups of fraud categories
  geom_segment(aes(x = 7.5, xend = 7.5, y = 28.5, yend = -1),
               linetype = "dashed", color = "darkgrey") +
  geom_segment(aes(x = 9.5, xend = 9.5, y = 28.5, yend = -1),
               linetype = "dashed", color = "darkgrey") +

  ## add horizontal lines stating the number of related features
  annotate("segment", x = 0.3, xend = 7.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 3.5, y = -2, label = "1 related\nfeature",
           color = "black", size = 2.5) +
  annotate("segment", x = 7.7, xend = 9.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 8.5, y = -2, label = "6 related\nfeatures",
           color = "black", size = 2.5) +
  annotate("segment", x = 9.7, xend = 10.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 10, y = -2, label = "11 related\nfeatures",
           color = "black", size = 2.5) +

  ## adjust axis
  scale_y_reverse(breaks = seq(28, 0, by = -4), limits = c(28.5, -2)) +

  ## adjust colors for "tp" and "fn"
  scale_color_manual(values = c("tp" = "green", "fn" = "red")) +

  ## settings for labels
  labs(
    title = "LIME, normalized",
    x = "Category",
    y = "Maximum rank of related feature"
  ) +
  theme_minimal() +
  theme(
    panel.grid = element_blank(),
    axis.line = element_line(color = "black"),
    plot.title = element_text(size = 12, hjust = 0.5),
    axis.title.x = element_text(margin = margin(t = -10), size = 10),
    axis.title.y = element_text(size = 10),
    axis.text.x = element_text(margin = margin(t = 13), angle = 45),
    legend.position = "bottom",
    legend.box.margin = margin(t = -10),
    legend.title = element_text(size = 10)
```

```
  )

print(plot_expl_lime_norm)
```

Combined visualization of LIME and Shapley Values

```
## save illustrations
ggsave(file="plot_expl_lime_norm.svg",
       plot = plot_expl_lime_norm,
       width = 5, height = 5)

ggsave(file="plot_expl_lime_raw.svg",
       plot = plot_expl_lime_norm,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_norm.svg",
       plot = plot_expl_shapley_norm,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_raw.svg",
       plot = plot_expl_shapley_raw,
       width = 5, height = 5)


## additional combined plot
plot_expl_lime <-
  plot_expl_lime_norm +
  plot_expl_lime_raw

plot_expl_shapley <-
  plot_expl_shapley_norm +
  plot_expl_shapley_raw

plot_expl <-
  plot_expl_lime /
  plot_expl_shapley


## save combined plot
ggsave(file="plot_expl.svg",
       plot = plot_expl,
       width = 10, height = 13)
```

Boxplot Visualization

The code below was run four times manually adjusted for LIME vs Shapley Values and normalized vs. non-normalized variants. Here shown for LIME based on models trained on normalized data.

```
## create ggplot object
plot_expl_lime_norm_box <- ggplot(
  df_norm_lime_visualize_long,
  aes(x = factor(Category), y = Rank, fill = classification_result)) +
  geom_boxplot(
    width = 0.4,
```

```
    position = position_dodge(width = 0.5),
    alpha = 0.8,
    outlier.size = 1) +

  ## add vertical lines to separate between groups of fraud categories
  geom_segment(aes(x = 7.5, xend = 7.5, y = 28.5, yend = -1),
               linetype = "dashed", color = "darkgrey") +
  geom_segment(aes(x = 9.5, xend = 9.5, y = 28.5, yend = -1),
               linetype = "dashed", color = "darkgrey") +

  ## add horizontal lines stating the number of related features
  annotate("segment", x = 0.3, xend = 7.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 3.5, y = -2, label = "1 related\nfeature",
           color = "black", size = 2.5) +
  annotate("segment", x = 7.7, xend = 9.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 8.5, y = -2, label = "6 related\nfeatures",
           color = "black", size = 2.5) +
  annotate("segment", x = 9.7, xend = 10.3, y = 0,
           yend = 0, color = "darkgrey") +
  annotate("text", x = 10, y = -2, label = "11 related\nfeatures",
           color = "black", size = 2.5) +

  ## adjust axis
  scale_y_reverse(breaks = seq(28, 0, by = -4), limits = c(28.5, -2)) +

  ## adjust colors for "tp" and "fn"
  scale_fill_manual(values = c("tp" = "green", "fn" = "red")) +

  ## settings for labels
  labs(
    title = "LIME, normalized",
    x = "Category",
    y = "Maximum rank of related feature"
  ) +
  theme_minimal() +
  theme(
    panel.grid = element_blank(),
    axis.line = element_line(color = "black"),
    plot.title = element_text(size = 12, hjust = 0.5),
    axis.title.x = element_text(margin = margin(t = -10), size = 10),
    axis.title.y = element_text(size = 10),
    axis.text.x = element_text(margin = margin(t = 13), angle = 45),
    legend.position = "bottom",
    legend.box.margin = margin(t = -10),
    legend.title = element_text(size = 10)
  )

print(plot_expl_lime_norm_box)
```

Combined visualization of LIME and Shapley Values boxplots

```
## save illustrations
ggsave(file="plot_expl_lime_norm_box.svg",
       plot = plot_expl_lime_norm_box,
       width = 5, height = 5)

ggsave(file="plot_expl_lime_raw_box.svg",
       plot = plot_expl_lime_norm_box,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_norm_box.svg",
       plot = plot_expl_shapley_norm_box,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_raw_box.svg",
       plot = plot_expl_shapley_raw_box,
       width = 5, height = 5)


## additional combined plot
plot_expl_lime_box <-
  plot_expl_lime_norm_box +
  plot_expl_lime_raw_box

plot_expl_shapley_box <-
  plot_expl_shapley_norm_box +
  plot_expl_shapley_raw_box

plot_expl_box <-
  plot_expl_lime_box /
  plot_expl_shapley_box


## save combined plot
ggsave(file="plot_expl_box.svg",
       plot = plot_expl_box,
       width = 10, height = 13)
```

Descriptive illustration of all explanations by feature

The code below illustrates the explanation ranks for all features and all explained observations.

Preprocessing of data frames

```
## preprocess dataframes into long format
df_expl_norm_ranks_final_long_lime <- df_expl_norm_ranks_final %>%
  select(fyear, gvkey, classification_result, fraud_cat_count, all_of(features_rank
_lime)) %>%
  pivot_longer(cols = all_of(features_rank_lime), names_to = "Variable", values_to
= "Rank") %>%
  filter(Rank >= 1 & Rank <= 28) %>%
  mutate(Variable = str_remove_all(Variable, "^w_|_rank$")) %>%
  mutate(classification_result = ifelse(classification_result %in% c("tp", "fn"), "
other", classification_result))
```

```
df_expl_raw_ranks_final_long_lime <- df_expl_raw_ranks_final %>%
  select(fyear, gvkey, classification_result, fraud_cat_count, all_of(features_rank
_lime)) %>%
  pivot_longer(cols = all_of(features_rank_lime), names_to = "Variable", values_to
= "Rank") %>%
  filter(Rank >= 1 & Rank <= 28) %>%
  mutate(Variable = str_remove_all(Variable, "^w_|_rank$")) %>%
  mutate(classification_result = ifelse(classification_result %in% c("tp", "fn"), "
other", classification_result))

df_expl_norm_ranks_final_long_shapley <- df_expl_norm_ranks_final %>%
  select(fyear, gvkey, classification_result, fraud_cat_count, all_of(features_rank
_shapley)) %>%
  pivot_longer(cols = all_of(features_rank_shapley), names_to = "Variable", values_
to = "Rank") %>%
  filter(Rank >= 1 & Rank <= 28) %>%
  mutate(Variable = str_remove_all(Variable, "^s_|_rank$")) %>%
  mutate(classification_result = ifelse(classification_result %in% c("tp", "fn"), "
other", classification_result))

df_expl_raw_ranks_final_long_shapley <- df_expl_raw_ranks_final %>%
  select(fyear, gvkey, classification_result, fraud_cat_count, all_of(features_rank
_shapley)) %>%
  pivot_longer(cols = all_of(features_rank_shapley), names_to = "Variable", values_
to = "Rank") %>%
  filter(Rank >= 1 & Rank <= 28) %>%
  mutate(Variable = str_remove_all(Variable, "^s_|_rank$")) %>%
  mutate(classification_result = ifelse(classification_result %in% c("tp", "fn"), "
other", classification_result))
```

Jitter plot for all features and their epxlanation ranks

Below, jitter plots for all explanations are created, highlighting false positives classifications in contrast to other classificiation results. This is again manually conducted four times for the different variants LIME vs Shapley Values and normalized vs. non-normalized data.

```
## create ggplot object
plot_expl_shapley_norm_feature_ranks <- ggplot(df_expl_norm_ranks_final_long_shaple
y,
       aes(x = Rank, y = fct_rev(factor(Variable)), color = classification_result))
+
  geom_point(position = position_jitterdodge(jitter.width = 0.3, jitter.height = 0.
3, dodge.width = 0.5), alpha = 0.2, size = 0.5) +

  ## adjust x- and y-axis
  scale_x_reverse(limits = c(29, 0), breaks = 28:1, labels = 28:1) +

  ## set color for classification_result
  scale_color_manual(
    values = c("fp" = "red", "other" = "blue"),
    name = "Classification Result"
  ) +

  ## set lables and title
  labs(
    title = "Shapley Values, normalized",
    x = "Rank of explanation",
```

```
    y = "Feature"
  ) +

  # style
  theme_minimal() +
  theme(
    plot.title = element_text(size = 12, hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    axis.title.x = element_text(size = 10),
    axis.title.y = element_text(size = 10),
    axis.line = element_line(color = "black"),
    panel.grid.minor = element_blank(),
    legend.position = "bottom",
    legend.box.margin = margin(t = -10),
    legend.title = element_text(size = 10)
  )
```

Combined visualization of feature ranks of LIME and Shapley Values

```
## save illustrations
ggsave(file="plot_expl_lime_norm_feature_ranks.svg",
       plot = plot_expl_lime_norm_feature_ranks,
       width = 5, height = 5)

ggsave(file="plot_expl_lime_raw_feature_ranks.svg",
       plot = plot_expl_lime_raw_feature_ranks,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_norm_feature_ranks.svg",
       plot = plot_expl_shapley_norm_feature_ranks,
       width = 5, height = 5)

ggsave(file="plot_expl_shapley_raw_feature_ranks.svg",
       plot = plot_expl_shapley_raw_feature_ranks,
       width = 5, height = 5)


## additional combined plot
plot_expl_lime_feature_ranks <-
  plot_expl_lime_norm_feature_ranks +
  plot_expl_lime_raw_feature_ranks

plot_expl_shapley_feature_ranks <-
  plot_expl_shapley_norm_feature_ranks +
  plot_expl_shapley_raw_feature_ranks

plot_expl_feature_ranks <-
  plot_expl_lime_feature_ranks /
  plot_expl_shapley_feature_ranks


## save combined plot
ggsave(file="plot_expl_feature_ranks.svg",
       plot = plot_expl_feature_ranks,
       width = 10, height = 13)
```

Appendix F: Compustat Data Items used as Features for Model Training

Abbreviation	Feature
act	Current Assets, Total
ap	Account Payable, Trade
at	Assets, Total
ceq	Common/Ordinary Equity, Total
che	Cash and Short-Term Investments
cogs	Cost of Goods Sold
csho	Common Shares Outstanding
dlc	Debt in Current Liabilities, Total
dltis	Long-Term Debt Issuance
dltt	Long-Term Debt, Total
dp	Depreciation and Amortization
ib	Income Before Extraordinary Items
invt	Inventories, Total
ivao	Investment and Advances, Other
ivst	Short-Term Investments, Total
lct	Current Liabilities, Total
lt	Liabilities, Total
ni	Net Income (Loss)
ppegt	Property, Plant and Equipment, Total
pstk	Preferred/Preference Stock (Capital), Total
re	Retained Earnings
rect	Receivables, Total
sale	Sales/Turnover (Net)
sstk	Sale of Common and Preferred Stock
txp	Income Taxes Payable
txt	Income Taxes, Total
xint	Interest and Related Expense, Total
prcc_f	Price Close, Annual, Fiscal

The table lists the 28 COMPUSTAT data items identified by Bao et al. (2020) and used for model training.

Appendix G: Categorization of Misstatement Types

Abbreviation	Account category affected
asset	Capitalized costs as asset
cogs	Misstated cost of goods sold
debt	Misstated allowance for bad debt
inc_exp_se	Misstatement of other expense/shareholder equity account
inv	Misstated inventory
liab	Misstated liabilities
mkt_sec	Misstated marketable securities
pay	Misstated payables
rec	Misstated accounts receivable
reserve	Misstated reserve account
rev	Misstated revenue

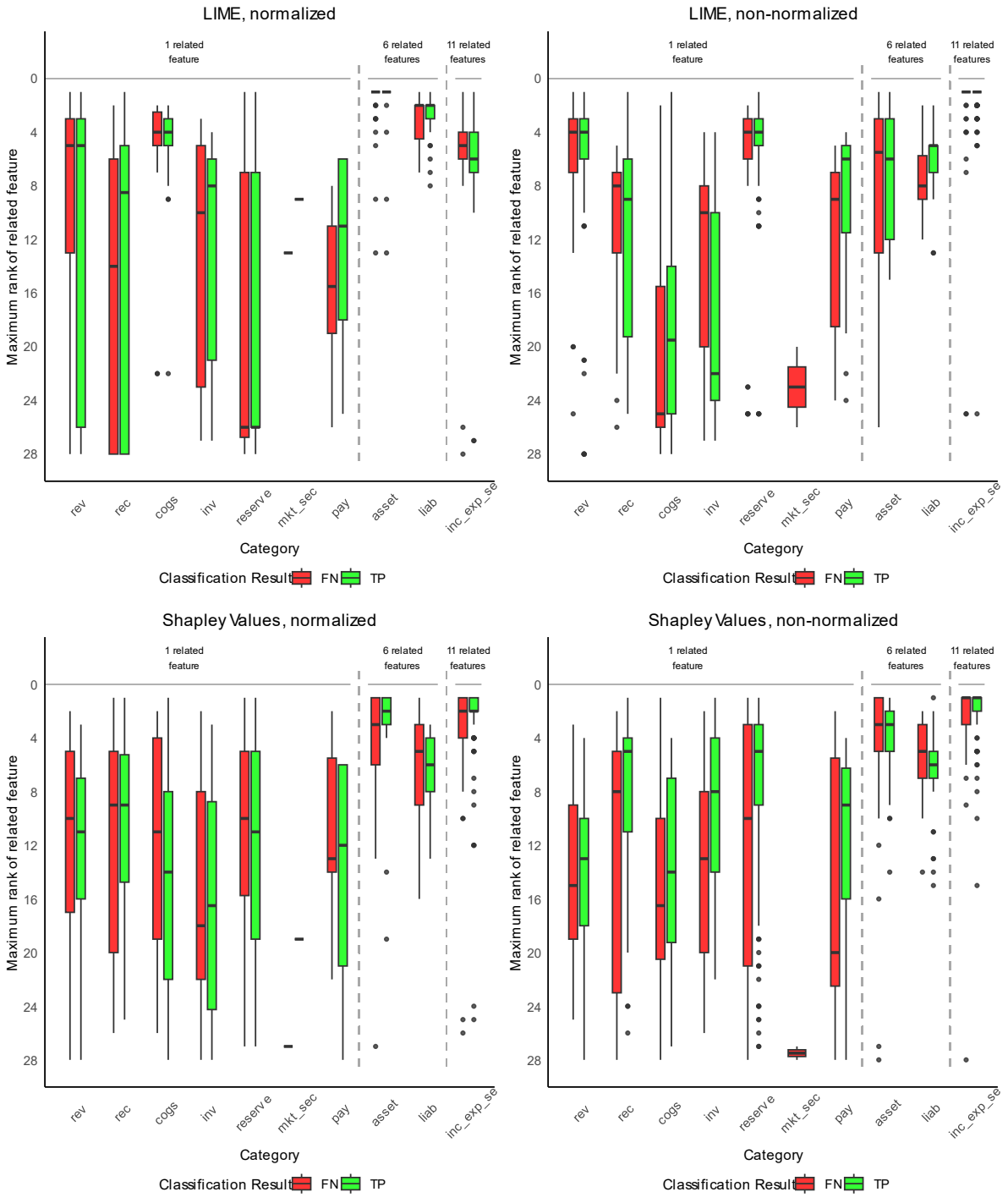
The table lists the types of misstatements and their abbreviations as categorized by Dechow et al. (2011) in their AAER database.

Appendix H: Detailed Classification Performances of RUSBoost Models

Features	Training Periods	Perform. Measure	Measures for each Test Period																
			2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Non-normalized	1990 – (testyear–3)	AUC	0.795	0.803	0.816	0.785	0.802	0.705	0.741	0.760	0.816	0.795	0.710	0.783	0.753	0.786	0.623	0.800	0.659
		Accuracy	75.3%	69.9%	72.0%	72.7%	73.9%	76.1%	81.1%	77.8%	78.0%	78.7%	78.6%	77.2%	79.0%	80.0%	76.1%	78.1%	80.5%
		Sensitivity	70.4%	77.4%	72.9%	71.0%	76.0%	42.1%	55.2%	64.0%	66.7%	48.0%	53.3%	64.7%	45.5%	41.7%	50.0%	75.0%	0.0%
		Precision	3.4%	2.7%	2.2%	1.4%	1.3%	0.6%	1.6%	1.4%	1.0%	1.0%	0.7%	0.9%	0.5%	0.5%	0.3%	0.3%	0.0%
		Specificity	75.4%	69.9%	72.0%	72.7%	73.9%	76.2%	81.3%	77.9%	78.0%	78.8%	78.7%	77.2%	79.1%	80.1%	76.2%	78.1%	80.5%
		FPR	24.6%	30.1%	28.0%	27.3%	26.1%	23.8%	18.7%	22.1%	22.0%	21.2%	21.3%	22.8%	20.9%	19.9%	23.8%	21.9%	19.5%
	rolling 10 years	AUC	0.803	0.780	0.804	0.788	0.732	0.669	0.722	0.790	0.814	0.807	0.752	0.807	0.730	0.711	0.749	0.714	0.594
		Accuracy	77.6%	74.5%	73.8%	74.7%	73.9%	82.1%	79.5%	76.4%	76.6%	82.1%	81.6%	78.9%	81.4%	78.8%	80.7%	79.1%	80.8%
		Sensitivity	70.4%	66.1%	75.0%	71.0%	60.0%	31.6%	34.5%	60.0%	66.7%	36.0%	53.3%	70.6%	36.4%	50.0%	66.7%	50.0%	0.0%
		Precision	3.8%	2.8%	2.4%	1.5%	1.0%	0.6%	0.9%	1.2%	1.0%	0.9%	0.8%	1.0%	0.4%	0.6%	0.4%	0.2%	0.0%
		Specificity	77.6%	74.6%	73.8%	74.7%	74.0%	82.3%	79.8%	76.5%	76.6%	82.3%	81.7%	79.0%	81.5%	78.9%	80.7%	79.2%	80.9%
		FPR	22.4%	25.4%	26.2%	25.3%	26.0%	17.7%	20.2%	23.5%	23.4%	17.7%	18.3%	21.0%	18.5%	21.1%	19.3%	20.8%	19.1%
Normalize	1990 – (testyear–3)	AUC	0.777	0.812	0.804	0.813	0.738	0.692	0.676	0.824	0.852	0.784	0.795	0.806	0.742	0.707	0.670	0.760	0.555
		Accuracy	83.5%	81.8%	81.2%	80.4%	81.6%	85.1%	84.4%	83.9%	84.8%	86.2%	88.8%	85.0%	88.8%	88.9%	87.1%	84.3%	88.3%
		Sensitivity	54.9%	61.3%	66.7%	83.9%	52.0%	21.1%	20.7%	76.0%	72.2%	32.0%	40.0%	52.9%	36.4%	33.3%	33.3%	25.0%	0.0%
		Precision	4.0%	3.6%	3.0%	2.3%	1.2%	0.5%	0.8%	2.2%	1.6%	1.1%	1.0%	1.1%	0.7%	0.7%	0.3%	0.1%	0.0%
		Specificity	83.8%	82.0%	81.4%	80.4%	81.8%	85.3%	84.8%	84.0%	84.8%	86.4%	89.0%	85.1%	88.9%	89.0%	87.1%	84.4%	88.3%
		FPR	16.2%	18.0%	18.6%	19.6%	18.2%	14.7%	15.2%	16.0%	15.2%	13.6%	11.0%	14.9%	11.1%	11.0%	12.9%	15.6%	11.7%
	rolling 10 years	AUC	0.792	0.814	0.799	0.763	0.720	0.671	0.752	0.824	0.838	0.814	0.787	0.800	0.803	0.808	0.724	0.746	0.672
		Accuracy	84.8%	84.7%	82.6%	82.2%	81.9%	86.7%	87.4%	85.4%	84.4%	87.2%	86.2%	85.4%	82.7%	86.4%	83.2%	83.3%	85.5%
		Sensitivity	54.9%	64.5%	58.3%	58.1%	44.0%	21.1%	44.8%	64.0%	61.1%	40.0%	33.3%	47.1%	45.5%	58.3%	50.0%	50.0%	0.0%
		Precision	4.4%	4.5%	2.8%	1.8%	1.1%	0.6%	2.0%	2.1%	1.3%	1.4%	0.7%	1.0%	0.6%	1.0%	0.4%	0.3%	0.0%
		Specificity	85.2%	85.0%	82.8%	82.3%	82.0%	87.0%	87.6%	85.5%	84.5%	87.4%	86.3%	85.6%	82.8%	86.5%	83.2%	83.4%	85.5%
		FPR	14.8%	15.0%	17.2%	17.7%	18.0%	13.0%	12.4%	14.5%	15.5%	12.6%	13.7%	14.4%	17.2%	13.5%	16.8%	16.6%	14.5%
		No of obs.	5809	5766	5693	5734	5704	5456	5215	5240	5244	5485	5504	5498	5252	4937	4804	4724	4546

The table lists the classification performances for the RUSBoost model. First, measures for the model trained on non-normalized data items are shown, and second, for a model trained on normalized data. Sections with training periods "1990 – (testyear–3)" cover e.g. for the test year 2003 the financial years from 1990 to 2000. The "rolling 10 years" covers for each test year 10 years of training but leaves out the 2 preceding periods before the test year. Performances are displayed by: 1) Area Under the Receiver Operating Curve (AUC); 2) Accuracy = (TP + TN)/(TP + TN + FP + FN); 3) Sensitivity = TP/(TP + FN); 4) Precision = TP/(TP + FP); 5) Specificity = TN/(FP + TN); 6) False Positive Rate (FPR) = FP/(FP + TN) Performance measures 2) to 5) are based on classifications with the previously determined cost efficient thresholds of 0.51 for the models based on raw financial items and 0.53 for normalized financial items. The table shows aggregated results for different time ranges.

Appendix I: Features' Maximum Ranks in Relation to Misstatement Types



Each panel illustrates boxplots for the highest rank of a feature's explanation which is related to the type of misstatement of the analyzed observation. As the first six categories are matched to a single feature. "asset" and "liability" in contrast are matched with 6 features and "inc_exp_se" with 11 features. The category "debt" is not included as there is no misstatement case for this type between 2003 and 2019 in the database.

Eidesstattliche Versicherung

Ich, Leonhard J. Lösse, versichere an Eides statt, dass die vorliegende Dissertation von mir selbstständig und ohne unzulässige fremde Hilfe unter Beachtung der ‚Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf' erstellt worden ist.

_____

Düsseldorf, 31. März 2025