Handelshøyskolen BI

GRA 19703 Master Thesis

Thesis Master of Science 100% - W

Predefinert informasjon

Startdato:	09-01-2023 09:00 CET	**Termin:**	202310
Sluttdato:	03-07-2023 12:00 CEST	**Vurderingsform:**	Norsk 6-trinns skala (A-F)
Eksamensform:	T		
Flowkode:	202310||11184||IN00||W||T		
Intern sensor:	(Anonymisert)		

Deltaker

Navn:	Liam Alexander Høgtun

Informasjon fra deltaker

Tittel *:	Automated Detection of Front Companies: Exploring Machine Learning Potentials and Limitations
Navn på veileder *:	Emil Aas Stoltenberg

Inneholder besvarelsen konfidensielt materiale?:	Nei	**Kan besvarelsen offentliggjøres?:**	Ja

Gruppe

Gruppenavn:	(Anonymisert)
Gruppenummer:	349
Andre medlemmer i gruppen:	Deltakeren har innlevert i en enkeltmannsgruppe

Automated Detection of Front Companies: Exploring Machine Learning Potentials and Limitations

Hand-in date:
03.07.2023

Campus:
BI Oslo

Examination code and name:
GRA1974 - Master Thesis

Programme:
MSc in Business Analytics

Acknowledgements

First, we would like to express our gratitude to our supervisor, Assistant Professor Emil Aas Stoltenberg, of the Department of Data Science at BI Norwegian Business School. His superb guidance and feedback have proven invaluable throughout our project's development.

Equally, we are grateful to Nigel Krishna Iyer, CEO and Founder of B4 Investigate AB, for his enlightening insights on front companies, fraud, and global beneficial ownership transparency. His willingness to collaborate and allow us to use some of B4 Investigate's data has significantly contributed to our work.

We would also like to acknowledge Christian Kalbakk-Bohler of B4 Investigate for his exemplary work in creating datasets grounded in big data and real cases.

Finally, our sincere thanks go to Dalia Breskuvien for sharing her expert knowledge in dealing with machine learning problems with high cardinality nominal variables in imbalanced datasets.

Abstract

This project is aimed at understanding the potential and limitations of modern machine learning algorithms in automatically detecting front companies. We employed predictive classification models to analyse a company's transaction history, with the goal of developing the most effective approach to solve the classification task.

The research was conducted in collaboration with B4 Investigate, a young company specializing in developing software for fraud detection and financial damage mitigation. The dataset used for analysis and model development was provided by B4 Investigate and was analysed using various Python libraries and intrinsic tools.

Front companies have a pervasive presence worldwide. While there might be valid justifications for utilizing a front, in most cases, they are often employed to conceal engagement in illegal or dubious activities. One significant hurdle is the arduous task of identifying such entities, as inadvertent association with a front company can lead to substantial financial or reputational harm.

Through our analysis, several important insights emerged. We discovered the significance of creating suitable representations of relevant variables, such as the country of registration for each company in the dataset as well as conveying the context of a company's regular business activities when employing the predictive solution. The implementation of these measures significantly improved the performance of the prediction task. Additionally, tree-based algorithms appear to be the most suitable for learning the correct indicative patterns in this specific prediction task. The findings suggest that modern machine learning algorithms indeed have the potential to serve as effective tools for automated detection of front companies, underscoring the value of further exploration in this field through future research. This approach holds promise and can provide tangible value to companies and non-profit organizations by accurately identifying any undisclosed associations with front companies, thus mitigating the risks of financial and reputational damage.

Table of Contents

List of Figures and Tables

Chapter 1, Introduction

1.1: The Aim of This Project

In this thesis, we aim to explore the process of selection, fine-tuning, and implementation of a binary machine learning classification system geared towards detecting front companies, often referred to as "shell companies" (the concept of front companies is defined in Chapter 1.2). Our methodology involves striving to develop an optimal approach for this classification task. Throughout this exploration, our primary objective is to enhance our understanding of the potential and limitations of contemporary machine learning algorithms in the automated detection of front companies.

Enhancing the capability to detect front companies has the potential to generate significant value in contemporary society by mitigating the risk of fraud, corruption and other kinds of financial, reputational and cultural damage caused by commercial interactions with these entities, both for individuals, companies and organisations and society as a whole.

Examples of the damage inflicted on legitimate companies by fronts include tarnishing their reputation, imposing unjustified fees, refusing to pay for services rendered, sudden disappearance of assets, and acquiring large sums of money only to declare bankruptcy, and vanish with the funds. These examples represent just a fraction of the potential ways in which front companies can cause harm (Iyer, 2019).

To gain a clearer understanding of this, Iyer (2019) offers an extensive description of "harm" in this context, specifically, harm related to fraud. Harm, or loss, can be perceived as the total cost—whether directly quantifiable or indirectly related—to the following:

1. Any action that results in the loss of money or assets.
2. Any occurrence that leads to reputational damage.
3. Any event that contributes to the erosion of organizational culture.
4. Anything that hinders the organization from functioning as intended.

Moreover, cooperating with such companies may indirectly support a wide range of highly unethical practices. These practices encompass money laundering,

terrorist financing, arms trafficking, tax fraud, corruption, bribery, circumventing sanctions, and more. Essentially, in any scenario where funds are required to facilitate illicit activities, front companies, often as part of a **"**Dirty Money Constellations" (or DMC) serve as indispensable instruments (Higgins, Morino, & Iyer, 2018).

Specifically, our objective is to develop a solution capable of performing binary classification on individual companies in a given dataset. Here, each company is labelled by an algorithm as either 0 (indicating it is not a front company) or 1 (indicating it is a front company), with the goal of minimizing misclassified instances.

This process involves several steps:

1. Determine the most suitable algorithm or algorithms for identifying front companies.
2. Once selected, we explored methods for transforming the data to create optimal representations that facilitate effective learning by the algorithm.
3. Finally, the algorithm is fine-tuned to make specific improvements when trained on the best-identified data representations, aiming to enhance its identification ability on the chosen performance metric.

By following this approach of algorithm selection, data transformation, and tuning, the goal is to develop a robust classification system with minimal misclassifications. This research could contribute to the identification and understanding of front companies, providing valuable insights into distinguishing covert entities within the corporate landscape.

1.2: What do we Mean by Front Companies

Also known as "shell companies," a very common role of a front company is to act as a cover for questionable activities. While these activities may not always be explicitly illegal, they are often dubious enough that the involved individuals (or companies) prefer not to have their association recognized. The common underlying motivation behind such activities is financial gain. The perpetrators seek a means to acquire the proceeds generated by their questionable actions without exposing their involvement. One of the keys to achieving this lies in obscuring beneficial ownership, a point made abundantly clear in the revelations

of the Panama Papers (International Consortium of Investigative Journalists, 2016).

In essence, a front company is typically formed with the objective of either circulating and whitewashing illicit funds or generating revenue through illegal or unethical means. Different money laundering techniques are often combined to further obscure the money trail (Financial Action Task Force, 2006). By masking their beneficial ownership, these entities provide individuals with the means to carry out such activities discreetly.

From the perspective of a legitimate company inadvertently doing business with a front, if the true nature of the counterparty's business is not uncovered, it can result in substantial financial damage or damage to the reputation of many legitimate companies. This can be attributed to the fact that front companies are often established and utilised by criminals and are deliberately structured to make it difficult or even impossible to trace the actual beneficiaries. As a result, they can easily capitalize on the first opportunity to take advantage of unsuspecting victims and disappear without a trace.

1.3: Establishing the Premises for the Data Used in This Project

To provide context, the final model developed in this study is intended to serve as either a **fully developed component or a baseline for a component that will be integrated into the software of B4 Investigate AB**, a company.

B4 Investigate has developed a software product that can perform a set of different assessment tasks of a company’s transactions history to flag various instances of possible fraud. One of those tasks is to identify if a given counterparty of company is likely to be a front company or not. The current method that is employed for this task is relatively simplistic since it relies solely on a basic rule-based approach based on their country of registration, and does so with varying degrees of accuracy. Moreover, it fails to consider the context of the source company or employ any form of pattern recognition.

The dataset used for this project is provided by B4 Investigate and is specifically tailored to the operational process of B4's product. The dataset consists of 153,000 unique financial transactions, which include invoices and credit notes. Each transaction contains a column indicating whether it belongs to a front company or

not. This information is crucial for training an algorithm to make accurate predictions regarding the status of other companies as fronts. Throughout the project the binary label for front company serves as the “target-label” that the algorithm seeks to predict.

It is important to note that the data used is synthetic and does not represent the transaction history of a real company. However, B4 has meticulously curated this dataset to closely resemble real-world scenarios. Therefore, although synthetic, this dataset is designed to be as accurate as possible in capturing the characteristics of fraud and front companies. Furthermore, it is worth noting that the data portrays financial transactions from the perspective of a single "source-company". Among all the companies included in the dataset, only 0.2% of them are labelled as front companies, indicating a significant class imbalance. In summary, despite evident limitations, this dataset should serve as a good proxy for a real-world situation, potentially leading to valuable insights when used as a baseline for analysis.

Chapter 2, Literature Review

2.1: Front Company And Money Laundering Literature

The United Nations Office on Drugs and Crime conducted a meta-analysis of drugs and crime, estimating that approximately 2.7% of the global GDP circulates through money laundering UNODC report (2011). Importantly, front companies, employed to obscure illicit activities, serve purposes beyond money laundering. They are also utilized for activities such as bribery, corruption, tax evasion, terrorist financing, and as primary facilitators of crimes, such as evading liabilities. This prevalence of malignant front companies underscores the value of developing advanced detection methods.

Even though it happened for centuries before, globalization has allowed individuals to establish companies and bank accounts worldwide. The rise of criminal activities and organizations has led to a consistent demand for corporate and banking secrecy. Political and economic incentives have simultaneously encouraged politicians and international banks to cater to this demand (Balakina, D'Andrea, & Masciandaro, 2017), thereby facilitating the creation of effective front companies. However, the assertion that transparency laws are established

solely for enabling immoral activities, driven by financial gains, is likely inaccurate as there are also ethical reasons for allowing ownership secrecy. Author and Fraud Investigation expert Nigel Iyer introduced the term "DMC" (Dirty Money Constellations), referring to interconnected front companies forming extensive networks (Iyer, 2019). These networks aim to obscure the beneficial ownership of illicit or immoral funds or assets further, utilizing transparency regulations in various jurisdictions. Identifying beneficiaries becomes challenging due to the intricate web of involved parent companies (Higgins, Morino, & Iyer, 2018) many of them front companies.

This observation highlights the high demand for front companies. It is highly likely that most companies are somehow connected to DMCs. Identifying entire DMCs requires comprehensive data, which is not feasible to obtain. Nevertheless, this thesis aims to demonstrate that identifying a single DMC component is possible with the right approach, potentially aiding future identification of entire DMCs.

Specific transparency regulations in various jurisdictions fundamentally allow fronts to thrive. Higgins, Morino, & Iyer (2018) argue that the term "tax haven" is outdated, as such locations are used for more than hiding from tax authorities. Establishing fronts in such locations allows beneficiaries to control, move, and utilize assets undetected. The term "offshore" is evolving with the changing landscape of transparency regulations and compliance rules, suggesting that simple rule-based decisions to flag potential front companies are often too simplistic, assuming linear relationships where more nuances need to be considered. This insight inspired extra attention in handling the country of registration in our analysis.

The Financial Action Task Force (2006) points out that the international trade system's evident loopholes allow criminals and terrorist financiers to "legitimize" their funds. The high volume of daily trade flows effectively conceals individual transactions, and when combined with complexities from multiple foreign exchange transactions and various trade financing arrangements, the comingling of legitimate and illicit funds becomes inevitable. The limited tools and resources available to customs agencies make it challenging to identify suspicious trade transactions.

The complexity of a good, in terms of its trade, makes it more challenging for customs agencies to identify suspicious pricing. This is partly because such entities lack the data and resources to determine many goods' fair market prices. Inadequate cooperation between countries restricts their ability to obtain a complete view of both sides of a transaction (Financial Action Task Force, 2006). Common money laundering methods include over- and under-invoicing goods and services, issuing multiple invoices for the same international trade, and phantom shipments, which entail invoicing non-existent goods while ensuring routine customs document processing (Financial Action Task Force, 2006). In all such instances, the use of front companies is advantageous for concealing identities, and financial institutions involved usually unknowingly facilitate illegal money laundering activities.

In other instances, stolen goods can be unloaded by selling them to legitimate companies at fair prices (Financial Action Task Force, 2006). This process merely requires establishing a front, creating a seemingly legitimate company website, and falsifying documents. Once the goods are sold, the money may be circulated through a network of other fronts, making it extremely difficult to identify the beneficiaries.

These works collectively highlight the scale and variety of illicit and immoral activities where front companies can effectively be used to conceal wrongdoing. The complexity of identifying such entities emphasizes the potential of machine learning approaches to detection, given their superior ability to capture nuances that current rule-based solutions overlook.

2.2: Analytics Literature

Fraudulent transaction detection is a well-researched area in machine learning. Credit card fraud, identity theft, account takeover, online banking fraud, and insurance fraud are among the areas that have been extensively studied in the context of machine learning. However, fraud is a broad term encompassing a wide range of activities. In contrast, front company detection is still in its early stages of machine learning research. This may be attributed to the relatively lower notoriety of front companies compared to the aforementioned fraud categories.

Nonetheless, as highlighted in a 2019 report by the FBI (D'Antuono), front company detection is gaining increasing attention as a potential threat.

We will discuss and review two notable papers that investigate a subject similar to front company detection, specifically the detection of money laundering activities. A significant portion of front companies is established with the intention of facilitating money laundering. The traditional approach to anti-money laundering (AML) employed in financial institutions relies on rule-based systems. However, these systems have notable limitations, such as their inability to effectively capture non-linearities and implement sufficiently complex rule structures and conditions. Additionally, despite their substantial benefits, rule-based systems are designed by humans and are thus susceptible to biases that hinder their ability to generalize well. As a result, there is a growing interest in replacing these systems with machine learning or other more dynamic approaches.

However, it's crucial to note that the applicability of machine learning to such problems comes with its own set of drawbacks. These include interpretability issues, the risk of overfitting, and the necessity for large, high-quality datasets for model training. Furthermore, successfully developed models, even those that have displayed substantial utility, require careful validation and periodic monitoring to ensure their continued high performance as external circumstances evolve.

Usman A, Naveed N, and Munawar S (2023) conducted research in a field that overlaps with the current topic of interest, focusing specifically on the financial domain. They explored the largely uncharted territory of using graph-based machine learning models, particularly Graph Convolutional Networks (GCNs), for detecting potential money laundering activities within transactions made by financial institutions.

Due to access restrictions to real-world financial data, their analysis relied on synthetically generated data designed to mimic bank transactions. The researchers created graphical representations of transactions between thousands of accounts and tested the efficacy of graph convolutional networks in identifying instances of money laundering. Their study revealed that the best-performing model achieved an F1-score of 0.69, showing promising potential.

However, it is critical to note the limitations of the study. Given that the data was synthetically generated, the generalizability of the results might be constrained to the context of banking. This study underscores that the field of using machine

learning for anti-money laundering purposes, including front company detection, is an emerging and rapidly evolving area where innovative approaches are continuously tested and adopted.

Rocha-Salazar J, Segovia-Vargas M, and Camacho-Miñano M (2022) propose a rule-based system approach for detecting money laundering activities within financial institutions. Their methodology employs a dynamic social network analysis of transactions between legal entities, utilizing both the characteristics of these entities and their interrelations to calculate risk. Notably, their dataset comprises real transactions, supplied by a Mexican financial institution.

The proposed methodology commences with expert risk assessments of each legal entity, informed by a predefined set of attributes. Following this, the system undertakes a network analysis, using mathematical calculations to measure the risk intensity of connections between entities. Key aspects of this process include temporal comparisons of intensity and the determination of a risk threshold, which discerns whether or not a relationship should be labelled as suspicious. This approach integrates human evaluation and automatic detection to form a comprehensive expert system. Importantly, their system reportedly surpasses the accuracy of traditional rule-based systems in detecting money laundering activities.

One notable limitation of this approach is the resource-intensive nature of the initial risk assessment for each legal entity. Nevertheless, the methodology provides an innovative and promising foundation for further development, particularly with the potential for reducing extensive human intervention. An intriguing proposition would be the incorporation of an AI language model, such as the Chat-GPT architecture, which could streamline the risk assessment process and significantly enhance the effectiveness of anti-money laundering measures.

The data and problem addressed in this thesis set it apart from the two previously described works. The significant distinction lies in the fact that the dataset does not assume the perspective of a financial institution but aims to be applicable to most companies. Unlike the other works that rely on data from an external viewpoint, encompassing transactions between multiple legal entities, this solution only necessitates transaction data specific to the company implementing the detection algorithm. The primary purpose is internal risk management rather than strict compliance requirements. Consequently, this thesis stands out in terms

of its intended utility and approach, rendering it unique in its field. This also addresses the limitations of scope that are seen in the other works.

Chapter 3, Data Structure

To uncover potential front companies that may be involved with the organization in focus, our analysis is based on five separate spreadsheets that contain concise data from the company's various financial transactions. These spreadsheets make up the aforementioned dataset, comprised of 153,000 unique transaction observations.

1. Spreadsheet containing customer invoices and credit notes.
2. Spreadsheet containing supplier invoices and credit notes.
3. Spreadsheet containing invoices and credit notes from other business counterparts, referred to as loose-money.
4. Spreadsheet containing master data for customers.
5. Spreadsheet containing master data for suppliers.

The spreadsheets are merged and grouped by company name using Pandas version 2.0.2 (Pandas Development Team, 2023). This process compresses all transactions belonging to a given company into one row, and summary statistics are calculated for all relevant transactions. Each company is assigned a binary column indicating whether it is a front company or not. This means that all transactions, regardless of whether they appear in the customer, supplier, or loose-money spreadsheet, will be attributed to the correct company. A company may appear across multiple spreadsheets.

Once the spreadsheets are consolidated, grouped by company name, and cleaned, a data-frame is created to contain all the useful information in the dataset on which the analysis is based. The data-frame consists of the following columns:

Company name	**Company name specified in transaction.**
Company country code	2-letter country code corresponding to country of registration.
Currencies used by company	List of 3-letter currency codes representing each unique currency in which the company has conducted transactions.

Mean transition amount	Average value of all transactions conducted by the company that exist in the dataset.
Median transaction amount	Median value of all transactions conducted by the company that exist in the dataset.
Minimum transaction amount	Minimum value of all transactions conducted by the company that exist in the dataset.
Maximum transaction amount	Maximum value of all transactions conducted by the company that exist in the dataset.
Total sum of transaction amounts	Summed value of all transactions conducted by the company that exist in the dataset.
Standard deviation of transaction amount	The standard deviation of all transactions conducted by the company that exist in the dataset.
Number of transactions with round values	The number of whole-numbered transactions conducted by the company that exist in the dataset.
Number of transactions that are invoices	The number of positive-numbered transactions conducted by the company that exist in the dataset.
Number of transactions that are credit notes	The number of negative-numbered transactions conducted by the company that exist in the dataset.
Binary value indicating whether the company has appeared as a customer	1 if the company has ever conducted a transaction in the customer data.
Binary value indicating whether the company has appeared as a supplier	1 if the company has ever conducted a transaction in the supplier data.
Binary value indicating whether the company has appeared as a loose counterpart	1 if the company has ever conducted a transaction in the loose-money data.

Binary value indicating whether the company is a front. This is the target label	1 if the company is labelled as a front company. This value is 1 for all transactions belonging to the company.

Table 1: Columns in original dataset

The figures below provide a general impression of how values are spread across the observations in the dataset.

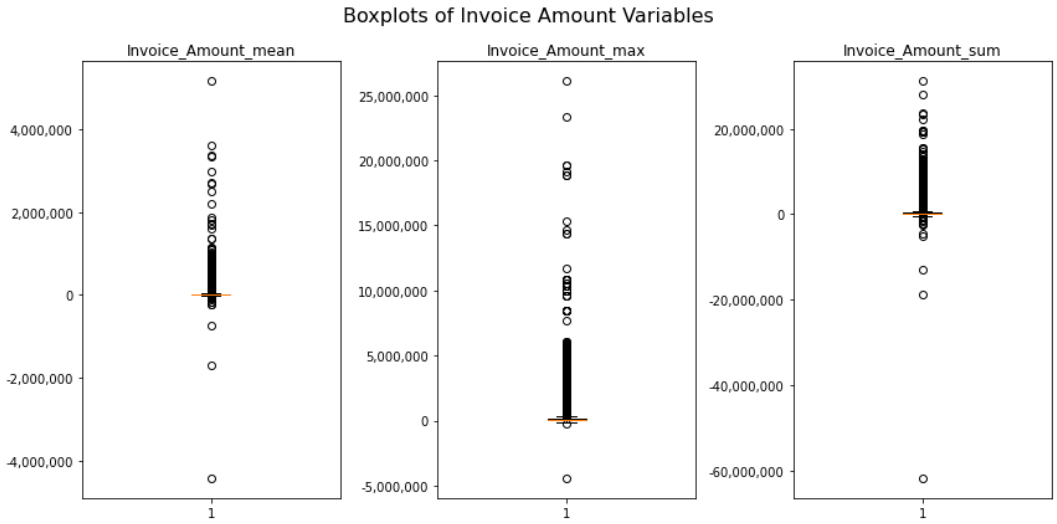

Figure 1: Invoice boxplots mean, max, sum

Consolidation of the transactions by using summary statistics is practical in light of the business problem, the objective is to identify front companies (not individual front-transactions). Justification for the choice of the variables is presented in section 5.1.

Chapter 4, Testing and Development

Through multiple stages of development and testing, we successfully devised a solution that significantly enhanced the front company identification ability of a machine learning classification algorithm beyond the baseline expected performance. Specifically, we achieved an increase in the F1-score from 0.02 to 0.48. This section provides the necessary context regarding the measurement and significance of the F1-score, details how the baseline expected performance was determined, and presents an overview of the procedure that led to the notable improvement in performance.

4.1: Performance Metric in Focus (F1-Score)

In this project, each unique combination of algorithm type, feature composition, and algorithm hyperparameter is evaluated based on the average F1-score it achieves in a five-fold cross-validation of the dataset.

To better understand the rationale behind the use of the F1-score, consider the specifics of the prediction task and the relevant terminology. In the dataset, each observation is assigned a binary label (0/1), indicating whether or not it is a front company. The algorithm's goal is not only to correctly predict 1 for all companies labelled as 1 but also to accurately predict 0 for all companies labelled as 0. The following classification terminology aids in defining this assessment:

- ***True positive:*** Predicts 1 for a company labelled as 1.
- ***True negative:*** Predicts 0 for a company labelled as 0.
- ***False positive:*** Predicts 1 for a company labelled as 0.
- ***False negative:*** Predicts 0 for a company labelled as 1.

- ***Recall:*** Measures the ability to identify all positive instances from a dataset. It represents the fraction of positive instances in the dataset that the algorithm correctly classifies as positive. Recall disregards the fraction of true negatives. The metric is calculated as:

  $$Equation\ 1) \qquad \textbf{Recall} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}}$$

  In simpler terms, recall quantifies the algorithm's capability to correctly identify positive instances, without taking into account the number of true negatives. It is expressed as the ratio of true positives to the sum of true positives and false negatives.

- ***Precision:*** Like recall, precision is also concerned with the model's ability to identify positive instances. However, the two metrics differ in their handling of misclassifications. Precision takes into account the instances where a negative instance is incorrectly identified as positive, also known as false positives. It is computed as:

  $$Equation\ 2) \qquad \textbf{Precision} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Positives}}$$

In simpler terms, precision assesses the proportion of instances that the model correctly classified as positive out of all instances it classified as positive. Therefore, precision does not account for false negatives.

- ***Harmonic Mean:*** The harmonic mean is used to calculate the average of a set of values in a way that achieves a balanced result. To calculate the harmonic mean of two numbers, denoted as A and B, the following formula is used:

$$\textit{Equation 3)} \qquad \textbf{Harmonic Mean} = 2 * \left(\frac{\mathbf{A} * \mathbf{B}}{\mathbf{A} + \mathbf{B}}\right)$$

In essence, the harmonic mean formula balances the influence of both numbers A and B to obtain an average value that is representative of the overall dataset.

The primary objective of this project is to maximize true positives while minimizing false positives. Achieving this balance is crucial since misclassifying front companies can result in significant financial or reputational damage. Therefore, recall, which captures the ability to identify true positives, plays a vital role.

At the same time, it is crucial to minimize the generation of false positives in an automated solution, as this would necessitate manual investigation, thereby undermining the primary purpose of automation. Thus, striking a balance between recall and minimizing false positives is of utmost importance.

The F1-score, representing the harmonic mean of precision and recall, emerges as the most suitable performance metric for this project. It considers the relative proportion of false negatives and false positives, rather than solely focusing on the total numbers. This makes it particularly valuable when dealing with imbalanced datasets like the one used in this project, where positive observations are rare (0.2% of the dataset).

For instance, in this dataset, there are approximately 6000 observations, with only 13 being positive observations. If the algorithm were to predict all observations as negative, it would have classified 99.8% of the observations correctly, but it would miss all 13 positive observations. Since these 13 observations make up 100% of the positive class, the F1-score would be 0, effectively capturing the

important nuances such as the 100% misclassification of positive instances in an otherwise 99% accurate prediction performance.

By considering the purpose and utility of the model in a real-world scenario, the F1-score ensures effective risk mitigation by accurately identifying true positives while minimizing false positives. It serves as the central evaluation metric referenced throughout this thesis, reflecting the underlying business problem to maximizing the value of a potential deployed model.

In this project, the F1-score was calculated using scikit-learn's metric module (scikit-learn Development Team, 2021).

4.2: Chronological Development Structure (Step 1-3)

The project followed a structured approach to develop an effective front company detection solution. Each step and the decisions described within them were solely based on the objective of identifying the optimal combination of algorithm type, feature representation, and hyperparameter specification that yields the highest F1-score for predicting whether a given company in the dataset is labelled as 1 or 0.

This development structure was selected taking into account the nature of the components. The initial and most logical step was to pinpoint the most suitable algorithm type. This strategy aimed to prevent spending unnecessary time analysing algorithms that are unsuitable for learning from this type of data in the given classification problem. Since the hyperparameters depend on specific data and algorithms, it was most reasonable to determine the best feature representation before proceeding with hyperparameter tuning, hence leaving the tuning as the final step.

Brief descriptions of the three stages are provided below.

Step 1 encompassed evaluating the F1-score of various prevalent machine learning classifiers, each trained using three distinct sampling strategies. The goal during this phase was to pinpoint the most suitable algorithm or algorithms, as well as determine the sampling strategies that synergize effectively with these selected algorithms and merit further consideration**. The Decision Tree classifier, Gradient Boost classifier, and XGB Classifier emerged as the optimal algorithms.** Consequently, the remaining algorithms were eliminated from the rest of the

project. The results obtained from training on both the original dataset and the SMOTE dataset warranted their inclusion in subsequent analyses.

Step 2. In this step, various measures were tested in feature engineering and encoding to evaluate their impact on performance. It was found that employing an **unsupervised neural network known as auto-encoder** to compress the dimensionality of the country code variable, along with incorporating new "tailored" features related to geo-spatial clusters and currency context in the dataset, effectively assisted the algorithm in learning valuable patterns within the data.

Step 3. **Hyperparameter tuning** was conducted based on the two previously mentioned algorithms and the optimal feature representations to assess their impact on performance. Despite investigating a wide array of potential hyperparameter combinations, none were identified that could enhance performance beyond that which was achieved using the default parameters provided by scikit-learn version 1.2.0 (scikit-learn Development Team, 2021).

For further details:

- Extended result summaries of steps 1-3 are provided in section 4.3.
- Descriptions of the initial motivations, as well as the specific procedures of steps 2 and 3, are provided in Section 4.

4.3: Development Descriptions and Results

4.3.1: Step 1, Algorithm and Sampling Method

All algorithms used were implemented using scikit-learn version 1.2.0 and retained their default hyperparameters during this stage.

The classification algorithm types tested include:

1. Logistic Regression.
2. Support Vector Machine.
3. Naives Bayes.
4. Decision Tree.
5. Gradient Boosting.
6. Extreme Gradient Boosting (XGB).

The sampling strategies tested include:

1. Original dataset.

2. Synthetic up-sampling of minority class.
3. Synthetic down-sampling of majority class.

All combinations of algorithm type and sampling method were tested and evaluated. Resulting F1-score for each combination are given in the table below:

Algorithms	Original data	Synthetically up-sampled minority class	Synthetically down-sampled majority class
SVM	0.0	0.03	0.03
Logistic Regression	0.08	0.003	0.05
Naives Bayes	0.06	0.05	0.02
Decision Tree	0.0	0.10	0.01
Gradient Boosting	0.0	0.19	0.01
XGB	0.0	0.24	0.01

Table 2: F1-scores of classifiers trained on different sampling strategies

Furthermore, the figure below presents the F1 scores of each model, each trained using different sampling methods. A red horizontal line indicates the average performance of all models when trained on the original dataset. Note that the lines connecting the dots are purely for illustrative purposes and do not impart any additional information.

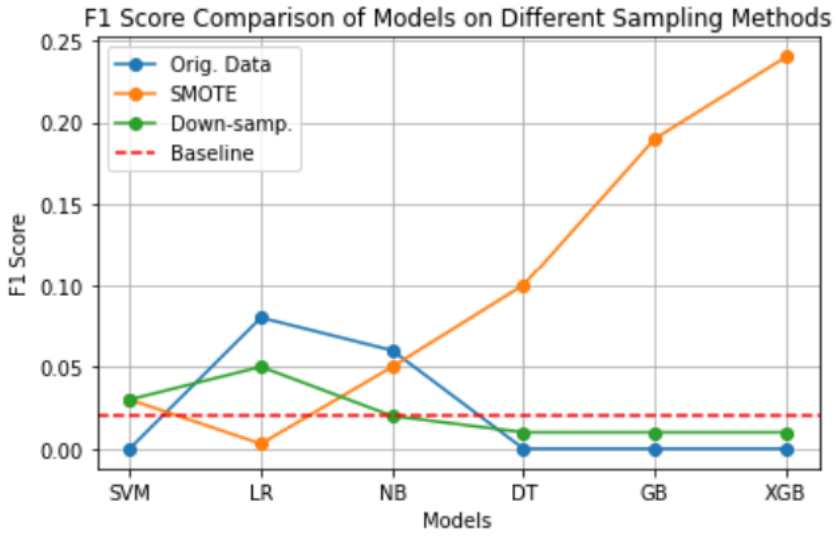

Figure 2: Visualized F1-scores of classifiers trained on different sampling strategies

Summary of algorithm and sampling method

Upon training on the dataset without implementing any resampling strategies, Logistic Regression proved to be the most effective classifier. However, all models demonstrated exceptionally poor performance, with an F1-score of less than 0.1. This observation held true even when the models were trained on a dataset where the majority class was down-sampled to match the minority class.

Conversely, when SMOTE was used to synthetically up-sample the data, tree-based algorithms emerged as the superior classifier algorithms. The technique known as SMOTE is frequently employed to mitigate the problem of class

imbalance. It does so by over-sampling the minority class - in this case, positive instances of front companies - through the generation of synthetic observations, rather than merely duplicating existing instances. This methodology fosters the creation of larger and more generalized decision regions, which, as per Chawla, Bowyer, Hall, & Kegelmeyer (2002), has demonstrated improved learning outcomes in decision trees.

Each tree-based algorithm achieved an F1-score of more than 0.1, with XGB notably reaching an F1-score of 0.24. For the Decision Tree, Gradient Boosting, and XGB algorithms, the highest achieved scores from this initial test serve as baseline performances and were used for comparison purposes throughout the project. Although they represented the best performance resulting from a specific intervention, namely SMOTE, the scores serve as a useful baseline that we aimed to improve upon using methods that were not as immediately apparent as up-sampling the minority class in this highly imbalanced dataset. The actual baseline expected performance of a machine learning algorithm for this problem may have been the average F1-score of all tested models when trained on the original dataset, which was found to be 0.02.

The stark contrast between the poor performance of all models, when trained on the original dataset, and the more promising results achieved with the SMOTE data, suggests that the dataset's imbalance (0.2% minority class) makes using the original dataset a challenging approach. **Consequently, subsequent steps also incorporated tree-based algorithms, trained on synthetically up-sampled data**. Training the algorithms on the down-sampled data proved to be ineffective across all models and was consequently excluded from further testing and analysis.

SMOTE was implemented using **imbalanced-learn library version 0.10.1** (Lemaître, Nogueira, & Aridas, 2017).

4.3.2: Step 2, Feature Engineering and Encoding

In step 2, we do further refinement. In Step 1, we opted to employ the Decision Tree Classifier, Gradient Boosting Classifier, and XGB Classifier, trained on both the original data and the data oversampled by SMOTE. Throughout this step, we scrutinized the effects of various feature engineering and encoding strategies on the performance of these algorithms and sampling methodologies. In the field of

machine learning, the term 'feature' often refers to an independent variable within the dataset, which is used to generate predictions. Feature engineering, on the other hand, is the process of manipulating existing data values to create new features. This process can potentially enhance the algorithm's ability to discern patterns.The testing and assessment of these strategies was done in chronological order, with the documented enhancements at each stage serving as a benchmark for subsequent evaluations. (Zhou, 2015)

1. Training an auto-encoder (an unsupervised algorithm) to compress the dimensionality of the country code. Different compression dimensions ranging from 1 to 10 were tested for each combination of algorithm and sampling strategy. It was found to be an effective approach that aided the models in learning useful patterns in the data. Best achieved F1-scores when including auto encoded representations of country codes best performance achieved by the algorithms were Decision (F1-score = 0.29, up from 0.1) from, Gradient Boost (F1-score = 0.38, up from 0.19), and XGB (0.34, up from 0.24). Specifics of approach described in section 4.2.

2. Using a target-based James Stein encoder, to compress the dimensionality of country codes was found to have a very slight positive effect on performance. However, this effect was noticeably lower compared to compressing country codes using an autoencoder. The specifics of this approach are described in section 4.3.

3. Constructing a new feature based on clustering geographic regions of country codes that appeared in the dataset. The clusters were identified using DBSCAN (Density-Based Spatial Clustering of Applications with Noise). A single "main cluster" was identified to represent the expected area of trade for the source company. For each observation, a feature indicating the distance metric between their country code and the main cluster was added. This improvement slightly enhanced the performance compared to the best baseline for Gradient Boosting (F1-score = 0.23, up from 0.19) but did not yield similar results for XGB and Decision Tree. The specifics of this approach are described in section 4.4.1.

4. A new binary feature was constructed based on the most commonly observed currency in the dataset, as well as the corresponding country codes. These new variables indicate whether a given company used a currency that did not match their country of registration, matched the most common currency observed in the dataset, or applied to both cases. Including this information had positive effects on

performance for Gradient Boosting (F1-score = 0.27, up from 0.19) and XGB (F1-score = 0.40, up from 0.24). However, it showed detrimental effects on performance in the case of Decision Tree. The specifics of this approach are described in section **5.4.2.**

5. This step involved combining the autoencoding approach from point 1 with the features described in points 3 and 4, with the aim of creating the most comprehensive representation of the data explored so far. The combined approach proved to be effective, particularly for the Gradient Boosting algorithm, achieving an F1-score of 0.48, which was the highest performance observed thus far.

Summary of feature engineering and encoding

The implementation of the following alterations to the dataset resulted in a substantial improvement in prediction performance. The average F1-score on the original data was 0.02, but with the following specifications, the F1-score reached 0.48, indicating a significant enhancement in performance.

1. Gradient Boosting Classifier using scikit-learn version 1.2.0 ensemble module default hyperparameters (scikit-learn Development Team, 2021).

2. Compressing country codes from their original 40 dimensions into 9-dimensional representation constructed through an auto encoder.

3. Retaining one-hot-encoded binary values for currencies, with each unique currency represented by a separate column.

4. Including the geographic clustering feature described in point 3.

5. Including the currency related features described in point 4.

6. Synthetically up sample minority class using SMOTE.

7. The rest of the dataset remained unchanged.

In summary, step 2 led to a significant improvement in the F1-score.

4.3.3: Step 3, Hyperparameter Tuning

Further tuning was tested in step 3. Grid search and random search were examined to determine if the hyperparameters of Gradient Boosting and XGB could be tuned for better predictions based on the optimal feature representation described in previous steps. However, there were inherent limitations to this. The primary issue was that training an autoencoder is an incredibly time-intensive process, and it must

be redone for each test. Given the constraints of the available processing hardware, conducting a thorough grid search or random search on the best feature representation wasn't feasible, as this could potentially take several hours, if not days, to complete. Therefore, this process was only conducted on the optimal representation without compressing country codes. Nevertheless, it was found that this made only marginal improvements to the models, and performance declined when applying the identified hyperparameters in conjunction with auto-encoded country code representations. This could be attributed to the fact that the default hyperparameters used in scikit-learn version 1.2.0 are based on expert heuristics, making them well-suited to many problems.

In summary, the main benefits of the development process were achieved in step 1 and 2, while further hyperparameter tuning in step 3 was found to have no positive effect. Better hardware capabilities may have led to better results from hyperparameter-tuning.

Chapter 5, Specifics of the Development Process

This section goes more into detail about the tools and models used, both from a theoretical and practical perspective.

Numerical computations and array operations were performed using NumPy version 1.23.5 (NumPy Contributors, 2023). We generated the plots using Matplotlib version 3.7.1 (Hunter, 2007) and Seaborn version 0.12.2 (Waskom, 2021).

5.1: Theoretical Motivation Behind Encoding for Dimensionality Reduction

After assigning a binary column to each unique country code and currency observed in the data, the dataset contains 73 different columns. This data representation enables the training of an algorithm to predict which observations are front companies. However, the dataset is quite high-dimensional. If this solution is to be refined by training on new and broader datasets to improve generalization ability, a substantially larger variety of country codes and currencies may need to be included, potentially increasing the dimensionality to the hundreds.

When a dataset contains a large number of features, the complexity that the model needs to handle increases significantly. Adding just a few additional features can result in an exponential growth in the possible number of unique row combinations. Consequently, it becomes more challenging for an algorithm to effectively generalize. As the dataset becomes sparser, it becomes increasingly difficult to capture the true underlying patterns. Compounding the issue is the extremely imbalanced nature of the dataset, with the minority class accounting for less than 0.5%.

During an interview I conducted with Dalia Breskuvien, a data scientist and author of a peer-reviewed paper addressing encoding approaches in training machine learning algorithms for prediction tasks on imbalanced datasets (Breskuvien˙e & Dzemyda, 2023), she stated that utilizing binary encoding for high cardinality variables like country codes and currencies is a suboptimal solution that often results in poor performance. The choice of encoding technique for such variables can significantly influence the prediction performance (Breskuvien˙e & Dzemyda, 2023).

These factors incentivized us to further explore encoding methods to reduce the dimensionality of the representation of both country codes and currencies in the dataset. If done correctly, it improves the algorithm's ability to learn useful patterns that aid in accurately identifying front companies within the dataset. The choice was made to only focus on country codes and to not use dimensionality reduction for currencies, as any given entity may have multiple currencies attributed to it. The purpose of using encoding is to attribute an alternative representation to each unique value that may appear within the variable. Since there may be multiple values for a single observation, this would entail combining the representations, which could result in a value that is not useful for an algorithm. Feasible solutions exist for this issue, but they will not be explored in this thesis.

5.2: Auto-Encoder

During the interview, Dalia Breskuvien suggested that an auto-encoder could be an effective method for reducing the dimensionality of nominal variables, such as country codes. The architecture of autoencoders is premised on an encoder-

decoder approach. The encoder processes the input to generate a code vector. This code vector is then processed by the decoder to reconstruct the input. Both components are trained in tandem, with the objective being that the reconstructed data should closely resemble the original input. In this manner, the most significant characteristics are learned (Ranzato, Poultney, Chopra, and LeCun, 2007).

The autoencoder model was implemented using the Keras API from the TensorFlow library version 2.12.0 (TensorFlow, 2023)

Chronological description of how auto-encoder was developed and implemented in this project:

1. Create binary columns for each unique country code observed in the dataset. For each observation, the value will be 1 for its corresponding country code and 0 for the rest.

2. Create an isolated data frame consisting solely of the binary column for country codes. This will serve as the baseline data referred to in the remaining steps.

3. The autoencoder comprises several components: an input layer, an encoder layer, a dropout layer, and a decoder layer. The input layer is designed to match the size of the unique country codes and serves to feed inputs into the encoder layer. The purpose of the encoder is to extract valuable information from the data while reducing its dimensionality. In this case, it reduces the dimensions to a pre-specified number, set at 2. To capture complex patterns in the data, the encoder employs a non-linear activation function called Rectified Linear Unit (ReLU) to perform transformations on the input features.

4. The dropout layer plays a crucial role in preventing overfitting. It randomly sets a fraction of the inputs to 0 during training, discouraging over-reliance on specific weights and fostering a more robust learning process. Subsequently, the decoder attempts to reconstruct the compressed data into its original dimensions. The decoder uses a sigmoid activation function, which is particularly suited for binary data. The accuracy of this reconstruction process serves as a reference when the autoencoder adjusts its parameters during training.

5. Once the architecture is finalized, the autoencoder is compiled with the Adam optimizer and the binary cross-entropy loss function. The Adam optimizer is a

well-regarded and frequently used optimizer that has demonstrated to work well in practice and also performs favourably in comparison to other stochastic optimization methods (Kingma & Ba, 2015).

6. Based on the trained autoencoder, a separate "encoder" model is defined. It shares the exact same input and encoder layers as our autoencoder. This model is then used to generate compressed representations of the country codes in our data, which will subsequently replace the country code column in the dataset.

7. The result is that each observation is now represented by only two columns, instead of the 40 binary columns that previously represented each unique country code.

The figure below illustrates the results of training the algorithms using varying dimensional representations of country codes, compared with the pre-established baseline F1-scores. It's important to note that these outcomes were obtained after the dataset was initially augmented using SMOTE.

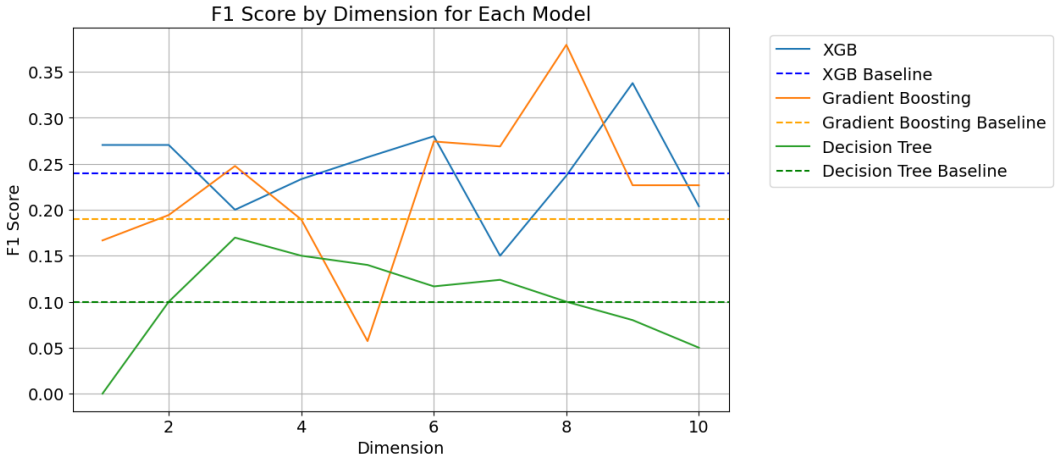

Figure 3: F1-scores using auto-encoded compressed representations of country codes in dataset (dimensions 1-10), SMOTE version

*Best Decision Tree F1 score = **0.17** (3-dimensions).*

*Best Gradient Boosting F1 score = **0.38** (8-dimensions).*

*Best XGB F1 score = **0.34** (9-dimensions).*

The figure below illustrates the results of training the algorithms using varying dimensional representations of country codes, compared with the pre-established baseline F1-scores. Results obtained from using original dataset.

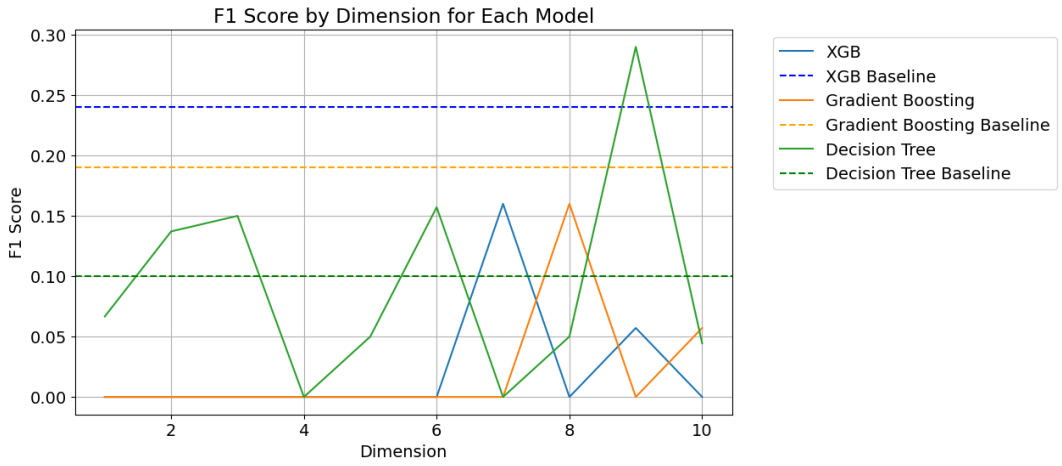

Figure 4: F1-scores using auto-encoded compressed representations of country codes in dataset (dimensions 1-10), original data version

*Best Decision Tree F1 score = **0.29** (9-dimensions).*

*Best Gradient Boosting F1 score = **0.16** (7-dimensions).*

*Best XGB F1 score = **0.0** (All-dimensions).*

Summary of auto-encoding method

Using an autoencoder to represent country codes appears to positively impact the algorithms when identifying useful patterns. The best achieved F1-scores when including auto encoded representations of country codes were Decision Tree (F1-score = 0.29, up from 0.1) from, Gradient Boosting (F1-score = 0.38, up from 0.19), and XGB (0.34, up from 0.24). Specifics of approach described in section 4.2.

This underscores the value of dimensionality reduction for high cardinality variables in highly imbalanced datasets, such as this one. Both the Gradient Boosting and XGB classifiers experienced notable improvements when utilizing SMOTE, whereas the Decision Tree classifier improved significantly when trained on the original data. A plausible explanation for this divergence is the vast disparity in complexity. Gradient Boosting and XGB are essentially highly complex iterations of the decision tree, incorporating numerous decision trees as weak learners during training. These may be too intricate to discern patterns in the original data, while the decision tree may be overly simplistic to learn patterns as effectively from SMOTE data.

The potential limitations of these findings are detailed in part 7, while specifics on how we've accounted for potential pitfalls, which could challenge the generalizability of the approach, are explained in section 6.4.

5.3: James-Stein Encoder

In this step, we implemented, tested, and evaluated the James-Stein encoder to assess its potential for improving performance. The James-Stein encoder is a target-based encoding technique. The motivation for adopting this encoding technique was its proven effectiveness in enhancing performance when training machine learning algorithms on highly imbalanced datasets, such as the one used in this project, compared to other popular encoding approaches.

However, it is important to note a potential drawback known as prediction-shift. Prediction-shift refers to the phenomenon where the underlying distribution of a dataset changes over time, resulting in a shift in the relationships between input features and the target value. Consequently, patterns and correlations observed in the training data may not hold for new data, leading to inaccurate predictions and reduced model performance (Breskuvien˙e & Dzemyda, 2023).

In our analysis we applied the James-Stein encoder using the imported module (Category Encoders, 2019).

The James-Stein encoder is a statistical method rooted in the James-Stein shrinkage phenomenon. Its purpose is to compress data dimensionality while preserving essential information.

The process of using the James-Stein encoder to compress country codes can be understood by following these steps:

1. Compute the mean value for each unique country code based on the binary target label indicating front companies. For example, if a country code has never been associated with a front company (label 1), the mean is 0. Similarly, if the front company label is 1 for half the occurrences of a country code, the mean is 0.5.

2. Calculate the global mean of the target variable, representing the average value of the binary front company label across the entire dataset.

3. Determine a shrinkage factor for each country code. This factor is applied to the mean estimate for that country code and determines the extent to which it is

shrunk towards the global mean of the target variable. The shrinkage factor considers the relationship between the estimated mean, the global mean, and the variance of the front company variable.

Generally, the shrinkage factor (s) for a given country code can be calculated using the following formula:

$$Equation\ 4) \qquad \mathbf{s = \max\left(0, 1 - \left(\frac{k * \sigma^2}{(n-1) * (\hat{\mu} - \mu)^2}\right)\right)}$$

Here, μ̂ equals the estimated mean of the country code, μ denotes the global mean of the target variable, and σ^2 equals the variance of the target variable. This formula determines the appropriate amount of shrinkage based on the discrepancy between the estimated mean and the global mean, taking into account the variance of the target variable.

4. Replace the original country codes with the shrunken values.

It is important to note that this description provides a general idea of the James-Stein encoder's concept, as the specific implementation may vary depending on the library or framework used.

The James-Stein encoder was easily implemented using the category encoder library in Python. This adaptation, proposed in (Zhou, 2015), makes the James-Stein encoder suitable for binary data like this. We specified that the encoding of country codes should be based on the binary front company column. Figures 5 and 6 offer an overview of how the James-Stein encoder transforms nominal categorical variables, specifically country codes, into numerical representations. This allows the country of registration information to be communicated through a single column, rather than needing one column for each unique country code.

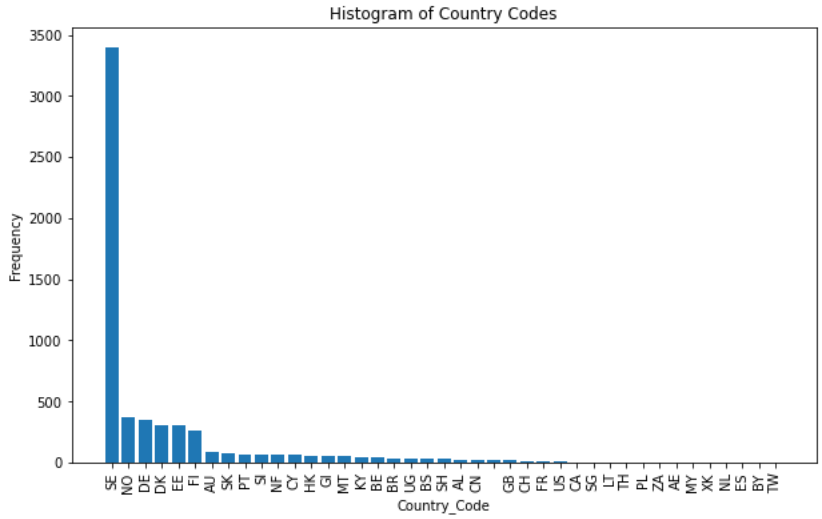

Figure 5: Histogram of country code distribution in dataset

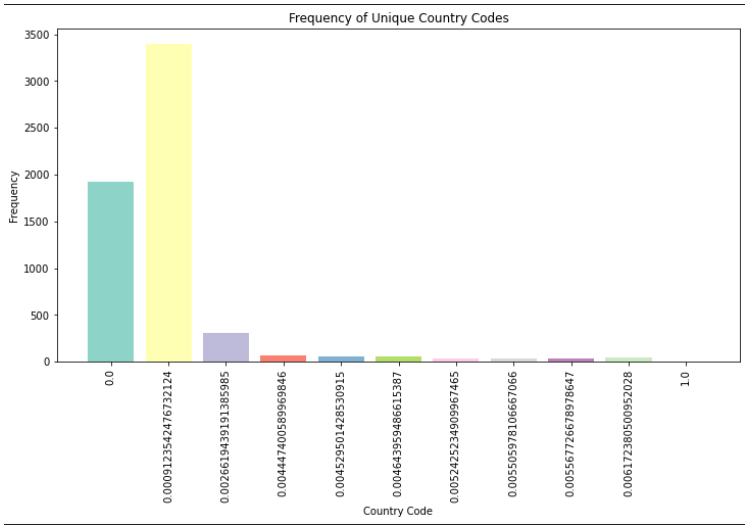

Figure 6: Illustrating the distribution of James-Stein encoded values in dataset

The figure below depicts the results derived from training the algorithms using 1-dimensional representations of country codes compressed via the James-Stein encoder. These outcomes are compared with the pre-established baseline F1-scores. It's important to note that these results include data obtained from both the original dataset and the dataset augmented using SMOTE.

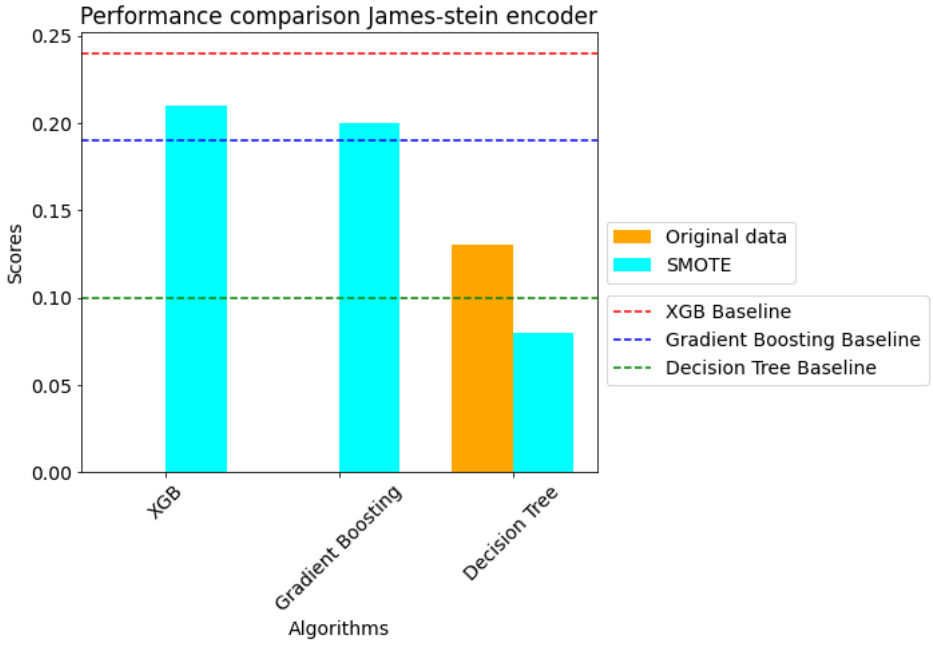

Figure 7: Illustrating the F1-score for each algorithm using James Stein compressed representations of country codes in the dataset. Includes original and SMOTE data

Summary of James-Stein encoding method

Using the James-Stein encoder appeared to yield slightly better results for Gradient Boosting and Decision tree, but poorer results for XGB. However, the improvements are unimpressive compared to the performance gains achieved by utilizing an autoencoder to compress country codes. This indicates that, while this encoding method might have potential benefits, other approaches were better at capturing the relationships within the data for this specific classification task. The James-Stein encoder is a linear method, yet the relationship between the country code and whether a given company is a front is context-dependent and therefore nonlinear. This nonlinearity can cause the James-Stein encoder to capture oversimplified representations, potentially misleading pattern learning. Autoencoders, on the other hand, function as unsupervised algorithms and do not assume any specific relationship between the features and the target variable.

5.4: Development, Testing, And Evaluation of New Custom Features

This structure of this descriptive section is:

1. Describe motivation and development approach of first feature.
2. Then describe motivation and development approach of second feature.
3. Finally, describe how including each feature on its own and in conjunction impacts the F1-score of the algorithm.

5.4.1: Feature Addition 1, Geographic Region Clustering

The forthcoming feature was intended to provide contextual information about a company's global position. It worked by establishing the geographical coordinates of all companies, identifying the region where the B4 client primarily conducted business, and defining the distance from this region for each observation.
It was deemed necessary to incorporate this feature, given the business problem at hand. The model being developed was targeted for deployment in a very specific setting. Consider this context: the model needed to be operated from the viewpoint of a single "source" company, such as a Danish pharmaceutical company. Based on our accumulated domain knowledge, it was widely accepted that the geographic location of a company was highly relevant when identifying suspicious companies. Therefore, it was crucial to consider the location of each company in the dataset. However, the significance of a location would change depending on the context of each source company.
For example, while dealing with a supplier located in the Cayman Islands might have raised suspicion for the Danish company, it wouldn't have been considered suspicious from the perspective of a fellow Cayman Islands company, as it was entirely expected to conduct business within one's own country. This reasoning underlined why this feature was deemed necessary. Domain knowledge mandated the inclusion of country code information in our dataset, but the business problem inherently necessitated that these country codes provoke suspicion at varying degrees depending on the situation. Consequently, this distance metric was needed to inform the model whether a given country code should be regarded as a red flag or not, relative to the source company.
In summary, the primary purpose of this feature was not to introduce information indicative of a front company. Instead, it aimed to establish a methodology that prevented the model from generating false positives by erroneously generalizing

patterns that were specific to other source companies conducting business in a context different from the current source company.

Below is an illustration of how the DBSCAN algorithm clustered observations based on longitude and latitude.

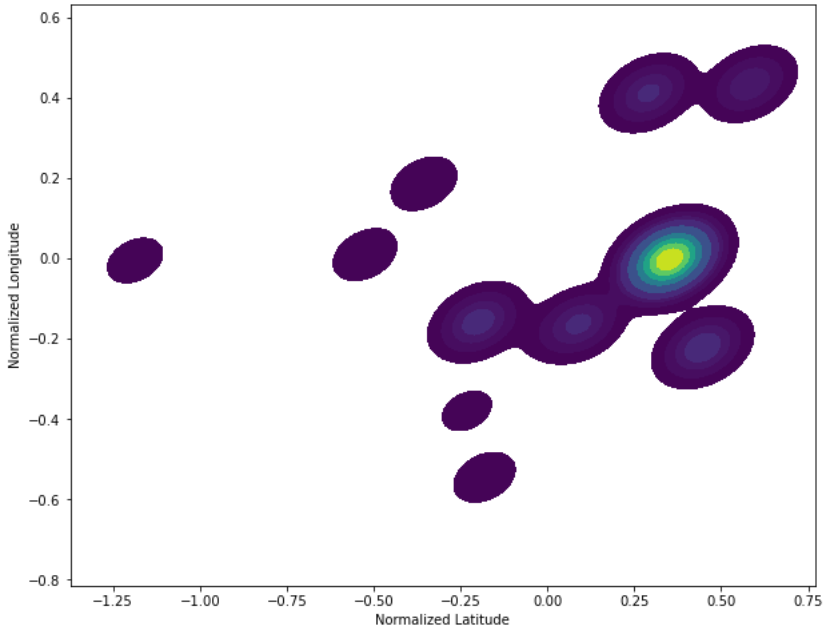

Figure 8: DBSCAN clusters of observed geographic locations in dataset

Technical Approach

Geocoding: This step involves converting the name of a place into corresponding coordinates, specifically longitude and latitude. By doing so, we assign a numerical value to represent a position. In this scenario, the conversion is based on the country code associated with each company. We utilized the Geopy library version 2.3.0 to obtain the coordinates used (Geopy Development Team, 2023).

Why missing values are not removed: The dataset contains some entries with missing country codes. In many situations, such observations are typically removed. However, keeping these observations may be vital as they could indicate suspicious companies. Nevertheless, for the feature creation, a position needs to be assigned for all companies. Therefore, companies without a specified country code are assigned to Point Nemo, the oceanic location furthest from any landmass. This assignment is somewhat arbitrary, but it allows the model to associate such observations with a fixed location which is far away from all other observations.

Clustering: To identify the primary area of business for the "source" company (i.e., the B4 user), we employ an unsupervised clustering algorithm known as

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) imported from using scikit-learn version 1.2.0 cluster module (scikit-learn Development Team, 2021). This algorithm was chosen due to its suitability for spatial clustering tasks and its ability to determine the number of clusters automatically, without the need for a pre-specified number. By considering the characteristics of the data, the algorithm creates clusters based on their inherent characteristics rather than an arbitrary number.

Distance Measures: Necessary adjustments have been made to the distance measurements to accommodate a spherical surface like that of the Earth. Below is an illustration of the spread of the derived distance variable.

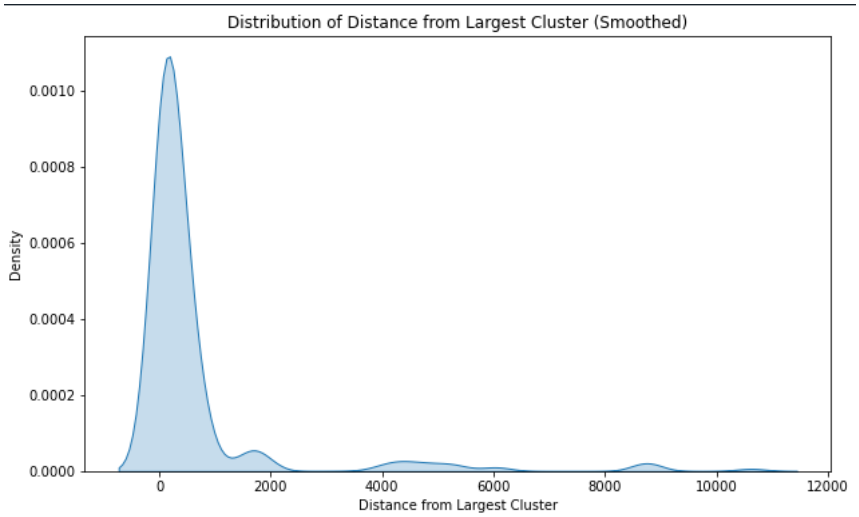

Figure 9: Spread of geographic distance from main cluster in the dataset

This can assist the algorithm in identifying outliers based on distance. Since physical distance frequently plays a significant role in selecting trade partners, this information can hold value.

5.4.2: Feature Addition 2, Adding Context to The Use of Currency

This strategy employed a logic similar to the one used above, but this time it focused on currencies. Three new, straightforward columns were added:

1. Currency corresponded: Indicated if the company used the same currency as their country of registration (0/1).

2. Used most common currency: Identified the currency most commonly traded by the B4 customer and specified if a given company used this currency (0/1).

3. Both negative: If the answers to the two previous columns were negative (0), a company was marked with a positive indicator (1) in this column.
The premise was that while using a currency that differed from the country of registration might have raised suspicion, this suspicion could be mitigated if the company was dealing in the preferred currency of its business partner. However, if a company was dealing in a currency that neither corresponded to its jurisdiction nor to its counterpart's preferred currency, it was deemed unusual. This circumstance was captured in the Both negative column.

Technical Approach

The implementation of this approach was quite straightforward. It involved the use of simple counting functions to return the currency with the highest value. The primary challenge was to correctly associate each country code with its respective currency. Automating the currency identification process was feasible for many countries, while others necessitated a more manual approach. This entailed a careful examination of the Currency corresponds column, where we reviewed the flagged rows and confirmed whether the identified currency indeed did not correspond. For instance, we manually defined that Norfolk's currency was the Australian dollar and that Saint Helena used the Saint Helena pound. The figure below depicts the proportion of observations in the dataset where the Both Negative condition holds true.

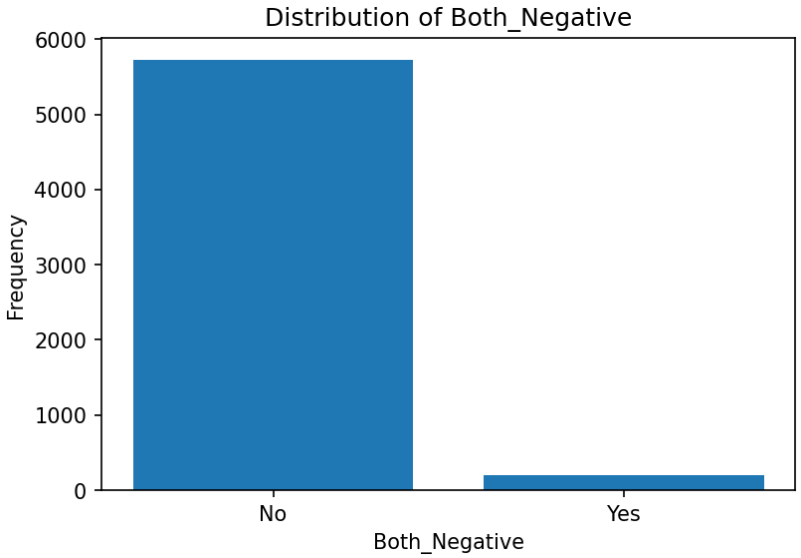

Figure 10: Proportion of observations in the dataset where the Both Negative feature is labelled 1

5.4.3: Impact of Including Custom Features

This subsections describes the how the inclusion of the different features outlined in sections 5.4.1 and 5.4.2 affected performance using the different algorithms.

Approach 1: Including all three currency related features.

This approach involves adding the following custom features:

- Currency corresponds.
- Uses most common currency.
- Both negative.

The figure below presents the outcomes resulting from the inclusion of the features discussed in this approach.

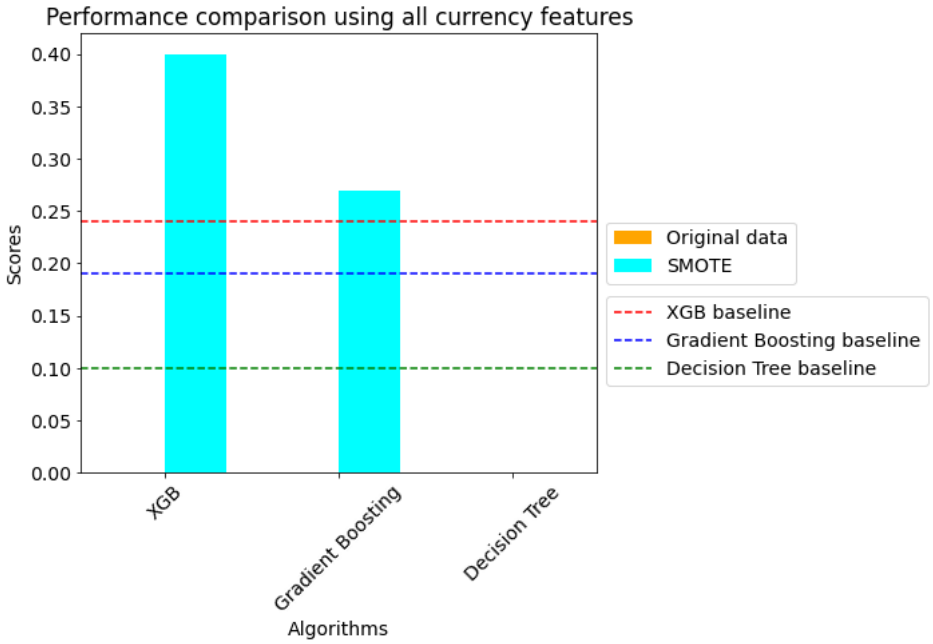

Figure 11: Illustrating the F1-score for each algorithm using all currency features. Includes original and SMOTE data

Including these features led to enhanced performance for when using SMOTE XGB (F1-score = 0.39, up from 0.24) and Gradient Boosting (F1-score = 0.27, up from 0.19). However, these same models scored a 0.0 when trained on the original data. The Decision Tree scored 0.0 for both sampling strategies, suggesting that the added features, and the potential multicollinearity they introduced, may have surpassed the algorithms' threshold for data complexity.

Approach 2: Including only both negative

As described during the construction of the features, the primary purpose of the three features is to comprise the third feature called **Both negative**. The other two features serve as indicators that a domain expert would consider, but they have limitations in accurately predicting the outcome in the real world. However, when both indicators are present, it serves as a strong red flag. The **Both negative** feature precisely indicates this scenario. Therefore, we remove the other two features since the feature already captures the information from both of them. This addresses the multicollinearity issue mentioned in the first method, and we are able to reduce the dimensionality by two.

The figure below presents the outcomes resulting from the inclusion of the features discussed in this approach.

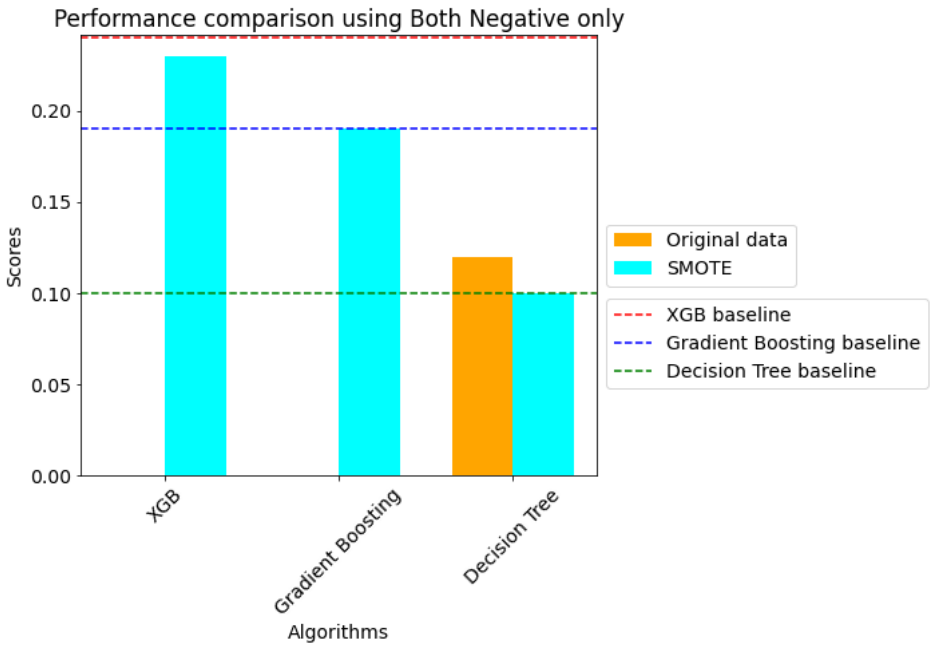

Figure 12: Illustrating the F1-score for each algorithm using Both Negative feature only. Includes original and SMOTE data

This method only introduced improvements for the decision tree when trained on original data. Gradient Boosting precisely matched its baseline performance, while XGB performed slightly worse.

Approach 3: Including cluster distance

Here, the cluster distance feature was introduced. The figure below presents the outcomes resulting from the inclusion of the features discussed in this approach.

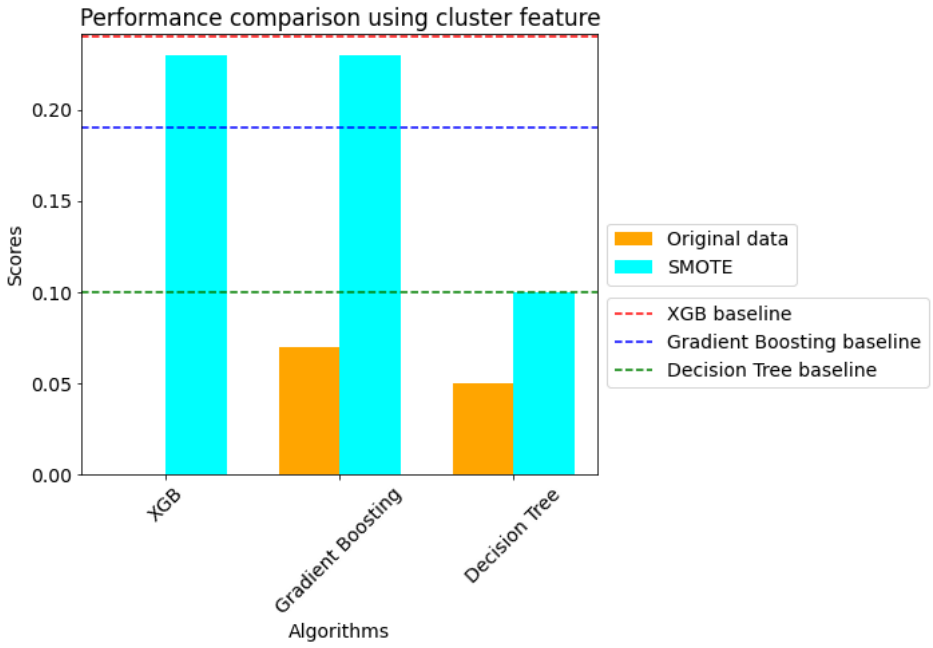

Figure 13: Illustrating the F1-score for each algorithm using cluster distance feature. Includes original and SMOTE data

Only Gradient Boosting demonstrated performance improvements compared to the baseline when using SMOTE. The other two algorithms, however, showed no improvement and even experienced slight decreases.

Summary of custom features' impact on performance.

When introduced individually, only the variation incorporating all three currency-related features demonstrated significant performance improvements compared to the baseline, especially for XGB using SMOTE. However, the inclusion of cluster features led to enhancements for the Gradient Boosting Classifier. At this point, we had enough testing results to conclude that the Decision Tree was inferior to Gradient Boosting and XGB when using SMOTE. Therefore, only these two models employing SMOTE were subjected to further testing.

5.4.4: Combining Custom Features With Auto-Encoding

In section 5.2, it was found that using auto-encoded representations had the most positive impact on the algorithm's ability to learn useful relationships in the data, resulting in the highest F1-scores. Furthermore, in section 5.4.3, we observed the positive impact of including the described custom features on performance. This section presents the results obtained when repeating the testing process described in section 5.2 with the inclusion of these features.

Here, the horizontal lines included in the figures represent the highest average F1 score achieved using the algorithms across all sampling techniques and feature representations. This addition allows us to gauge whether a particular alteration to the approach yields additional benefits when the model aims to learn patterns.

Approach 1: Auto-encoding with currency features included.

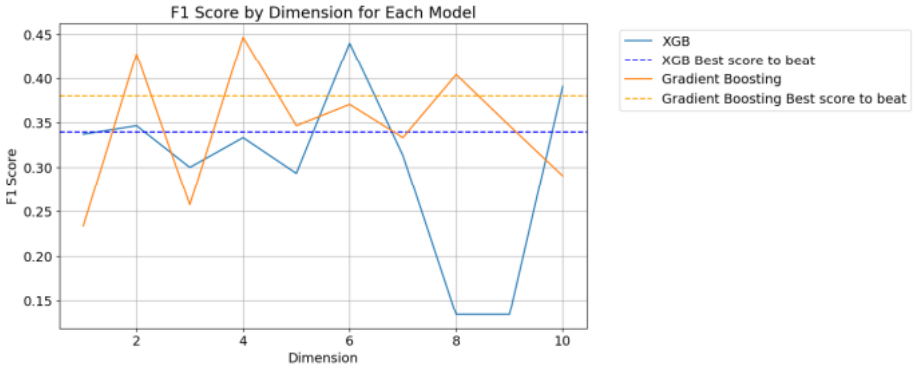

Figure 14: F1-scores using auto-encoded compressed representations of country codes in dataset (dimensions 1-10) while also including all currency features. SMOTE data

Gradient Boosting has achieved the best performance thus far with an F1-score of 0.45 using a 4-dimensional representation of country codes. XGB has also performed well, achieving the best F1-score of 0.44 with a 6-dimensional representation of country codes.

These specified scores form the horizontal line in the next plot, representing the benchmark score to surpass.

Approach 2: Auto-encoding with currency features and cluster distance feature

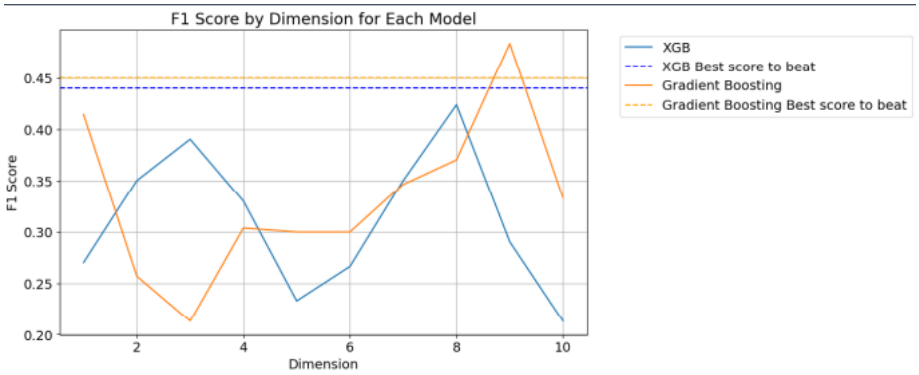

Figure 15: F1-scores using auto-encoded compressed representations of country codes in dataset (dimensions 1-10) while also including all currency features and cluster distance feature. SMOTE data

Gradient Boosting demonstrated further improvement with this representation, reaching an F1-score of 0.48 when using a 9-dimensional representation of country codes. On the other hand, XGB did not observe any enhancement in performance with this particular representation.

Summary of results when auto-encoding with custom features included

Both methods outlined in this subsection resulted in improvements. We found that Approach 2, which incorporates currency features along with the cluster distance feature, provided the most informative representation. This assisted the Gradient Boosting algorithm in achieving the highest results recorded in this thesis, with an F1-score of 0.48.

5.5: Incorporating hyperparameter tuning

A parameter is an intrinsic configuration within a model, estimated from data during model training. Once estimated, these parameters remain fixed when used for making predictions on new data. For example, a parameter could be the weight a model assigns to the country code "US", or a coefficient for a binary variable indicating whether a supplier has issued a certain percentage of sequential invoices.

In contrast, a hyperparameter is predetermined externally, often at the discretion of the data scientist, before model training. Hyperparameters are not typically influenced by the data. Examples of hyperparameters include the number of decision trees in a random forest, the learning rate used by the model to update its parameters, or the number of clusters to be defined when using unsupervised learning techniques.

To identify the optimal hyperparameter setting for model training, we can use grid search—a method inspired by results derived from the data. Consider this simple analogy to illustrate how grid search works:

Think of yourself as a cook preparing a new dish for the first time. The recipe has a set of ingredients with a *range* of measurements—1-2 cups of cream, 2-4 eggs, 0.5 – 2 teaspoons of baking powder, and 3-5 ounces of flour. As you are uncertain of the exact combination that yields the best result, you decide to cook the dish several times, each time using different measurements within the suggested ranges. At the end of each iteration, you taste the dish. By the end of the process, you'll have identified the optimal combination of ingredients. Grid search follows this logic, using different combinations of hyperparameters instead of ingredients. Because there may be numerous hyperparameters to adjust, we predefine which ones to include in the grid search and the ranges to test. This method automatically iterates through the possible combinations and identifies the optimal hyperparameter settings that yield the strongest prediction performance. Hyperparameters vary among algorithm types, so you must use your knowledge and judgement to determine the most critical hyperparameters to adjust. This process involves a substantial number of iterations and can be quite time-consuming, hence the initial testing stage of this project. Since we already have an initial understanding of which algorithms and sampling methods are better suited for the problem, we can eliminate candidates that would likely produce poor results.

Based on the final data representation described in approach 2 from section 5.4.4, both random search and grid search were performed to optimize the hyperparameters for the Gradient Boosting Classifier. Both methods were utilized for the following reasons:

1. Grid search is effective in finding the best solution among all specified combinations of hyperparameters. However, it can be computationally intensive

and time-consuming due to its exhaustive search approach. In this project, conducting a comprehensive grid search was not feasible due to hardware limitations.

2. Random search, on the other hand, allows for a broader range of hyperparameter combinations and is less computationally demanding. Since the search is random, it explores different areas of the hyperparameter space. However, there is no guarantee of finding the optimal solution.

3. By combining both methods, the likelihood of finding the best hyperparameters is increased. Random search can quickly explore a large portion of the hyperparameter space, while grid search can fine-tune the selected regions.

In summary, the combination of random search and grid search provides a balanced approach to hyperparameter tuning, considering both computational limitations and the exploration of a wide range of possibilities.

Grid search and random search were both implemented using scikit-learn version 1.2.0 model selection module (scikit-learn Development Team, 2021).

Summary of how hyperparameter tuning impacted performance.

Surprisingly, hyperparameter tuning failed to enhance the F1-score when the best-performing combination of features, sampling, and algorithm was used. The default hyperparameters from scikit-learn version 1.2.0 outperformed any combination we discovered in our searches.

Chapter 6, Discussion of potential pitfalls

In this section, we address and discuss potential pitfalls related to machine learning that we needed to be aware of or account for during our analysis. Neglecting these could hinder the progress of our analysis or lead to overly optimistic estimates of performance, resulting in false conclusions.

6.1: Benefits and drawbacks of summary features

Consolidation of the transactions by using summary statistics is necessary in light of the business problem: The objective is not to identify individual front-transactions but rather front companies.

Front companies are best identified when evaluating the bigger picture. Transactions on their own provide negligible information when searching for front

companies or fraud in general. It is when we zoom out and evaluate all transactions together that we can begin to identify distinguishing patterns.
As described in the data structure section, summary statistics pertaining to the transactions are utilized for this analysis. This approach has its benefits and drawbacks, which are discussed below:

Benefit 1: By using the mean values of invoices and credit notes for a company, the algorithm can consider their central tendency.

Benefit 2: Highlighting maximum and minimum values can be useful when identifying outliers. This is especially important in the field of fraud detection, as it often involves isolating outliers.

Benefit 3: Standard deviation provides a measure of variability and dispersion in transaction amounts. An algorithm may use this information to spot irregularities in transaction patterns.

Benefit 4: Including the volume of transactions provides insight into the activity level of a company and contextualizes the remaining summary information. The algorithm may benefit from this information when searching for patterns.

Benefit 5: The number of round amounts is the summary information most influenced by domain knowledge. I find it useful to quantify, as using round amounts is typical of fraudulent companies.

These summary features capture vital aspects of transactional behaviour. By evaluating them in conjunction, a model can learn distinguishing patterns that are indicative of front companies.
However, as is the case with many aspects of machine learning, the utilization of summary features comes with certain drawbacks. Here are some issues that may arise with this approach:

Caveat 1: Some of the summary features may exhibit high correlation, introducing redundant information to the model without providing additional value. In such

cases, we increase the dimensionality of the data without gaining much, or any benefit. Higher dimensionality adds complexity to the problem (Li et al., 2016).

Caveat 2: Redundancy can also lead the model to overfit the training data, impairing its ability to generalize to unseen observations (Li et al., 2016).

Caveat 3: Highly correlated features can introduce multicollinearity, which can confuse the algorithm when attempting to identify the true relationship between the features and the target variable. It is impossible to isolate the true causal variable with certainty (Dormann et al., 2013).

The figure below illustrates the correlations between different summary features, namely the aggregate data related to invoices, for each observation. A value of 1 signifies a perfect correlation, explaining why each feature registers a value of one when intersecting with itself in the matrix.

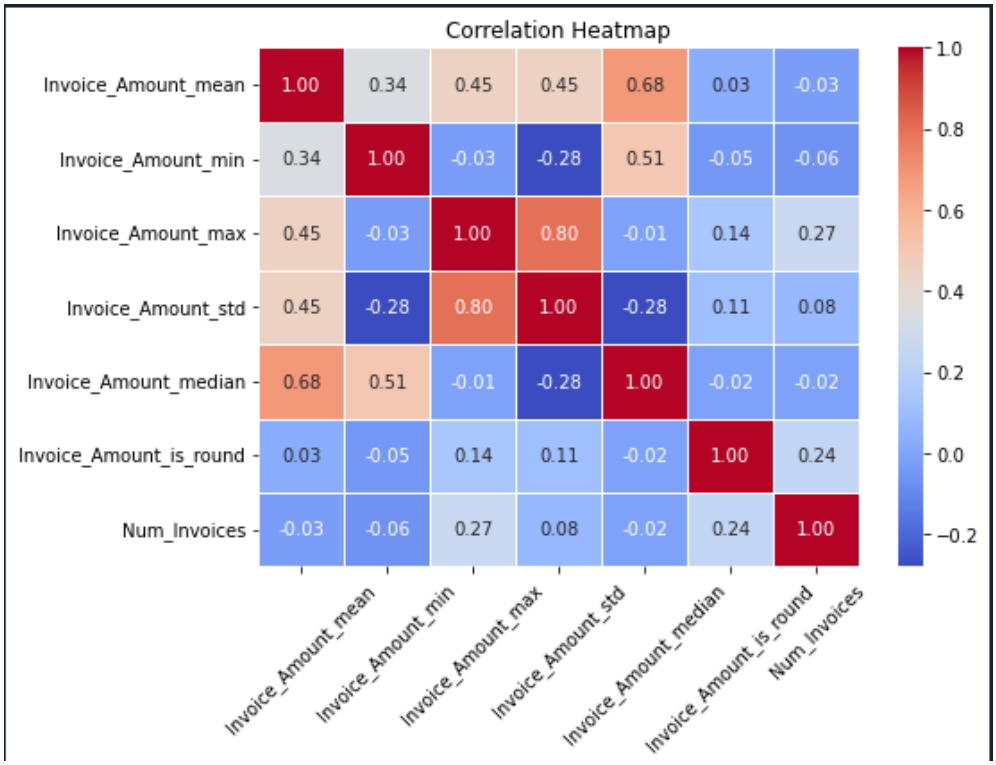

Figure 16: Correlation heatmap between all invoice related features

6.2: Method For Comparison Between Approaches

In this development process, test estimates are obtained by evaluating the F1-score of each iteration. For this project, we calculate the F1-score as the average result from a five-fold cross-validation method. Cross validation is a widely used method as it provides a reliable assessment of the general performance of predictive models (Berrar, 2018). To ensure reliable performance estimates, each validation fold stratifies the number of positive instances, maintaining the proportion of observations labelled as actual front companies similar to that in the training fold. This approach is widely regarded as a best practice as it reduces the variance of model performance estimates compared to a single train/test split. By employing this approach, we gained more confidence in attributing any variations in the F1-score between iterations to differences in generalization ability. Cross validation was implemented using scikit-learn version 1.2.0 model selection module (scikit-learn Development Team, 2021).

6.3: Scaling Data For Testing

The features were scaled using standardization instead of normalization, which is another commonly used approach. Standardization is often more practical for many machine learning algorithms, especially those that rely on gradient descent during optimization (Raschka, Liu, & Mirjalili, 2022). Furthermore, normalization is sensitive to outliers, which can pose challenges for algorithms aiming to identify patterns. Since the business problem can be characterized as a form of fraud detection, we prefer employing techniques that are robust to outliers, as detecting outliers is crucial in fraud detection.

Tree-based algorithms are generally robust to unscaled features, but their performance is not negatively affected by using scaled features either. Therefore, using the same scaling approach for the tree-based algorithms tested in section 3.3.1 (Decision Tree, Random Forest, Gradient Boosting) was a sound approach. However, in subsequent sections, feature scaling was not applied due to the inherent robustness of tree-based algorithms to unscaled features. Additionally, the cross-validation scaling approach described in section 5.3 becomes computationally expensive when conducting hyperparameter tuning.

6.4: Avoiding overoptimistic estimated by preventing data leakage

6.4.1: Leakage prevention when scaling

In section 3.3.1, the performance of each model is evaluated based on the average F1-score obtained through five-fold cross-validation. To avoid data leakage, scaling is applied separately to each fold during the cross-validation process. This ensures that when a particular fold is used as the test set, no information is leaked to the other folds during the scaling step. By preventing such leakage, we can obtain more reliable performance estimates that are not overly optimistic due to the inclusion of leaked information.

6.4.2: Leakage preventing when using auto-encoder

To prevent information leakage from the test sets into the training process during cross-validation, we employed a multi-step procedure. It begins with the binary encoding of the country code variable using a through using Pandas get_dummies function (Pandas Development Team, 2023).This encoding results in a separate data frame, containing binary columns for each unique country code, with indices that match the original dataset.

Next, the dataset is divided into five distinct, stratified segments or folds for evaluation. Each fold is further divided into training, validation, and testing subsets. A similar process is applied to the data frame with the binary-encoded country codes, ensuring that the indices of the subsets correspond, and thus they contain the same observations.

The aforementioned training and validation subsets are further split into separate training and validation subsets. This step is also applied to the corresponding binary-encoded country codes.

Following this, an autoencoder is trained on the binary-encoded country codes of the training subsets, aiming to reduce dimensionality.

Once trained, the autoencoder transforms the country codes for all subsets - training, validation, and testing, applying the compressed representations of the country codes.

Finally, these transformed country codes are concatenated with the corresponding subsets of the original data. The entire purpose of this procedure is to prevent any

information leakage between training and testing sets during cross-validation when using an autoencoder. This is ensured by carrying out preprocessing independently for each cross-validation fold.

6.4.3: Leakage prevention when applying synthetic up sampling of minority class (SMOTE)

To implement SMOTE anew for each fold within cross validation, we employed a pipeline using the imbalanced-learn library (Lemaître, Nogueira, & Aridas, 2017). This methodology ensures that SMOTE is only applied to the training set in each validation fold. As such, no positive samples in the test set are synthetically generated observations from SMOTE - they are only authentic examples from the original data. This strategy prevents data leakage, which could occur if synthetic observations were created based on traits from the entire dataset upon which SMOTE is applied. By adhering to this approach, we enhance the reliability of estimates in which SMOTE is utilized to support model training.

6.5: Correlation analysis

The goal was to examine the correlations among the input features to determine if any of them could be considered redundant to the extent that their removal would justify a reduction in dimensionality. It was evident that the summary features represented different aggregates of the same entity (Invoices), leading to a noticeable degree of correlation between them. The question arose as to whether the additional dimensionality introduced by these features outweighed the unique information they provided. The mean and median were likely to be quite similar, with the median offering a better representation of a randomly selected invoice but failing to capture information about central tendency like the mean did. Considering that we had information about minimum and maximum values, it was worth considering the removal of the mean feature since we also had the median. However, it was found that performance suffered when the mean feature was removed. In fact, although the aggregates were moderately correlated, removing any of them individually resulted in poorer performance. It was possible that the complexity of the gradient boost algorithm allowed it to handle the added

dimensions effectively, and the unique information provided by each aggregate feature served as a net benefit.

To summarize the chapter, we've highlighted that while the structure of the dataset for this analysis aligns with the business problem, it does carry potential drawbacks such as multicollinearity and redundant information. To ensure dependable estimates, we elected to evaluate each iteration's performance based on the F1-score from a five-fold cross-validation method rather than a single train/test split score. In terms of scaling the input features, standardization emerged as the superior option to normalization since it is more robust to outliers, making it more suitable for data related to fraudulent behavior where outliers are anticipated. We also identified correlations between input features, especially among the aggregate invoice features through a correlation analysis. Nevertheless, we concluded that the unique information provided by each feature justified the additional dimensionality.

Chapter 7, Limitation of Analysis

The findings in this analysis were obtained through methods designed to yield realistic estimates, adopting standard practice approaches. These methods encompass performance evaluation based on stratified multiple-fold cross-validation, selection of suitable performance metrics by carefully addressing the business problem, and adherence to correct procedures to avoid inadvertent information leakage about test sets during algorithm training and testing phases. Nevertheless, the findings in this thesis should be interpreted as preliminary indicators, suggesting promising strategies for implementing machine learning classification algorithms effectively. These are intended for binary front company detection within the slim financial transactions' context. The solution identified in this thesis has several limitations that question its applicability to unseen real-world data.

First, the dataset used in this study comprises solely synthetic observations. Despite being meticulously constructed by a domain expert based on actual cases, these observations might not perfectly mimic patterns present in real-world scenarios. Moreover, the dataset lacks the required variety to train a model capable of high accuracy across diverse users. It only represents the perspective of a single-source company operating in a specific location and industry.

The method employed in this study merits careful scrutiny, as it might have introduced a form of research bias arising from the development structure itself. The method implements and tests various interventions and techniques iteratively, selecting the modifications that yield the greatest improvements in F1-scores and discarding the others. This iterative process might lead to the solution being heavily biased towards the specific data and testing methods used, potentially compromising its robustness when applied to new data.

Furthermore, performance estimates obtained using auto-encoded compressed representations of country codes exhibited inconsistencies across different algorithms, dimensions, and even multiple iterations of the same test. The sources of these inconsistencies are challenging to definitively identify, but they could plausibly be attributed to the inherent nature of auto-encoders - unsupervised neural networks that produce unique outputs each time they are trained, even given consistent random states. This variability can result in differing levels of accuracy and thus scrutiny of the method's robustness is warranted.

Finally, it's important to acknowledge that the overall analysis and results presented in this thesis are somewhat limited in scope, and likely address just a small fraction of potential tests and interventions that could enhance an algorithm's proficiency in identifying front companies, as measured by the F1 score. Therefore, it's fair to suggest that the insights offered here, although valuable, might pale in comparison to the findings that could emerge from a more expansive, thorough, and expert-driven analysis.

Chapter 8, Conclusion And Further Steps

Enhancing our ability to detect front companies could provide significant societal benefits, as these entities often facilitate immoral and illegal activities, such as money laundering, corruption, bribery, theft, organized crime, and even terrorist financing. Identifying front companies may assist law enforcement agencies and international governmental bodies in preventing adverse outcomes. Moreover, unwitting business relationships with front companies can inflict considerable damage on legitimate companies, reinforcing the importance of effective detection mechanisms.

We sought to answer the research question by attempting to develop a machine learning classification system that could achieve the highest possible F1-score.

The prediction task involved accurately classifying companies as either front companies or not. Given that misclassifications can be highly detrimental in practical scenarios, we chose the F1-score as our performance metric because it penalizes both false negatives and false positives. B4 Investigate provided us with the dataset for this analysis, which they developed themselves. The dataset served as a useful basis for our analysis, as it contains only variables that real-world companies are likely to have readily available. This increases the likelihood that our insights could be useful for practical applications. The accessibility of this kind of data facilitates further research building on our findings, notwithstanding potential legal obstacles to data collection.

The primary limitation of the dataset is its extreme class imbalance: 99.8% non-front companies and only 0.2% front companies. Such imbalance presents substantial challenges for algorithms in identifying true relationships. The second concern stems from the synthetic nature of the data. While it is based on real cases and crafted by domain experts at B4 Investigate, its synthetic composition raises questions about the generalizability of our findings, as well as the applicability of the champion model outlined in this thesis.

We initiated our analyses by testing a variety of commonly used machine learning classification algorithms to gain a general understanding of the types best suited for our data and problem. We imported machine learning algorithms from scikit-learn version 1.2.0, including Logistic Regression, Support Vector Machine, Naive Bayes, Decision Tree, and Gradient Boosting. Additionally, we utilized the Extreme Gradient Boosting (XGBoost) algorithm, which was imported from the xgboost module (Chen & Guestrin, 2016). Notably, we found that tree-based algorithms significantly outperformed the others, particularly when trained on synthetically up sampled data using SMOTE. Consequently, we carried forward only Decision Tree, Gradient Boosting, and XGB for further analysis.

Dimensionality reduction was a central theme in this thesis. We discovered that using a neural network like an autoencoder is likely a more appropriate method to create compressed representations of high cardinality features, such as country codes, compared to using a target-based encoding method like James Stein encoding. When the algorithms were trained on data wherein country codes were presented as auto-encoded compressed representations, they achieved significantly improved F1-scores compared to the original encoding. This

suggests that considering the geographical location or registered domain of a company is highly relevant for identifying potential front companies.
We propose this is because target-based encoding presumes linear relationships, which we do not believe holds true for the association between front companies and country codes. Therefore, an agnostic approach like an autoencoder seems better suited.
Additionally, the introduction of new features tailored to provide useful context proved beneficial for the algorithms. The transformation of data to generate novel features, which provided insights into geospatial clusters within the dataset and the relative context of currencies used by each company, enhanced the model's capacity to discern relationships.
Through this iterative, step-by-step process, we determined that the Gradient Boosting Classifier, trained on a dataset in which the minority class was synthetically up-sampled to match the majority class using SMOTE, proved to be the most effective system. It's worth noting that this system incorporates the previously mentioned tailored features and auto-encoded representations of country codes. Interestingly, in this particular instance, we didn't find that tuning the hyperparameters significantly improved the results.
When trained on the original dataset, the list of common classification algorithms achieved an average F1-score of 0.02, which is practically useless to deploy in a practical setting. **The final system achieved an average F1-score of 0.48 when tested through a five-fold stratified cross validation. This demonstrates that there is in fact potential for effective adaptation of machine learning in front company detection, and as shown by the stark difference in performance, choosing the correct approach is crucial when doing so**.
The findings in this thesis can serve as useful preliminary indicators guiding the direction of future research. Greater efforts into the effective encoding of high-cardinality features such as country codes, and possibly currencies, warrant further investigation. In addition, refining the tailored features presented here using domain knowledge, or creating novel features that aim to depict other relevant aspects of front companies, is encouraged. Furthermore, an exploration of more specialized algorithms and comprehensive searches for optimal hyperparameters would also be beneficial.

The success of future efforts depends on the development of improved datasets. To our knowledge, the dataset used in this thesis is unique. The variables used in this dataset are readily available; the more challenging task would be to label observations in any future datasets using real company transactions. Perhaps an iteration of the system proposed in this thesis, combined with a rule-based filter, could aid in the task of labelling new datasets.

References

1. Raschka, S., Liu, Y. (Hayden), & Mirjalili, V. (2022). Machine Learning with PyTorch and Scikit-Learn. Packt Publishing.

2. Berrar, D. (2018). Cross-Validation. In Reference Module in Life Sciences. DOI: 10.1016/B978-0-12-809633-8.20349-X.

3. Breskuvienė, D., & Dzemyda, G. (2023). Categorical Feature Encoding Techniques for Improved Classifier Performance when Dealing with Imbalanced Data of Fraudulent Transactions. International Journal of Computers Communications & Control, 18(3), 5433. DOI: 10.15837/ijccc.2023.3.5433.

4. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature Selection: A Data Perspective. ACM Computing Surveys, 50(6), Article 94. DOI: 10.1145/3136625

5. Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., & Lautenbach, S. (2013). Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. Ecography, 36(1), 27-46. DOI: 10.1111/j.1600-0587.2012.07348.x.

6. Chawla, N. V., Bowyer, K., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16(1), 321-357. DOI: 10.1613/jair.953.

7. Zhou, X. (2015). Shrinkage Estimation of Log-odds Ratios for Comparing Mobility Tables. Sage Publishing, 45(1), 1-1. DOI:https://doi.org/10.1177/0081175015570097

8. Higgins, M., Morino, V., & Iyer, N. (2018). Imperialism, Dirty Money Centres, and the Financial Elite. In Jo Grady, Chris Grocott (Eds.), The Continuing Imperialism of Free Trade (pp. 140-150). Routledge.

9. Balakina, O., D'Andrea, A., & Masciandaro, D. (2017). Bank secrecy in offshore centres and capital flows: Does blacklisting matter? Review of Financial Economics, 32, 30-57. DOI: 10.1016/j.rfe.2016.09.005.

10. Financial Action Task Force. (2006). Trade Based Money Laundering. FATF/OECD. Retrieved from https://www.fatf-gafi.org/en/publications/Methodsandtrends/Trade-basedmoneylaundering.html.

11. Ranzato, M., Poultney, C., Chopra, S., & LeCun, Y. (2007). Efficient Learning of Sparse Representations with an Energy-Based Model. In Proceedings of the Advances in Neural Information Processing Systems (NIPS 2006). Courant Institute of Mathematical Sciences, New York University, New York, NY 10003.

12. scikit-learn Development Team. (2021). scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org.

13. Geopy Development Team. (2023). Geopy: Geocoding library for Python. Retrieved from https://geopy.readthedocs.io.

14. Pandas Development Team. (2023). Pandas: Flexible and powerful data analysis / manipulation library for Python. Retrieved from https://pandas.pydata.org.

15. NumPy Contributors. (2023). NumPy: The fundamental package for scientific computing with Python. Retrieved from https://numpy.org.

16. TensorFlow. (2023). TensorFlow: An end-to-end open source platform for machine learning. Retrieved from https://www.tensorflow.org.

17. Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. Journal of Machine Learning Research, 18(17), 1-5. Retrieved from http://jmlr.org/papers/v18/16-365.html.

18. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16 (pp. 785-794). DOI: 10.1145/2939672.2939785.

19. Category Encoders. (2019). Category Encoders: A set of scikit-learn-style transformers for encoding categorical variables into numeric with different techniques. Retrieved from https://contrib.scikit-learn.org/category_encoders/.

20. Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9(3), 90-95. DOI: 10.1109/MCSE.2007.55.

21. Waskom, M. (2021). Seaborn: Statistical Data Visualization. Journal of Open Source Software, 6(60), 3021. DOI: 10.21105/joss.03021.

22. Rocha-Salazar, J.-J., Segovia-Vargas, M.-J., & Camacho-Miñano, M.-M. (2022). Detection of shell companies in financial institutions using dynamic social network. Expert Systems with Applications, 207, 117981. DOI: https://doi.org/10.1016/j.eswa.2022.117981

23. Usman, A., Naveed, N., & Munawar, S. (2023). Intelligent Anti-Money Laundering Fraud Control Using Graph-Based Machine Learning Model for the Financial Domain. Journal of Cases on Information Technology, 25(1), 1–20. DOI: https://doi.org/10.4018/JCIT.316665

24. United Nations Office on Drugs and Crime (UNODC). (2011). Estimating illicit financial flows resulting from drug trafficking and other transnational organized crimes. Research Report. Vienna, Austria. Retrieved from https://www.unodc.org/documents/data-and-analysis/Studies/Illicit_financial_flows_2011_web.pdf.

25. D'Antuono, S. M. (2019, May 21). Combating illicit financing by anonymous shell companies. Federal Bureau of Investigation. Retrieved from https://www.fbi.gov/news/testimony/combating-illicit-financing-by-anonymous-shell-companies

26. Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR). DOI: https://doi.org/10.48550/arXiv.1412.6980

27. Iyer, N. (2019). How to Find Fraud and Corruption: Recipes for the Aspiring Fraud Detective. Routledge.
28. International Consortium of Investigative Journalists. (2016). The Panama Papers: Exposing the rogue offshore finance industry. Retrieved from https://www.icij.org/investigations/panama-papers/