KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Detecting spoofing in financial markets: An unsupervised anomaly detection approach

A case study at Nasdaq

EMILIA RIESCHEL

Stockholm, Sweden, 2024

Detecting spoofing in financial markets: An unsupervised anomaly detection approach

A case study at Nasdaq

EMILIA RIESCHEL

Degree Programme in Computer Science and Engineering
Date: June 3, 2024

Supervisors: Martin Rehn, Ivar Blohm
Examiner: Erik Fransén
School of Electrical Engineering and Computer Science
Host company: Nasdaq
Swedish title: Detektering av spoofing på finansmarknader med oövervakad maskininlärning
Swedish subtitle: En fallstudie på Nasdaq

© 2024 Emilia Rieschel

Abstract

Efficient methods for detecting illegal trading are essential for banks and stock exchanges to ensure market integrity and protect traders' interests. Traditionally, rule-based algorithms have been used to identify illegal trading activities. However, the potential of machine learning for fraud detection has increasingly been recognized in recent years, particularly for unsupervised anomaly detection, as labeled data is often rare in this field. This thesis evaluates the performance of such methods in detecting a type of illegal trading called spoofing. Spoofing involves placing large orders in the market with no intention of execution, aiming to influence financial market prices. Based on recent advancements, a hybrid model combining an autoencoder with a one-class classifier was developed and compared to common unsupervised anomaly detection methods, including the isolation forest, one-class support vector machine, and a standalone autoencoder. Additionally, feature importance was evaluated using two methods to determine which characteristics of order book data most significantly contribute to spoofing detection. Given this field's scarcity of labeled data, a synthetic dataset was generated for validation and performance evaluation. The results revealed that while the hybrid models underperformed, the isolation forest, particularly when trained on the most important features, achieved the highest performance, with an AUC ROC score of 0.82 and an AUC PR score of 0.30 on the final dataset. Despite these achievements, there is room for improvement, especially in reducing the false positive rate to make these models useful within trading surveillance. The synthetic dataset proved highly effective in representing real spoofing scenarios, and the feature importance methods offered valuable insights into the detection of spoofing.

Keywords

Unsupervised learning, anomaly detection, trade-based market manipulation, spoofing, layering

Sammanfattning

Effektiva metoder för att upptäcka illegal handel är avgörande för banker och börser för att säkerställa marknadens integritet och skydda handlarnas intressen. Traditionellt har regelbaserade algoritmer använts för att identifiera olaglig handelsverksamhet. Potentialen med maskininlärning för att upptäcka bedrägerier har dock blivit alltmer erkänd under de senaste åren, särskilt för oövervakad anomalidetektion, eftersom märkt data ofta är sällsynt inom detta område. Denna avhandling utvärderar prestandan hos sådana metoder för att upptäcka handelsbaserad marknadsmanipulation, med fokus på spoofing. Spoofing innebär att det läggs stora ordrar på marknaden utan avsikt att de ska gå till avslut, i syfte att påverka priserna. Baserat på de senaste framstegen utvecklas en hybridmodell som kombinerar en autoencoder med en one-class-klassificerare och jämförs med vanliga oövervakade metoder för upptäckt av anomalier, inklusive isolation forest, one-class support vector machine och en fristående autoencoder. Vidare utvärderades vilka egenskaper hos orderboksdata som mest signifikant bidrar till spoofingdetektering genom att använda två olika metoder för feature-importance. Med tanke på detta fälts brist på märkt data genererades ett syntetiskt dataset för validering och prestandautvärdering. Resultaten visade att medan hybridmodellerna underpresterade, uppnådde isolation forest, särskilt när den tränades på de viktigaste features, den högsta prestandan med ett AUC ROC på 0.82 och ett AUC PR på 0.30 på det slutgiltiga datasetet. Det finns dock utrymme för förbättringar, särskilt när det gäller att minska antalet falska positiva för att göra dessa modeller användbara inom handelsövervakning. Det syntetiska datasetet visade sig vara mycket effektiv för att representera verkliga spoofingscenarier, och metoderna för feature-importance gav värdefulla insikter om detektering av spoofing.

Nyckelord

Oövervakad maskininlärning, anomalidetektion, tradebaserad marknadsmanipulation, spoofing, layering

Acknowledgments

I want to thank everybody who has made this thesis possible. First, I want to thank my supervisor at KTH, Martin Rehn, who has given me invaluable support, guidance, and feedback throughout the project. Furthermore, I want to thank my examiner, Erik Fransén, for his thorough and constructive feedback. Finally, I want to thank my team at Nasdaq, especially my supervisor, Ivar Blohm, whose explanations of theory, as well as his support and feedback, have been essential during this project.

Stockholm, June 2024
Emilia Rieschel

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Financial markets can be seen as the engines of economic growth globally [1], facilitating capital flows and liquidity. Within these markets, participants exchange financial assets such as stocks and derivatives [2]. Although trading is the core activity, encompassing the buying and selling of financial assets [3], not all trading is lawful; practices like insider trading are illegal. Effectively detecting such illegal trading is essential to maintain market integrity and protect trader interests [4].

Traditionally, exchange regulators have detected illegal trading using a top-down approach. An automated surveillance system generates alerts based on known patterns and predefined thresholds. Surveillance analysts then handle these alerts by investigating the associated transactions further. However, this type of system faces several challenges. One challenge is the enormous volume of messages[1], which makes further investigation of an alert laborious. Another challenge is the difficulty of completely formalizing abnormal trading behavior using predefined patterns and thresholds. [4, 5, 6]

In recent years, the potential of machine learning in handling these challenges has gained increased attention [6]. However, the effectiveness of machine learning in this domain faces significant challenges, primarily due to the scarcity of labeled data. Labeled data is rare because most trades are legal, and confirming illegal trading is costly [5]. Another challenge is that an ideal machine learning model in this context would exhibit no false negatives, although false positives are considered tolerable. Nevertheless, a high rate of false positives requires extensive analyst intervention, increasing operational costs.

[1] The messages describe the changes in the limit order book, including when an order is added, amended, traded, or canceled.

So far, limited research has been conducted on using machine learning to detect illegal trading. A potential barrier to extensive research in this area is accessing requisite data, typically held by stock exchanges or banks. Consequently, conducting meaningful research on illegal trading requires collaboration with such institutions.

One area of illegal trading wherein machine learning may significantly impact is detecting trade-based market manipulation. This type of illegal trading is complex and may seem legal in appearance, which is why detecting trade-based market manipulation remains an open problem [1]. Additionally, with the rise of algorithmic trading, which has enabled traders to place and cancel orders in milliseconds, trade-based market manipulation has become more prevalent [7]. One such strategy, known as spoofing, involves placing large orders in the market with no intention of execution, aiming to influence financial market prices [8]. Since there has been a sharp spike in spoofing prosecution cases in recent years [8], it is of interest to all institutions and banks to discover new effective ways to detect spoofing in a reliable manner.

1.1 Purpose

This thesis aims to evaluate the performance of machine learning methods on spoofing detection in limit order book data. Since labeled data is often rare within this field, and spoofing strategies constantly evolve, it is motivated to use unsupervised anomaly detection methods trained on normal trading. However, labeled data is still needed to evaluate unsupervised methods properly. This thesis addresses the challenge by utilizing both a limited set of actual spoofing cases and a larger synthetic dataset. This methodology is widely accepted within the financial industry due to the scarcity of market manipulation cases [9]. Furthermore, in recent years, deep learning has shown promising performance in unsupervised anomaly detection, specifically autoencoders. However, it has been shown that using a hybrid model, combining autoencoders with basic machine learning methods, can achieve even higher performance. Therefore, this thesis proposes a hybrid approach that combines the autoencoder's ability to transform high-dimensional features into informative low-dimensional features and the one-class classifier's ability to distinguish anomalies from normal data.

1.2 Research Question

- How does machine learning methods perform in detecting spoofing strategies from limit order book data?
  - How does the proposed hybrid unsupervised anomaly detection method perform in detecting spoofing strategies?
  - How does the proposed method perform compared to three common unsupervised anomaly detection methods?
  - How does the proposed method perform compared to a rule-based benchmark algorithm?
  - Which features in limit order book data have the highest correlation to high performance in detecting spoofing strategies?

The performance of the machine learning methods is measured in three ways: using receiver operating characteristic (ROC) and precision-recall (PR) curves, analyzing the anomaly score distribution, and measuring the F1 score, precision, and recall at different thresholds for the anomaly score. These evaluation methods are performed on two datasets: a limited set of actual spoofing cases and a larger synthetic dataset.

1.3 Objectives

To answer the research question above, the following are the main objectives of the thesis:

- Create a dataset to be used for detecting spoofing by filtering and processing raw limit order book data.
- Train and evaluate the performance of unsupervised anomaly detection methods on this dataset.
- Create a synthetic dataset with instances of spoofing to be used during validation and evaluation.
- Create a rule-based algorithm that will be used as a benchmark method in the evaluation.
- Evaluate which features have the highest correlation with spoofing behavior.

1.4 Research Methodology

First, information about spoofing manipulation was collected through a literature search and discussions with domain experts. Based on this information, a method for transforming raw order book data into feature vectors was designed and improved iteratively. Second, information about unsupervised anomaly detection methods was researched, and a hybrid method was constructed based on the most promising methods for the spoofing detection problem. Finally, an experiment was conducted comparing the hybrid model to three other basic anomaly detection models.

1.5 Delimitations

The terms spoofing and layering are often used interchangeably. In this thesis, the definitions by Do et al. [8] are used: Spoofing is when one large order with no intention of execution is placed, and layering is when several such large orders are placed on several price levels. This thesis will focus on creating methods for detecting spoofing, thereby detecting many layering cases, but not all.

Furthermore, spoofing manipulators can take advantage of multiple venues to place market orders in one while canceling their limit orders in another [10]. This thesis focuses on detecting spoofing behavior in a single market using the same trading account.

Finally, spoofing can happen in different products in different financial markets. In this thesis, only stocks listed at Nasdaq First North Growth Market will be considered.

1.6 Structure of the Thesis

The structure of this thesis is as follows: Chapter 2 introduces the reader to the topic by describing the relevant technologies and the theory of spoofing. Chapter 3 describes the method used to answer the research question. Chapter 4 shows the results from the experiments. Chapter 5 analyzes and discusses the method and results. Chapter 6 concludes the thesis with the conclusion and future work.

Chapter 2

Background

2.1 Limit Order Book

The limit order book is a mechanism financial exchanges use to facilitate trading between buyers and sellers in financial markets. A *limit order* is a buy or sell order for an asset at a specified price and will remain in the limit order book until executed or canceled. The *limit order book* consists of all current limit orders, where the *bid side* consists of the buy limit orders and the *ask side* of the sell limit orders. The *best bid* is the price of the buy limit order with the highest price, and the *best ask* is the price of the sell limit order with the lowest price. The *spread* is the price difference between the best bid and the best ask, and the *mid-price* is the average of the best bid and best ask. Except for limit orders, there is also *market orders*, which is an order to sell or buy at the most advantageous price obtainable [11], and is mostly executed immediately. An example of a limit order book can be seen in Figure 2.1. [12]

When a market order is placed, it is checked against orders on the other side of the limit order book for execution. The selected limit orders for execution are matched according to a decided priority, which varies between different financial markets. On the Nasdaq Nordic markets, orders are matched after the following priority: price, internal[1], displayed[2], and time. [13]

Limit order book data comes in different levels of detail. Level 1 data provides the best bid/ask prices and volumes, level 2 data provides aggregated price and volume information for a larger number of price levels, and level 3 provides all information for the non-aggregated orders. In other words, level 3 gives the most details of the limit order book and level 1 the least. [14]

[1] A order placed by the same member (typically a trading or brokerage firm).
[2] Hidden limit orders have lower priority than displayed.

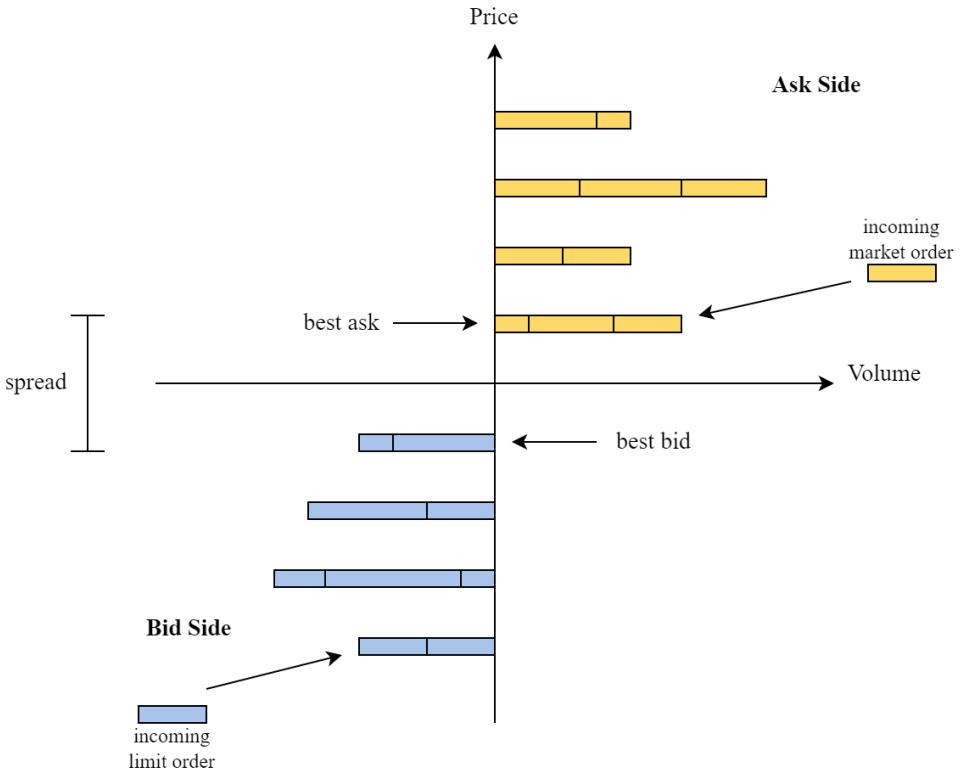


Figure 2.1: Limit Order Book

2.2 Market Manipulation

Market manipulation can be defined as encompassing all activities in which individuals or groups create artificial prices in various ways and present an unreal picture of market activity to mislead or defraud other investors and traders. This can be classified into three groups: information-based, action-based, and trade-based manipulation. Information-based manipulation means spreading incorrect or misleading information to change an asset's price in the desired direction. Action-based manipulation means taking actions beyond the trades, e.g., selling a company branch without informing the shareholders, to change an asset's actual or perceived value. Trade-based manipulation means trading and placing orders for an asset in a specific way with the aim of moving the price of an asset in the desired direction. Trade-based manipulation is more challenging to confront since the manipulation can seem legal in appearance, and no objective violation can be found. Some of the most common trade-based manipulation strategies are spoofing, pump-and-dump, and quote stuffing. [1]

2.3 Spoofing

2.3.1 Defining Spoofing

A spoofing strategy involves placing orders that are not intended to execute on one side of an asset's order book, with the purpose of influencing the prices by creating a false impression of trading interest. These orders are called *non-bona fide* orders. After the non-bona fide orders are placed, the manipulator executes a genuine order on the other side of the order book to take advantage of the market change. Lastly, the non-bona fide order is canceled. [8]

The variation of prosecuted spoofing cases is large. The manipulators range from day traders to large institutions, and the techniques used range from manual entries to automated algorithmic strategies. Therefore, to describe and define spoofing is a significant challenge. However, it is possible to define a ”typical” instance of spoofing as follows: [8]

1. The manipulator places a large order (non-bona fide) on one side of the market.
2. The manipulator places a relatively small order (genuine order) at or near the best bid/ask on the other side of the market. This step may happen before 1.
3. If the non-bona fide order has sufficient influence on the market, the genuine order will be executed.
4. Typically, the manipulator cancels the non-bona-fide order.

A spoofing instance is often repeated several times, and each instance may only yield a small profit. The manipulator may switch between spoofing the ask side and spoofing the bid side. [8]

The following parts of the spoofing strategy often have a large variation. [8]

- Manual entry or using algorithmic trading.
- One (spoofing) or several (layering) non-bona fide orders simultaneously.
- Position in the order book. Some manipulators place spoofing orders at the best bid/ask, and others place them lower in the order book to reduce the risk of executing the orders. However, generally, non-bona

fide orders are placed to have a low execution probability. This means either at price levels behind the best bid/ask, or at the back of a long queue of orders at best bid/ask.

- Repeating the spoofing or not.
- Switching between the sides in spoofing cycles or sticking to one.
- Duration of non-bona fide order. The non-bona fide order is mostly canceled shortly after the genuine trade. However, it is motivated for a manipulator to not cancel the non-bona fide order too quickly because of legal reasons. If the manipulator cancels the non-bona fide order shortly after a trade on the other side, it will likely attract regulatory attention. Additionally, a longer duration of the non-bona fide order would be easier for the manipulator to explain, e.g., say that s/he had a change of sentiment and decided to sell/buy instead and then forgot the non-bona fide order until later in the day. [15]

Even though there is a large variation between the prosecuted spoofing cases, there are also commonalities between them. The following are the main commonalities. [8]

- Non-bona fide orders are canceled before execution. However, in some cases, the orders will be inadvertently executed.
- Manipulators create substantial order imbalances in the limit order book.
- Manipulators execute trades on the opposite side of the non-bona fide orders.
- The non-hidden parts of genuine orders are typically smaller, individually or in aggregation, than the non-bona fide orders.
- Sometimes, orders on the genuine side are placed as hidden.
- Manipulators are often repeating the spoofing activities in a cyclical pattern.

2.3.2 Spoofing Environment

Spoofing strategies can happen in any security. However, some characteristics make spoofing more probable. Spoofing orders are more effective when

there is greater uncertainty about the value of a firm, which is why a higher return volatility makes spoofing more probable. Market cap is an important characteristic since a spoofing order would have a lower effect on a large-cap stock. Larger trading volumes can both mean that spoofing is more or less probable. On the one hand, spoofing stocks with large transaction volumes is difficult since the spoofing order must be very large to create a false impression of trading interest. On the other hand, larger trading volumes allow spoofing traders to hide their spoofing orders better, potentially attracting more spoofing investors. Additionally, small investors who are not well-informed tend to prefer low-price shares, indicating that stocks with lower prices might be more susceptible to spoofing activities. [15]

2.3.3 Detecting Spoofing

Detecting spoofing presents several challenges. Firstly, the vast amount of trading data, particularly from high-frequency trading, requires extensive analysis. Secondly, tracking the individuals or institutions behind trades is often difficult, as manipulators can use multiple venues and brokers. Thirdly, the vague definition of spoofing complicates the differentiation between legitimate and manipulative actions. Finally, the scarcity of prosecuted spoofing cases across exchanges indicates that detecting spoofing largely relies on unsupervised classification techniques. [10]

2.3.4 Possible Reasons for False Positives

As stated above, a spoofing strategy is carried out only through legal activities such as buying and selling securities. What makes spoofing illegal is the intent of the submitted orders. Since the definition of spoofing is vague and finding out the intent is challenging [10, 15], there is a high risk of many false positives. Some possible causes of false positives in the anomaly detection models could be market makers, quote stuffing, high-frequency trading, and legitimate trading on both sides.

2.3.4.1 Market Makers

A market maker is typically a member firm of an exchange that provides the market with liquidity by actively buying and selling stocks. A market maker profits from the difference in the spread, and by utilizing high-frequency trading, even a small spread can result in a large total profit. Just as in spoofing strategies, where orders are typically large to create false market

signals, market makers also place substantial orders and actively participate on both sides of the limit order book. However, market makers have lower cancellation rates than spoofing, execute trades on both sides, and place orders closer to the best bid/ask. [16]

2.3.4.2 Quote Stuffing

Quote stuffing is another trade-based manipulation strategy. The quote-stuffing strategy consists of flooding the order book with massive volumes of new orders, which are then canceled in rapid succession [9]. The purpose is to confuse the market and thereby create trading opportunities for the manipulator [17]. Quote stuffing can be differentiated from spoofing by its aggressively priced orders and substantially shorter cancellation time [9]. Therefore, the machine learning models in this thesis are expected to classify quote stuffing as normal trading.

2.3.4.3 Legitimate High-Frequency Trading

High-frequency trading (HFT) is a trading method that employs complex algorithms to execute many orders within seconds [18]. It is primarily used by banks, financial institutions, and institutional investors, aiming to capitalize on minor price discrepancies [18]. Thanks to the substantial volume of orders, even a small profit per trade can accumulate to significant earnings. With the rise of HFT, the occurrence of spoofing has increased [10]. However, most HFT strategies are legitimate [19] and differ from HFT spoofing cases by profiting from existing market conditions rather than creating artificial ones.

2.3.4.4 Legitimate Trading on Both Sides

Several instances exist where a trader's actions resemble typical spoofing, yet the large order is instead placed with a genuine intention to execute. In collaboration with domain experts, one such scenario was identified: A company is expected to release a report later in the day, leaving a trader uncertain whether stock prices will rise or fall. Nevertheless, the trader is interested in both buying and selling the stock at the right prices. Consequently, the trader places large orders at lower positions in the order book on both the ask and the bid sides. As the report is released and price movements occur, one of the orders will be executed while the trader cancels the other. If no price movement results from the released report, both orders will be canceled. This situation can appear similar to a spoofing attempt, with

the crucial difference being the trader's initial intention for the orders to be executed. This thesis does not aim to identify such sequences accurately. Instead, the ML methods aim to flag these as potential spoofing instances, leaving it to the human analyst to determine whether they should be reported as spoofing.

2.4 Machine Learning Concepts

2.4.1 Generalization

Generalization refers to how well a machine learning model predicts unseen data. It is crucial for a machine learning model to achieve high generalization, especially in fraud detection, where a model must be able to detect fraudulent activities it has not explicitly been trained on. Two of the challenges related to generalization are overfitting and underfitting. Overfitting occurs when the model learns the training data too closely, including the noise and outliers. Underfitting is when the model is too simple, resulting in the inability to predict the training data or generalize new data. These challenges are illustrated in Figure 2.2. [20]

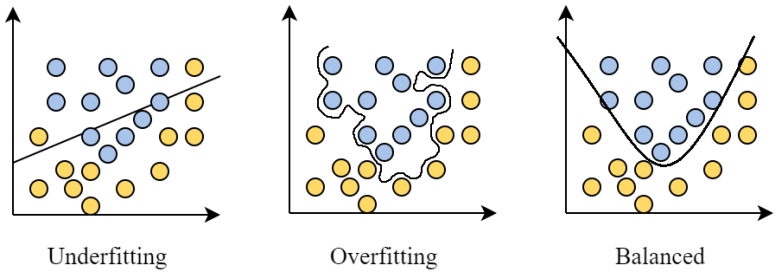


Figure 2.2: Underfitting and overfitting.

Techniques such as regularization and cross-validation can be used to improve the generalization of the models. Some commonly used regularization techniques are adding noise to the input data and implementing early stopping. Early stopping is a technique where the model's training is stopped when certain conditions are met. This technique can prevent overfitting by stopping the training when the performance on the validation dataset decreases, as illustrated in Figure 2.3. [21]

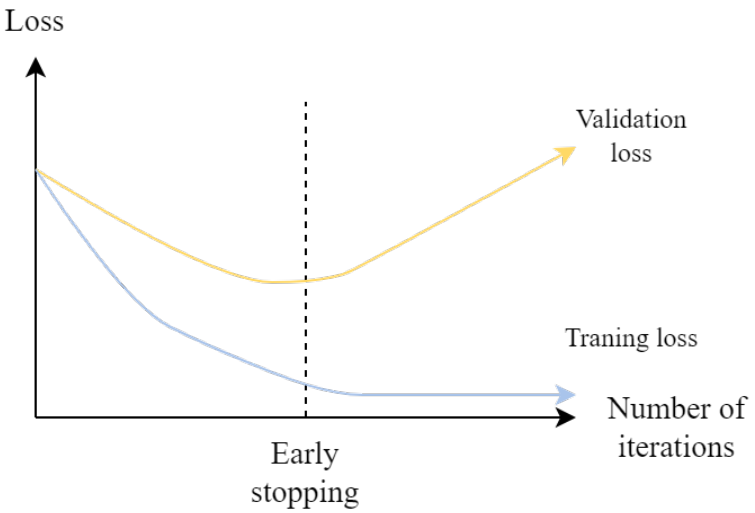


Figure 2.3: Early stopping.

One common technique of cross-validation is k-fold cross-validation, which is a method where the data is randomly split into k separate sets of approximately equal size. One set is used for validation, and the remaining $k - 1$ sets are used for training the model. Another set is then chosen for validation, and the remaining sets are used for training. This process is repeated until all sets have been used for validation. Finally, the average of the model's performance can be found and used to find the best parameters for high generalization. See figure 2.4 for an overview of 5-fold cross-validation, where blue indicates training data and orange validation data. [22]

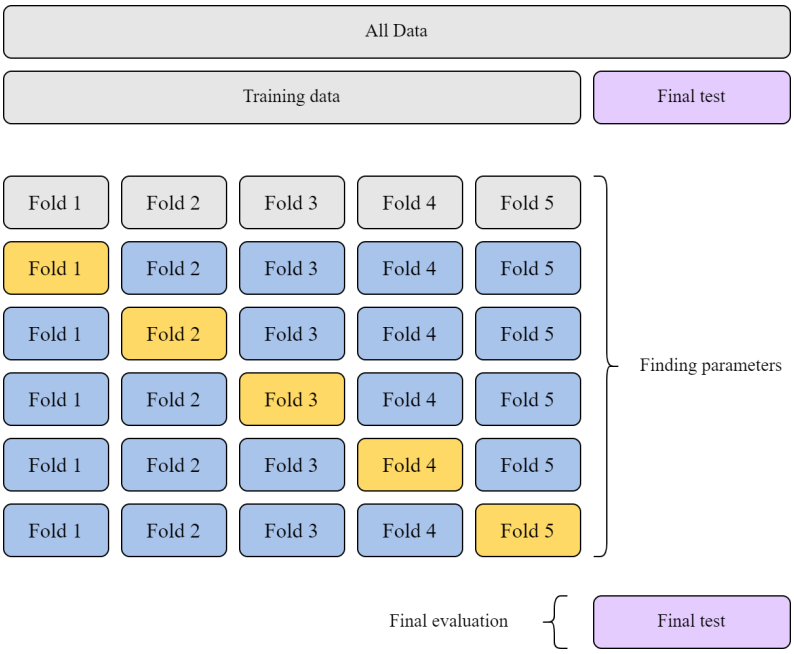


Figure 2.4: 5-fold cross-validation.

2.4.2 Features

Features are the individual variables that together describe a data instance. In other words, the features are input to a machine learning model. Features are also called variables or attributes; the number of features is known as dimensions. [23]

Features greatly impact the quality of the machine learning model and can be improved by using feature selection or feature engineering. Feature selection is the process of identifying the important features from the dataset, with the purpose of improving learning performance, preventing overfitting, and reducing computational costs. Two of the most common feature selection methods in an unsupervised setting are using domain knowledge and removing correlated features. [24]

Feature engineering is the process of transforming the features into the final set of insightful features used as input to the model. Methods included, but not limited to, in feature engineering are normalization and standardization. [25]

2.5 Unsupervised Anomaly Detection

Anomaly detection, also called outlier detection, can be divided into three categories: supervised, semi-supervised, and unsupervised. Most previous work follows an unsupervised approach. The unsupervised approach hinges on the assumption that anomalies in the data are either rare or absent and that models are learning usual system behaviors well enough to distinguish outliers out-of-sample. The main factor driving the use of unsupervised models is that available fraud cases are very limited, and supervised learning would require a substantial effort to generate a usable data set. Furthermore, supervised models can only detect known patterns in the data, rendering them useless in unveiling emerging trade-based manipulations or when market data's distribution inevitably drifts. These limitations support the adoption of unsupervised learning methods. [6]

2.5.1 Anomalies

Anomalies can be divided into point, contextual, and collective anomalies. The point anomaly is the simplest of the three types and is a single instance that is anomalous with respect to the rest of the data. A contextual anomaly is only anomalous in a specific context, not when considering the dataset as a whole. A collective anomaly is a set of related instances that behave differently than the rest of the dataset, although the individual instances should be considered normal. The difference between the three types of anomalies can be seen in Figure 2.5. [26]

2.5.2 Anomaly Detection Methods

This section introduces three common unsupervised machine-learning techniques applied in this thesis for anomaly detection: Autoencoder, Isolation Forest (IForest), and One-Class Support Vector Machine (OC-SVM). Each method offers distinct strategies for identifying anomalies in datasets without labeled examples. Autoencoders detect anomalies through data reconstruction errors, IForest efficiently isolates outliers, and OC-SVM distinguishes anomalies based on their distance from a decision boundary.

2.5.2.1 Autoencoder

An autoencoder is a feed-forward neural network (NN) used to learn efficient data encodings. An autoencoder consists of an encoder and a decoder,

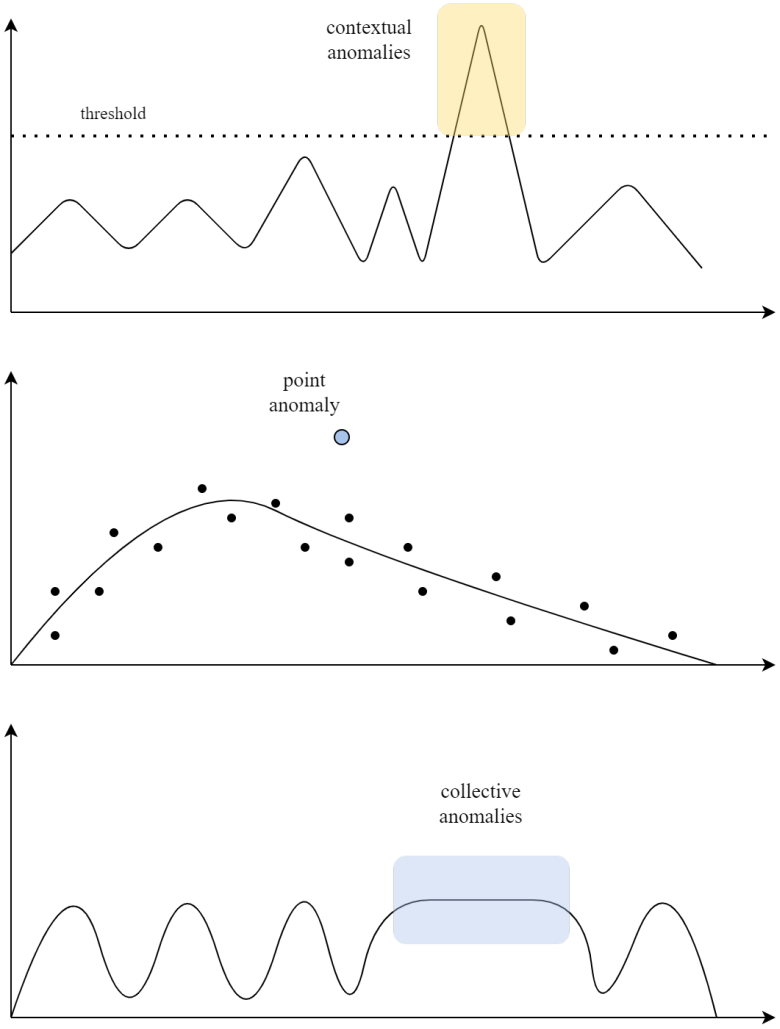


Figure 2.5: Comparing point, contextual, and collective anomalies.

consisting of several hidden layers. An example of a basic autoencoder can be seen in figure 2.6. The encoder transforms the high-dimensional input data into a low-dimensional latent space, and the decoder aims to transform this new representation into the original form. Since the latent space has lower dimensions, the autoencoder must learn to recognize structures and patterns in the data. The goal of an autoencoder is, in other words, to transform high-dimensional data into informative low-dimensional encoding. [26]

Autoencoders can be used in anomaly detection by using the reconstruction error as an anomaly score. This approach is based on the assumption that anomalous data is more difficult to reconstruct than normal data due to its limited exposure to the autoencoder during the training phase [26].

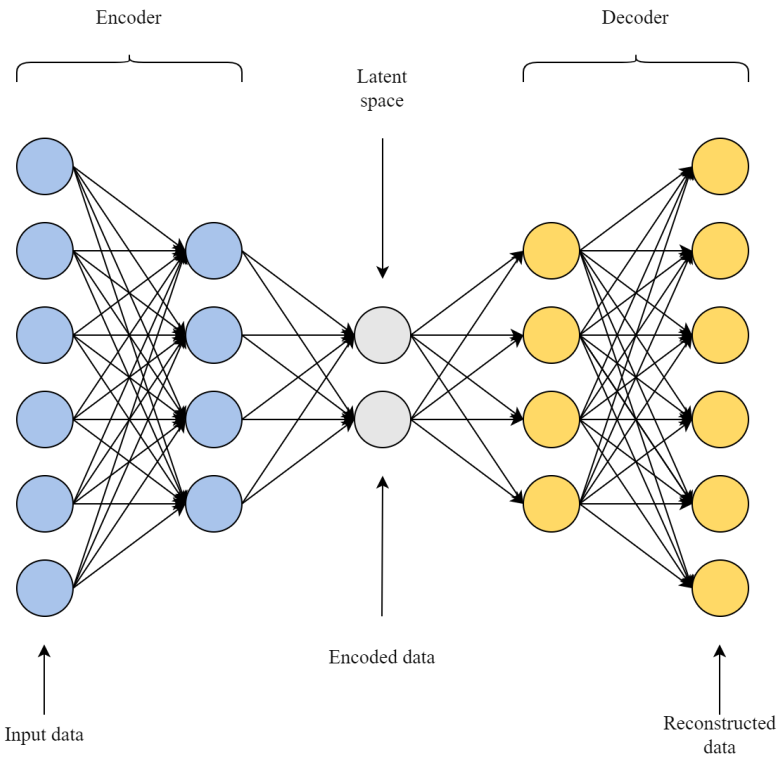


Figure 2.6: Basic Autoencoder

2.5.2.2 Isolation Forest

Isolation Forest (IForest) is another widely used method in anomaly detection, and it is different from other methods in that it starts from outliers rather than normal data. The core idea is that it should be easy to isolate anomalies based on their unique characteristics. [27]

IForest is based on a tree-based structure and functions by randomly selecting a feature and subsequently randomly selecting a split value between the maximum and minimum values of the chosen feature. The algorithm proceeds by partitioning the data recursively, and the number of partitions required to isolate a point is employed as a measure of anomaly. The algorithm is based on the assumption that fewer steps are required to isolate anomalies, meaning that the lower the anomaly score, the more abnormal a given data point is assumed to be. [28]

In an IForest, several trees are created with a random data set sample. The anomaly score for a data instance is based on the average path length from the root node to the terminal node across all trees where the instance is isolated. The equation for the anomaly scores for an instance x in a forest of n trees is

defined as in (2.1), where $E(h(x))$ is the average path length of x over all trees in the forest, and $c(n)$ serves as a normalization factor. Given the exponential function, the final anomaly score will be between 0 and 1. [29]

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{2.1}$$

2.5.2.3 One-Class Support Vector Machine

The core idea of the support vector machine method is that data mapped to a higher-dimensional feature space can be more easily separated and classified. The most basic example of a support vector machine is linear classification, where the data is divided into two groups with a line or a plane, referred to as maximum-margin hyperplane (MMH). The hyperplane is chosen to maximize the distance to the nearest points. However, in most cases, the data cannot be separated by simple linear classification. This is solved by transforming the data into a higher-dimensional feature space and finding the MMH in that space instead. [30]

The One-Class Support Vector Machine is based on the same idea but with only one class. The hyperplane is set to maximize the distance between the dataset and the origin. All data points closer to the origin than the hyperplane will be classified as outliers, and the distance from the hyperplane can be seen as the anomaly score. [30]

2.5.2.4 Hybrid Methods

One challenge with using autoencoders as anomaly detectors is that they, in many situations, generalize too well and thereby reconstruct anomalies with low error [31]. A potential solution is a hybrid model approach, which has demonstrated notable efficacy in the domain of unsupervised anomaly detection in recent years. Primarily, these approaches use an autoencoder for feature extraction purposes, utilizing the representations from the hidden layers as inputs for traditional anomaly identification methods like OC-SVM [32]. A more thorough explanation of these hybrid models is provided in section 2.6.

2.5.3 Feature Importance

Feature importance involves assigning a score to each feature to assess its contribution to the final classification. This thesis has chosen two recently

developed approaches for feature importance: one for use with autoencoders and another for isolation forests.

Gomes et al. [33] propose a feature importance method for autoencoders based on the reconstruction error; however, instead of considering the reconstruction error per data instance, it considers the reconstruction error per feature. The methodology comprises three main steps. The first step involves predicting the test data set using the trained autoencoder and identifying those with reconstruction errors exceeding a predetermined threshold. This approach is grounded in the assumption that higher reconstruction errors signify a greater probability of an instance being fraudulent. In the second step, the feature reconstruction error $RE_{feature}$ is calculated for these selected test instances for all features. The final step extends the $RE_{feature}$ calculation to the test instances excluded in the first step, aiming to disregard features that consistently show high reconstruction errors in both the second and third steps. The features exhibiting high errors in the second step yet lower errors in the third step are regarded as the most important in terms of their contribution to the autoencoder's anomaly detection performance.

Carletti et al. [34] propose a method for defining feature importance scores for Isolation Forests called Depth-based Isolation Forest Feature Importance (DIFFI). DIFFI calculates feature importance by observing how individual features contribute to isolating anomalies at small tree depths and their ability to produce an imbalance between inliers and outliers. The method starts with calculating Cumulative Feature Importances (CFIs) for all features. CFIs are calculated by evaluating each feature's role in isolating anomalies, including the depth at which data instances are isolated and the produced imbalance at the split node. The next step is deriving the Global Feature Importances (GFIs) from the CFIs by normalizing them. The normalization is performed to ensure that the feature importance scores are not affected by the frequency of feature usage in tree splits, which can vary greatly due to the random feature selection in Isolation Forest. The GFIs represent the most important features in the isolation of anomalies.

2.6 Related work

2.6.1 Detecting Spoofing

Do et al. [8] develop empirical techniques for detecting spoofing by analyzing a global sample of prosecuted spoofing cases. Most notably, they identify the most significant indicators of spoofing as unbalanced order book quotes, high order activity, abnormal order cancellations, and cyclical patterns in market depth and cancellations. Additionally, they created a random forest and boosted tree classification models based on these indicators, achieving AUC ROC scores of 0.96 and 0.97, respectively. However, Do et al. had access to many spoofing cases, making their problem a supervised classification challenge. Typically, access to spoofing cases is limited, rendering this work and most others as unsupervised problems. In this thesis, the identified indicators will be considered in the feature selection phase, but the machine learning methods will differ.

Poutré et al. [6] introduced an innovative hybrid anomaly detection model using a Transformer autoencoder combined with a one-class support vector machine (OC-SVM). This model performs state-of-the-art fraud detection, focusing on quote stuffing, pump-and-dump, and layering as trade-based manipulation types. However, the method performs better when detecting quote stuffing and pump-and-dump than layering. Additionally, the study only uses level 1 data, making it challenging to detect layering/spoofing. Finally, the models are only evaluated on synthetic data, making the results less reliable. This thesis takes inspiration from the hybrid approach but uses level 3 data instead. Additionally, by only focusing on detecting spoofing, the data can be more filtered, and the features can be better adapted to learn how to detect spoofing.

Tao et al. [10] assesses quantitatively spoofing behavior in high-frequency trading. They describe the optimal spoofing strategy and its impact on the imbalance in the limit order book. Derived from these findings, they propose a quantification instrument to monitor real-time spoofing behavior on the market using a conditional Wasserstein distance. The main shortcoming of the detection method is that it only includes the limit order book's imbalance as features, where they define the following: a spoofing behavior creates an imbalance in the market, and after the genuine market order has been executed, the imbalance should be back to its equilibrium. More features can be included using machine learning methods, enabling learning patterns of spoofing that have not been predefined earlier to be identified.

Zhai et al. [9] propose a static model to detect spoofing and quote stuffing. This anomaly detection model is designed to learn what is normal to identify abnormalities effectively. Zhai et al. demonstrate that the combination of order price, volume, and the interval between submission and execution/cancellation indicates legitimate trading behavior. This led to creating a transformed time series feature set for the static model based solely on price, volume, and time. The methods employed were the k-nearest neighbor and one-class support vector machine, achieving impressive results with a precision of 1 and a recall of 0.9933. However, the test set's manipulative case data only included one minute of data from July 10, 2013, for the Apple stock, during which a High-Frequency Trader engaged in spoofing. The quoting and trading activity during this minute was unusually high, surpassing even that during the FOMC news release at 14:00 on the same day, known to be one of the market's most volatile periods. The remaining anomalies in the test set were synthetically generated. Thus, achieving a precision of 1 is not particularly surprising. This thesis differs from their work in that it aims to detect all types of spoofing, including those not performed by high-frequency traders. This necessitates considering additional features beyond price, volume, and time, especially to maintain a low false positive rate.

2.6.2 Unsupervised Anomaly Detection

A popular approach in unsupervised anomaly detection has been to use the autoencoder's reconstruction error. However, several recent studies show that a hybrid approach, where the autoencoder reduces the dimensions and a one-class classifier is trained on the latent space, achieves higher performance. Iglesias et al. [35] create a hybrid approach using a convolutional autoencoder (CA) and a support vector machine (SVM) on industrial textures data, which outperforms using CA and SVM separately. Zhang et al. [36] proposes a hybrid approach combining an autoencoder and a sparse weighted least squares support vector machine, AE-SWLS-SVM. The autoencoder is used to extract low-dimensional features of high-dimensional data, thereby reducing data dimensionality and the complexity of the search space. In the latent space, the sparse weighted least squares support vector machine separates normal and abnormal features. The results show that the proposed method is superior to competitors in terms of anomaly detection ability of high-dimensional data.

Additionally, there are other approaches to creating a hybrid model. Jeragh et al. [37] introduce a new unsupervised learning model based on combining

an autoencoder with a one-class support vector machine (OC-SVM). In this model, an input is fed into an autoencoder, and the reconstruction error is produced and passed on to an OC-SVM to determine whether a transaction is fraudulent. This resulting model can be viewed as a deep autoencoder equipped with an automatic thresholding mechanism, which identifies a soft threshold (the OC-SVM's hyperplane) using unsupervised learning, in contrast to setting an arbitrary hard threshold when only using an autoencoder. Mohammed et al. [28] propose two methods to combine an autoencoder with an isolation forest. The first method is the typical hybrid approach, where the isolation forest algorithm is trained on the encoded data. The second method involves training the isolation forest algorithm on the decoded data. Both methods achieved higher performance than using only the autoencoder or the isolation forest separately as an anomaly detection method.

Chapter 3

Method

3.1 Data Processing

3.1.1 Data

To limit the scope of this thesis, data from only a few stocks was selected. Since spoofing is more common in stocks with higher return volatility, lower market capitalization, lower price, and lower managerial transparency, it was decided only to collect data from stocks listed at Nasdaq First North Growth Market, which both have lower market capitalization and lower managerial transparency than the stocks listed at the Stockholm Stock Exchange's main market. The chosen stocks had a volatility of at least 50%, and a price lower than 10 SEK per stock. Additionally, to ensure that a significant amount of data could be collected for each stock, it was decided that every stock should have an average daily trading volume of at least 100 000 SEK. 20+ stocks fulfilled these criteria. For the training dataset, 6 months of trading data was collected for these stocks.

The dataset consists of level 3 limit order book data, including non-aggregated bids and asks submitted by individual market participants. Each row in the dataset represents a single event within the order book, categorized into one of four types: Enter, Delete, Amend, and Trade. The attributes characterizing these events include but are not limited to, price, volume, time, client, and position. The position attribute specifies the order's rank in the order book at the time of the event, determined by both price and entry time. Consequently, among two bid orders placed at the best bid price level, the one entered first is assigned position 0, while the subsequent order is assigned position 1.

3.1.2 Defining the Classification Problem

This thesis focuses on detecting spoofing activities using limit order book data. Given this data's unlabeled nature and the rarity of spoofing incidents, this task falls under unsupervised anomaly detection.

Order book data, which is inherently multivariate time series information, allows for classifying anomalies into point, contextual, and collective types. Spoofing involves sequences of legitimate actions - placing and canceling trade orders - that, in isolation, appear normal within the broader context of limited order book data. However, when considered as a set, these actions outline a spoofing strategy, setting them apart from standard trading behaviors. Thus, the challenge of identifying spoofing within raw order book data can be framed as detecting a collective anomaly.

Alternatively, this problem can be approached as a point anomaly detection issue by converting raw order book data sequences into aggregated feature vectors. This thesis adopts this latter approach for several reasons: it simplifies the detection task, offers a wider selection of machine learning methodologies, and enhances computational efficiency.

The transformation of raw limit order book data into aggregated feature vectors is achieved through a methodology based on a theoretical understanding of spoofing and insights from domain experts. This process is further elaborated upon in the subsequent section.

3.1.3 Transforming Raw Data to Aggregated Feature Vectors

Together with domain experts, it was decided that spoofing is most likely to happen if the same client fulfills the following two criteria:

1. A large passive order has been placed on one side of the order book.
    - Large is defined as the volume of the order / total volume on the same side of the order book > fixed threshold value depending on stock.
    - Passive is defined as an order not placed on position 0, with the motivation that a non-bona fide order has a low execution probability.
2. A trade happens on the other side of the order book.

Given the machine learning models aim to uncover patterns in the data that are not discernible to humans, it is essential to exclude data that does not meet the above criteria. Moreover, by filtering out certain data, the time required to aggregate the data into the final feature vectors can be reduced. In collaboration with domain experts, it was thus decided that the instances to be trained on and classified by the ML methods would be modeled as follows:

1. Find all placed orders that have a large volume and where the same client is active on the other side afterward. Additionally, market makers are filtered out to reduce the false positives.

2. For each selected order, define the sequence as: start time = entry time of selected order, end time = time of first trade on the other side.

3. If a client has overlapping sequences, combine them into one.

4. For each sequence, aggregate information that describes the client's trading behavior during that sequence.

This approach generates a 1xN-sized feature vector for each sequence, describing the client's trading behavior within the specified time sequence. Consequently, the task for the ML models is to classify instances of size 1xN as either normal or anomalous, where an anomaly indicates potential spoofing.

3.1.4 Feature Selection

Selecting features is an important step in any machine-learning project. Yet, in an unsupervised setting, where the ground truth is not predetermined, identifying the important features of the specified classification problem poses a significant challenge. Consequently, the feature selection process has been iterative. The final set consisted of the features as seen in 3.1.

Index	Feature
1	cancellation_rate
2	max_imbalance_active
3	client_imbalance_at_trade
4	trades_on_genuine
5	enters_on_nbf
6	start_position
7	time_enter_to_trade
8	volume_share
9	hidden_nbf
10	effect_on_price
11	traded_money

Table 3.1: Final set of features.

1. cancellation_rate

    - **Description**: The portion of the client's volume entered on the non-bona fide side that was canceled.
    - **Explanation**: Non-bona fide orders are mostly canceled before execution, meaning that spoofing instances should have a high cancellation rate.

2. max_imbalance_active

    - **Description**: The maximum order book imbalance during the sequence among the top 10 positions.
    - **Explanation**: The non-bona fide orders should create a substantial order book imbalance. The maximum was chosen since the biggest non-bona fide orders may not be at the start of the sequence. The top 10 positions were chosen instead of the complete order book since it was decided that below position 10, an imbalance would not be able to create a false impression of trading interest.

3. client_imbalance_at_trade

    - **Description**: The client's order imbalance, including orders entered before the sequence and still in the order book, at the time of the trade on the genuine side.

- **Explanation**: Typical for spoofing, but not a requirement, is that the client has a substantially larger volume on the non-bona fide side than on the genuine side when a genuine trade occurs, meaning a high client_imbalance_at_trade. This feature is further motivated by the fact that it considers all client orders, including those entered before the sequence and still in the order book.

4. trades_on_genuine

   - **Description**: The portion of the client's total traded volume during the sequence made on the genuine side.
   - **Explanation**: This feature was mainly added to reduce the amount of false positives. A spoofing strategy should have no or few trades on the non-bona fide side, while legitimate strategies such as high-frequency trading are often active on both sides.

5. enters_on_nbf

   - **Description**: The portion of the client's total entered volume during the sequence made on the non-bona fide side.
   - **Explanation**: This feature was also added to reduce the amount of false positives. In several of the false positives found, the client was more active with larger orders placed on the genuine side than the non-bona fide during the sequence.

6. start_position

   - **Description**: The client's average start position of the placed orders on the non-bona fide side during the sequence.
   - **Explanation**: The position of non-bona fide orders is chosen to have low execution probability but is sufficiently close to the best bid/ask to influence prices. Therefore, it is important to look at the position to detect spoofing.

7. time_enter_to_trade

   - **Description**: The time in seconds from the last enter on the non-bona fide side during the sequence to the trade on the genuine side.
   - **Explanation**: This feature was added to reduce false positives caused by either a very short time difference between order entry and trade, where the non-bona fide orders had not yet impacted the market, or by a too large time difference.

8. volume_share

    - **Description**: The client's total entered volume during the sequence on the non-bona fide side, divided by the non-bona fide side's total volume before the start of the sequence.
    - **Explanation**: The non-bona fide orders should be a significantly large portion of the non-bona fide side's total volume at the start of the sequence, to be able to create a false impression of trading interest.

9. hidden_nbf

    - **Description**: The portion of the client's entered volume on the non-bona fide side that is hidden.
    - **Explanation**: Non-bona fide orders are not hidden since they will not be able to influence the market if they are hidden.

10. effect_on_price

    - **Description**: The price movement of the best bid/ask on the non-bona fide side during the sequence, divided by the best bid/ask on the non-bona fide side at the start of the sequence.
    - **Explanation**: Spoofing enables trading on the other side of the order book at favorable prices, which requires a movement of the best price on the non-bona fide side. The price change is divided by the initial price to improve the generalizability of the models.

11. traded_money

    - **Description**: The total value traded on the genuine side: volume x price.
    - **Explanation**: If the final trade is larger, the significance of the spoofing instance is larger. Additionally, considering the money instead of the volume should make the ML models more generalizable between stocks.

3.1.5 Feature Engineering

Normalization and standardization were tested and evaluated as part of feature engineering. Normalization is a technique for transforming features to a common scale, specifically a range between 0 and 1. This technique is also known as min-max scaling, and the equation used can be seen in 3.1, where x is a feature vector from the training dataset. [38]

$$x_{\text{normalized}} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

If the test set has values larger than $\max(x)$ or smaller than $\min(x)$, the resulting normalized values will be outside the range [0,1]. This issue was addressed by accepting values outside this range as potential anomalies.

Standardization is a technique for centering the features around the mean with a unit standard deviation. The equation used can be seen in 3.2, where μ is the mean of the feature values in the training dataset and σ is the standard deviation of the feature values in the training dataset. [38]

$$x_{\text{standardized}} = \frac{x - \mu}{\sigma} \tag{3.2}$$

Both normalization and standardization were tested initially when evaluating the models, and normalization achieved higher results. Therefore, normalization has been used on all data during the rest of the training and evaluation.

3.1.6 Training, Validation, and Test sets

After performing the previously mentioned filtering process, the final dataset comprised 25,252 rows. Of these, 20,201 were allocated for training, 2525 for validation, and another 2525 for the final test set.

In this thesis, a small number of spoofing instances were available. To address this limitation, a synthetic dataset was generated. The real spoofing instances were randomly divided into two equally large groups. The first one, called *validation real*, would be used to create the *validation synthetic* dataset. The second group, *final real*, would only be used in the end during the final evaluation of the models and also to create the *final synthetic* dataset.

Two methods were developed to create the synthetic dataset: the min-max method and the distribution method.

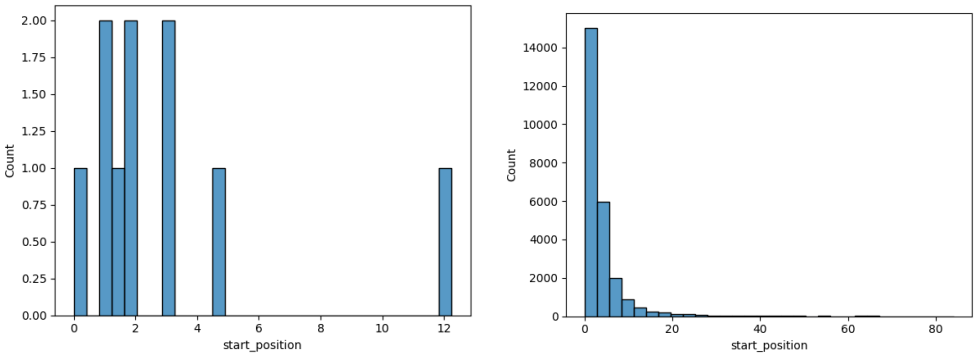
The min-max method worked as follows:

- The filtering and feature selection methods described above were implemented on the *validation real* dataset.
- For each feature, the min and max were calculated.
- 500 instances of synthetic data were created by randomizing a value for every feature within the range.

However, during the training and evaluation of the models, it was realized that this method created anomalous combinations of feature values that did not represent the *validation real* dataset. Therefore, it was often easier for the models to detect the synthetic data than the *validation real* instances. This resulted in the creation of the distribution method, which tried to imitate the feature distribution in the *validation real* dataset instead of looking at min and max.

The distribution method worked as follows:

- The filtering and feature selection methods described above were implemented on the *validation real* dataset.
- For each feature, the distribution was visualized in a histogram diagram. An example of the feature *start_position* can be seen in figure 3.1.
- From looking at the diagram, it was decided if the distribution was most similar to an exponential or normal distribution.
- For each feature, an exponential or normal distribution with 500 instances was calculated with the same min and max values as the *validation real* data. An example of the feature *start_position* can be seen in figure 3.2.



(a) Validation real dataset (b) Training dataset

Figure 3.1: Distribution method

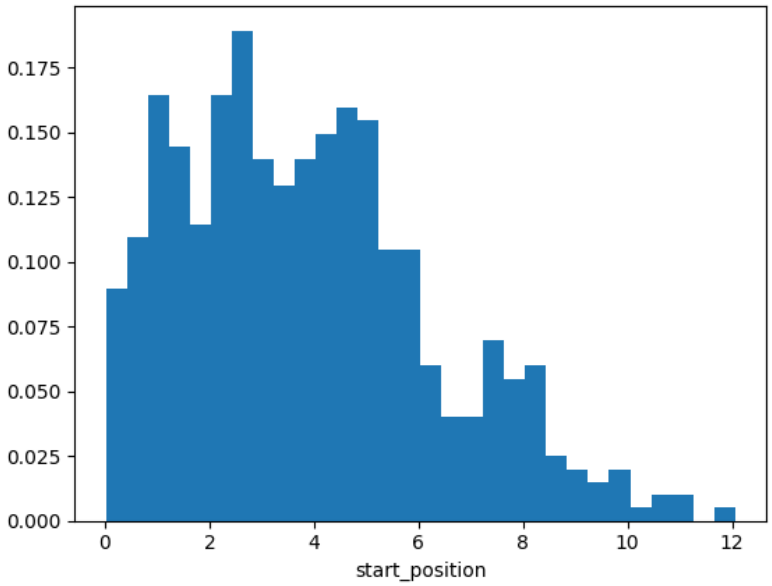
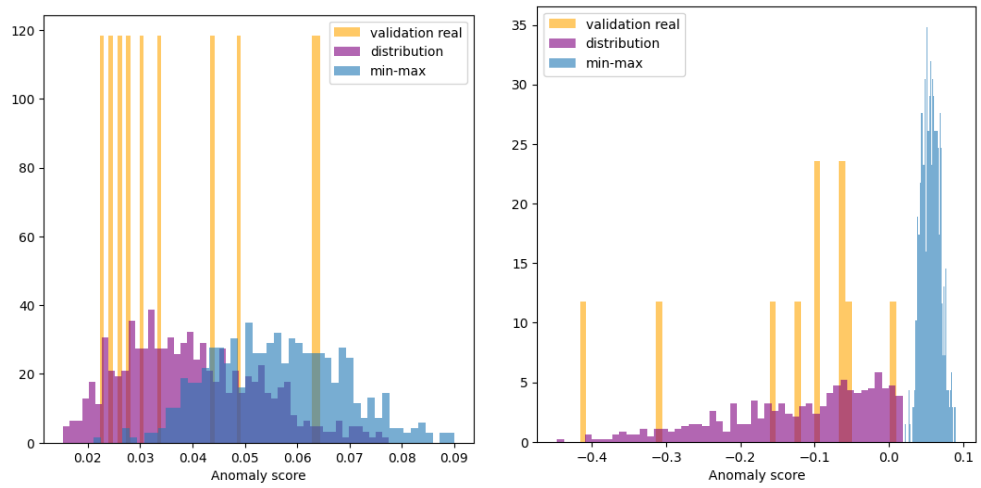


Figure 3.2: Synthetic dataset - start_position

The two synthetic data tests were compared using the trained models, and the results from the autoencoder and the OC-SVM can be seen in figure 3.3. From the anomaly score distribution, it can be said that the distribution synthetic dataset performs better at representing the *validation real* dataset than the min-max synthetic dataset. Therefore, it was decided to use only the distribution synthetic dataset for the rest of the project.



(a) Autoencoder anomaly score.

(b) OC-SVM anomaly score.

Figure 3.3: Comparison of datasets derived from the min-max and distribution methods.

The datasets used in the experiments can be seen in figure 3.4, where *n_real* is the number of real instances in each dataset and *n_normal* is the number of normal instances added to the real datasets. As can be seen, it is the same amount of data instances in the *validation real* and the *final real* dataset. Since there is a lower number of positive instances in the *validation real* and *final real* datasets than the synthetic, a lower number of normal data was added to the real datasets than to the synthetic ones. This was primarily done to simplify the analysis of the precision-recall curve, as a high number of false positives significantly reduces the precision.

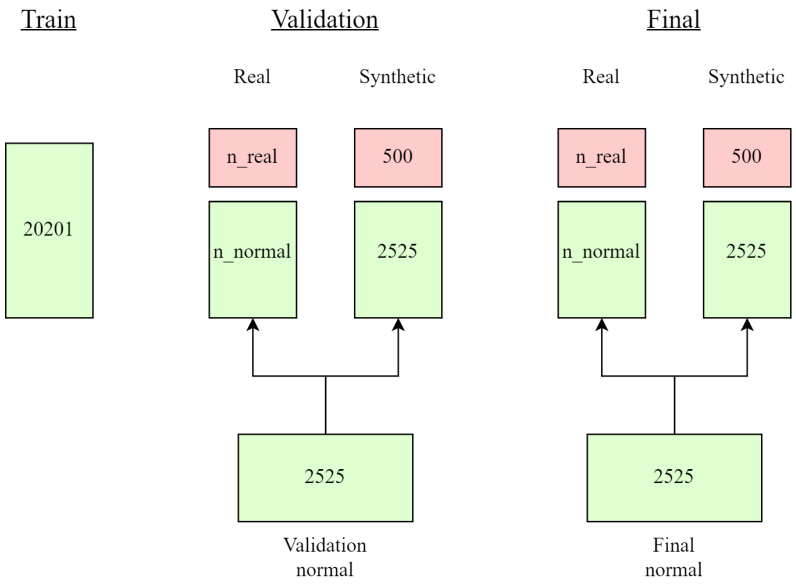


Figure 3.4: Datasets used in experiments, green indicating negative/normal instances and red indicating positive/spoofing instances.

3.2 Training

3.2.1 Model Selection

In this thesis, the following five models were evaluated:

1. Hybrid model consisting of an autoencoder and an Isolation Forest (AE + IForest)
2. Hybrid model consisting of an autoencoder and one-class support vector machine (AE + OC-SVM)
3. Autoencoder using the reconstruction error as anomaly score (AE)
4. Isolation Forest (IForest)
5. One-class support vector machine (OC-SVM)

IForest and OC-SVM were chosen as one-class anomaly detectors since they are both among the most well-established unsupervised anomaly detection methods, with different strengths. Isolation Forest has proven effectiveness in high-dimensional datasets and has low computational complexity [34]. OC-SVM is sensitive to outliers and is best suited for novelty

detection, that is when the training dataset is not contaminated by outliers [39]. Finally, using only an autoencoder as an anomaly detector is a common method and was added as a method to be evaluated.

3.2.2 Model Implementation

The code for this thesis was developed using Python 3.9.12. The models were implemented mainly using the following libraries: the Tensorflow Keras library [40] and the Sklearn library [41]. The IForest and OC-SVM models were constructed using standard functions available in Sklearn, requiring only parameter tuning, which will be discussed further. In contrast, the architectures of the autoencoder and the hybrid models require a more detailed explanation, which will be provided in this section.

As an anomaly score, the reconstruction error was used for the autoencoder, where a higher reconstruction error meant more anomalous. The reconstruction error for one data instance was calculated as in 3.3, where m is the number of features in each instance, y_{ij} is the original value of feature j in data instance i, and $\hat{y}_{ij}$ is the predicted value of feature j in data instance i [42]. For the rest of the models, Sklearn's built-in functions were used to get the anomaly scores for the data instances. For these models, a lower anomaly score meant more anomalous.

$$\text{RE} = \frac{1}{m}\sum_{j=1}^{m}(y_{ij} - \hat{y}_{ij}) \tag{3.3}$$

The architecture of the autoencoder can be seen in table 3.3. It was experimented with different numbers of layers and nodes, and this was the one that gave the best performance on the validation datasets. Mohammed et al. [28] used the ReLU activation for all layers, Adam as the optimization algorithm, and the Mean Squared Error (MSE) for their autoencoder in their proposed hybrid model. Since they achieved high performance, the same parameters were chosen for the autoencoder in this thesis. However, in the last output layer, Sigmoid is often used as an activation function instead to constrain the output to the interval (0,1) [42]. Both ReLU and Sigmoid were experimented with as activation functions in the last layer, and Sigmoid performed the highest.

MSE quantifies the average squared reconstruction error over all instances in the data set and is defined in 3.4, where n is the number of data instances in the training data set and is used to train the final models in this thesis [43]. Finally, early stopping was used to prevent overfitting.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{m}\sum_{j=1}^{m}(y_{ij} - \hat{y}_{ij})^2\right) \tag{3.4}$$

Table 3.2: Architecture of the Autoencoder

Layer Type	Number of Nodes	Activation function
Input Layer	11	ReLU
Hidden Layer 1	10	ReLU
Hidden Layer 2	8	ReLU
Latent Layer	6	ReLU
Hidden Layer 3	8	ReLU
Hidden Layer 4	10	ReLU
Output Layer	11	Sigmoid

Table 3.3: Autoencoder's architecture

The architecture of the hybrid models can be seen in figure 3.5. After the autoencoder has been trained, the training data is encoded to its latent space using the encoder. In this format, the data is sent further to the one-class classifier. In other words, the latent space training data is the new training data for the one-class classifier. After training the one-class classifier, the encoded test data will be classified as spoofing or normal.

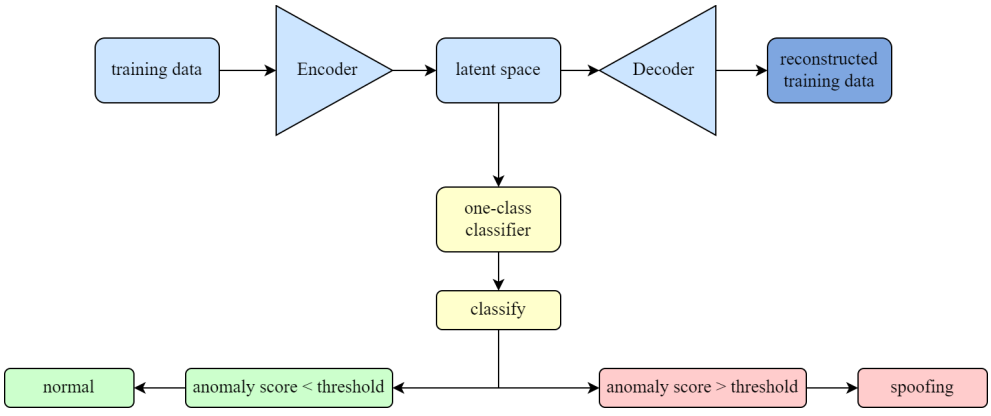


Figure 3.5: Architecture of hybrid model.

3.3 Evaluation of Models

During validation, the models were primarily evaluated by analyzing ROC-curves and precision-recall curves. Additionally, the distribution of anomaly scores was visualized using histograms to enable further insights. Two key aspects were considered when analyzing these distributions. Firstly, it was analyzed if the distributions of the training data and the normal data were similar, which helped identify any potential overfitting or underfitting by the model. Secondly, the effectiveness of the models in distinguishing between normal and spoofing instances was assessed, ensuring that the models could effectively separate the two types of instances.

In the final evaluation phase, the models were assessed using three primary methods: first, analyzing the distribution of anomaly scores; second, examining the ROC curves and precision-recall curves; and third, calculating the F1 score, precision, and recall at various anomaly score thresholds. The selected thresholds included the 70th, 80th, and 90th percentiles, as well as the threshold where all *final real* instances were detected, achieving a recall of 1.

3.3.1 Evaluation Metrics

The most common evaluation metric for classification models is accuracy, defined as in equation 3.5, where TP = number of true positives, TN = number of true negatives, FP = number of false positives, and FN = number of false negatives. These abbreviations are also used in the rest of the equations below. However, accuracy does not give a full picture of the performance on imbalanced data sets; for example, a model predicting all anomalies as false can achieve a high accuracy if the number of anomalies is small. Therefore, accuracy was not used to evaluate the models. [44]

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3.5}$$

Other common methods when evaluating anomaly detection methods are precision, recall, and F1 score. Precision answers the question about what proportion of identified anomalies are true positives and can be seen in equation 3.6. Recall answers the question about what proportion of positives were identified and can be seen in equation 3.7. The F1 score is a metric describing the model's overall performance by calculating the harmonic mean of the recall and precision and can be seen in equation 3.8. [45]

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.7}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.8}$$

Furthermore, the precision-recall curve can evaluate recall and precision for different thresholds, and the area under the curve (AUC PR) can be calculated to measure the model's general performance. Another common evaluation curve is the receiver operating characteristic (ROC), which plots recall against the false positive rate (FPR), seen in equation 3.9. Similarly, the area under the ROC curve (AUC ROC) can be used to measure the model's general performance. A model with an AUC ROC of 1 classifies all instances correctly, whereas a model with an AUC ROC of 0.5 classifies instances randomly. Since the axes are proportional to the class they represent, the ROC curve is insensitive to imbalanced datasets. However, the ROC curve can still give an overly optimistic view in imbalanced datasets since the true negatives would dominate the false positive rate. [26]

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{3.9}$$

3.3.2 Benchmark Algorithm

A simple benchmark algorithm was developed to quantify the performance of the trained models. This benchmark algorithm resembled a rule-based spoofing detection algorithm, allowing insights into whether machine learning methods are better at detecting spoofing than traditional rule-based methods. The benchmark algorithm was designed by setting a fixed interval for all 11 features. The intervals were chosen so that all instances of the *validation real* dataset would be detected as spoofing but so that the false positives were minimized.

The benchmark algorithm was evaluated using the *final real* dataset. For this evaluation, each upper threshold of the intervals was multiplied by $1 + c$, and each lower threshold of the intervals was multiplied by $1 - c$, where c is a constant between 0 and 1. This approach was used to determine if varying the value of c, meaning making the intervals larger, could enhance the performance of the benchmark algorithm. Additionally, this approach made

it possible to compare the results of the benchmark algorithm to the results of the machine learning models.

3.3.3 Stationarity of Data

Since normal trading behavior is constantly evolving and the definition of normal behavior now may not sufficiently represent future behaviors [26], the stationarity of the data was evaluated. This evaluation involved collecting limit order book data from the same month from 2020 to 2024, followed by performing the filtering and feature aggregation processes described earlier. Each year's dataset consisted of 500 data instances. The model demonstrating the highest performance was then used to classify these instances as anomalies or normal data. An anomaly score distribution diagram was visualized to compare the distribution across the datasets from different years. Additionally, the benchmark algorithm was used to compare these datasets. All datasets consisted only of normal data, indicating that this evaluation focused on how the false positive rate varied between the datasets from different years.

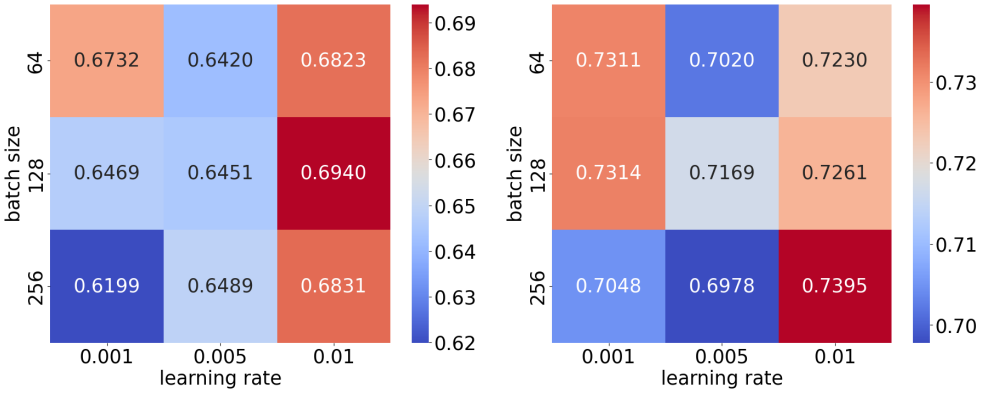
3.4 Parameter Tuning with 5-fold Cross Validation

To optimize the models, parameter tuning was performed using 5-fold cross validation. For the 5-fold cross validation, the complete dataset of 25,252 instances was divided into five folds of 5050 instances each. This resulted in a *validation synthetic* dataset for each fold consisting of 5050 normal data instances and 500 synthetic spoofing instances. The 500 synthetic spoofing instances did not change over the different folds.

For each of the five models, two parameters were chosen with three values that should be tested. This resulted in 9 different combinations for each model that should be tested for every fold. For each combination of parameters, the AUC ROC and the F1 score were measured. After performing all five folds, the average for each combination of parameters was calculated and visualized in a heatmap.

The two parameters tested for the autoencoder were the batch size and the learning rate. For the batch size, the three tested values were 64, 128, 256, and 0.001, 0.005, 0.01 for the learning rate. As can be seen in figure 3.6, a learning rate of 0.01 and a batch size of 256 gave the highest F1 score and the second highest AUC ROC, which was, therefore, the parameters chosen for the final

model.

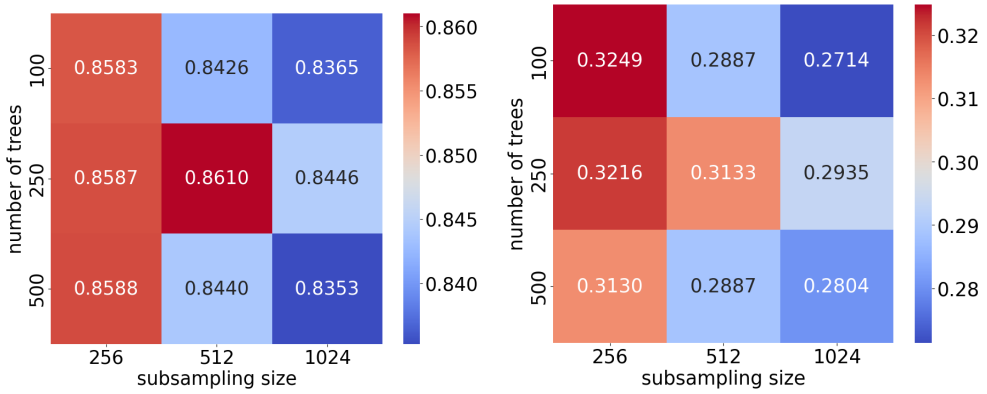


(a) AUC-ROC (b) F1 score

Figure 3.6: Autoencoder parameter tuning.

During the training, the IForest algorithm takes two input parameters: subsampling size ψ and number of trees t, which also were the ones tested during the cross-validation. Liu et al. [29] find that under the assumption that anomalies are few and different, a small subsampling size is enough for IForest to distinguish anomalies from normal points. Setting ψ to 256 was found to be enough to perform anomaly detection across a wide range of data. Regarding the second parameter t, Liu et al. find that path lengths converge well before $t = 100$.

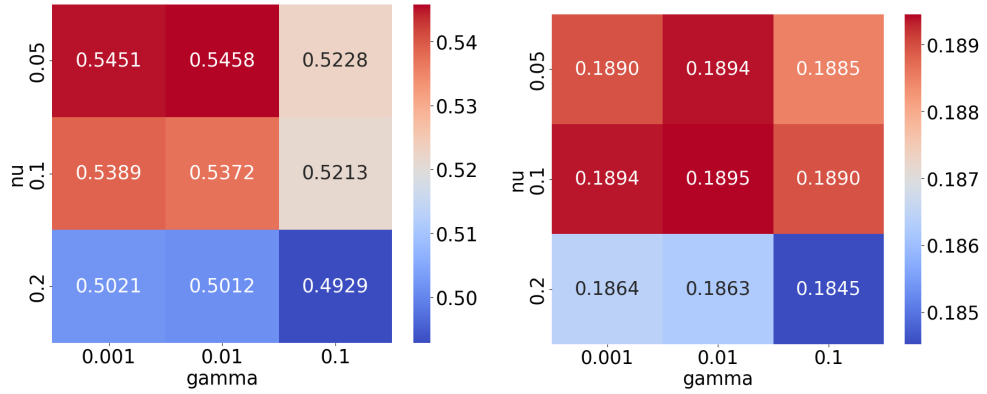
The three tested values for the t were 100, 250, 500, and 256, 512, 1024 for ψ. As can be seen in figure 3.7, setting ψ to 256 and t to 250 gave the second highest F1 score while being the top three value in the AUC ROC heatmap, which was therefore the chosen parameters for the final IForest model. In Sklearn's isolation forest implementation, more parameters can be set. All where left to default, except contamination which was set to 0.05.



(a) AUC-ROC (b) F1 score

Figure 3.7: IForest parameter tuning.

The two parameters tested for the OC-SVM were *nu* and *gamma*. The three tested values of *nu* were 0.05, 0.1, 0,2, and 0.001, 0.01, 0.1 for *gamma*. As can be seen in figure 3.8, a *gamma* value of 0.01 and *nu* set to 0.05 gave the second-highest F1 score and the highest AUC ROC score, which was, therefore, the chosen parameters for the final OC-SVM model. In Sklearn's OC-SVM implementation, more parameters can be set. All where left to default, except kernel which was set to *rbf*.



(a) AUC-ROC (b) F1 score

Figure 3.8: OC-SVM parameter tuning.

3.5 Feature Importance and Selection

Feature Importance and Selection was implemented using the Depth-based Isolation Forest Feature Importance (DIFFI) and autoencoder reconstruction

error method. Carletti et al. [34] had made their code publicly available, so DIFFI was implemented by cloning their Github repository[1]. The Isolation Forest models in the DIFFI were initialized with the same parameters as the IForest model used in the rest of the project. Apart from resulting in a feature importance ranking, the DIFFI also returns a feature selection graph, which shows the optimal number of features to select from the ranking to achieve the highest performance. Except for the five models previously mentioned, an IForest and an OC-SVM trained on the features selected from this method were also evaluated in the final evaluation.

The autoencoder reconstruction error method was implemented by following Gomes et al. [33] proposed method. First, the validation data was predicted using the trained autoencoder, resulting in reconstruction errors for the validation data. From here, the reconstruction error distribution was analyzed, and a threshold was chosen where the model seemed to distinguish the normal data from the spoofing data at best. For all the validation instances above this threshold, the average reconstruction error for each feature was calculated. The same was done for the normal validation instances below the chosen threshold. These results were then evaluated, where the features with high reconstruction error for the instances above the threshold but lower reconstruction error for the instances below the threshold were seen as the most important features.

[1] https://github.com/mattiacarletti/DIFFI

Chapter 4

Results and Analysis

4.1 Feature Correlation

A feature correlation analysis was performed on all features of the training dataset. The resulting feature correlation matrix can be seen in figure 4.1. All features can be said to have low correlation to each other, except *cancellation_rate* and *trades_on_genuine*. *cancellation_rate* is a value between 0 and 1, indicating the portion of the client's volume entered on the non-bona fide side that was canceled. *trades_on_genuine* is a value between 0 and 1, indicating the portion of the client's total traded volume during the sequence made on the genuine side. The feature correlation was decided to be low overall, so all features were kept.

4.2 Feature Importance and Selection

DIFFI and the autoencoder reconstruction error method were used to determine feature importance. The methods were applied to two datasets: the *validation real* dataset and the *validation synthetic* dataset.

4.2.1 DIFFI

The ranking of the most important features applied on the *validation real* dataset can be seen in table 4.1. The corresponding feature selection graph can be seen in figure 4.2. The feature selection graph shows the F1 score when training with a selected number of features: with the highest-ranking feature, the two highest-ranking features, the three highest-ranking features, and so on. As can be seen, the F1 score is low; a model with a score below 0.5

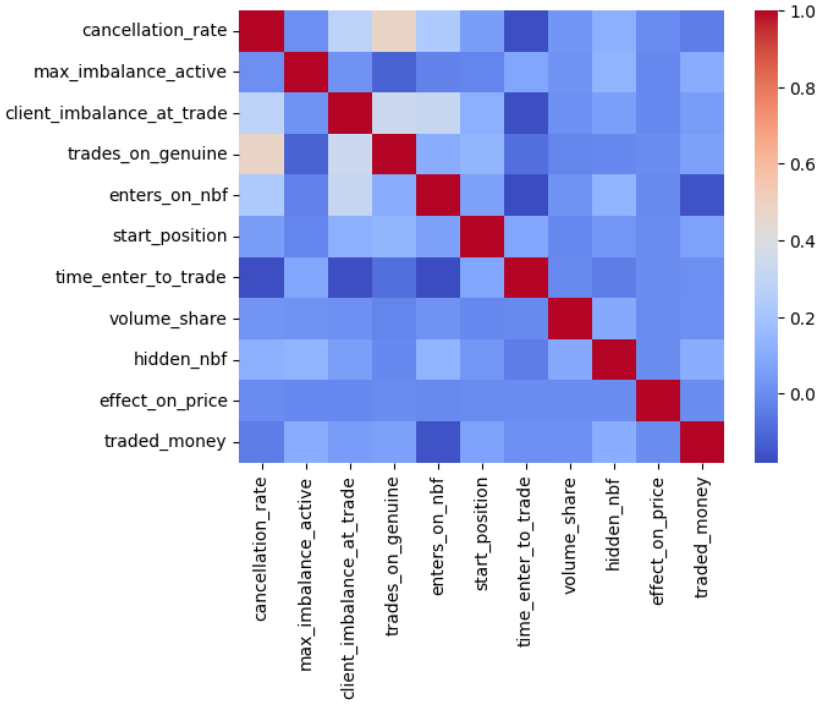


Figure 4.1: Feature correlation.

is considered poor performance. The main reason is that the Isolation Forests in the DIFFI method are evaluated using the full training dataset of 20201 data instances, labeled as negative instances, and the positive instances in the *validation real* dataset as positive instances. The negative instances largely outnumber the positive instances, resulting in an abnormally low F1 score.

Rank	Feature
1	hidden_nbf
2	volume_share
3	effect_on_price
4	traded_money
5	start_position
6	time_enter_to_trade
7	max_imbalance_active
8	enters_on_nbf
9	trades_on_genuine
10	cancellation_rate
11	client_imbalance_at_trade

Table 4.1: DIFFI - validation real dataset.

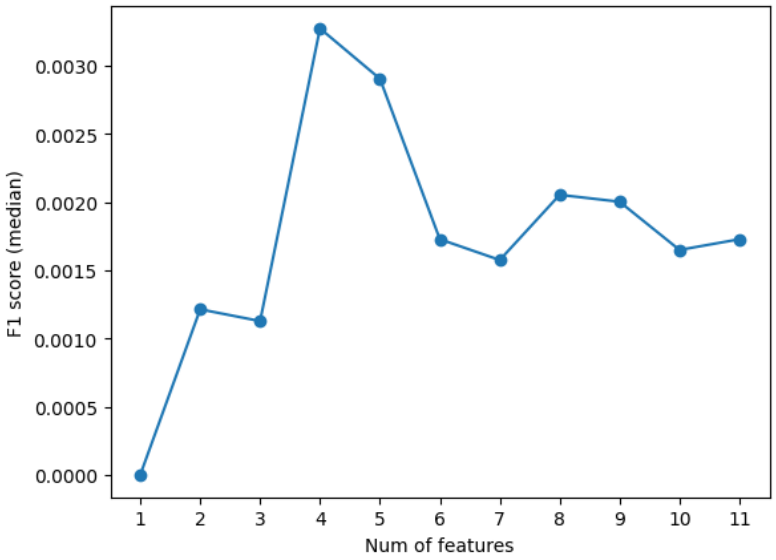


Figure 4.2: DIFFI - validation real dataset.

The ranking of the most important features applied on the *validation synthetic* dataset can be seen in table 4.2. The corresponding feature selection graph can be seen in figure 4.3. The ranking of the features is similar between the two test datasets; only *start_position* and *traded_money*, as well as *client_imbalance_at_trade* and *cancellation_rate* have switched spots.

Rank	Feature
1	hidden_nbf
2	volume_share
3	effect_on_price
4	start_position
5	traded_money
6	time_enter_to_trade
7	max_imbalance_active
8	enters_on_nbf
9	trades_on_genuine
10	client_imbalance_at_trade
11	cancellation_rate

Table 4.2: DIFFI - validation synthetic dataset.

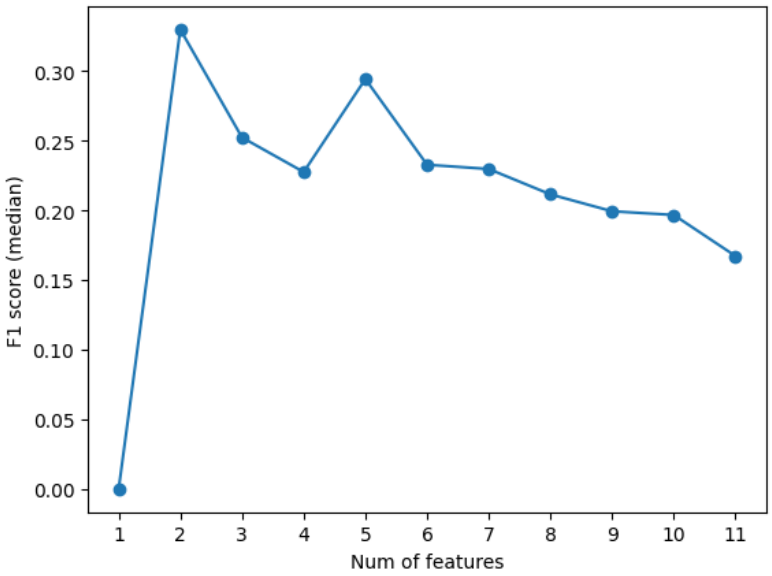


Figure 4.3: DIFFI - validation synthetic dataset.

4.2.2 Autoencoder Reconstruction Error

The second feature importance method used was the autoencoder reconstruction error method. The distribution of the reconstruction error for the validation datasets can be seen in figure 4.4. The threshold was set to 0.022 since, at that threshold, all *validation real* positive instances would have been classified as an anomaly.

The ranking of the most important features applied on the *validation real* dataset can be seen in table 4.3, and applied on the *validation synthetic* dataset above the threshold can be seen in table 4.4. The ranking of the most important features applied on the normal instances of the validation datasets below the threshold can be seen in table 4.5. The most important features when distinguishing spoofing from normal data according to the method are the ones that have a high error in tables 4.3 and 4.4 but a low error in table 4.5.

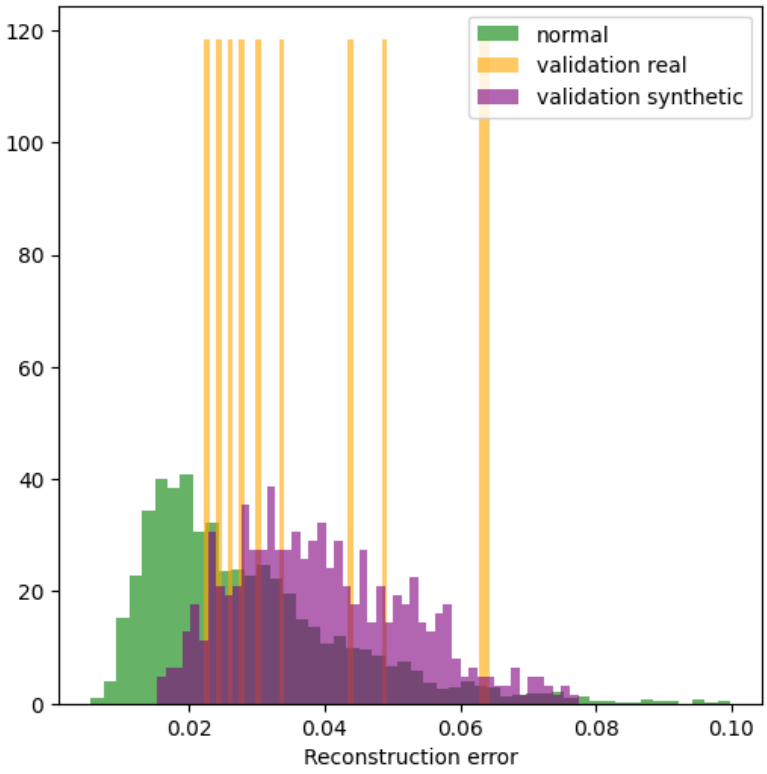


Figure 4.4: AE reconstruction error - distribution.

Rank	Feature	Average Error
1	effect_on_price	1.25e-01
2	volume_share	6.63e-02
3	hidden_nbf	6.14e-02
4	trades_on_genuine	4.57e-02
5	time_enter_to_trade	3.62e-02
6	client_imbalance_at_trade	3.42e-02
7	start_position	1.94e-02
8	enters_on_nbf	1.68e-02
9	cancellation_rate	1.10e-02
10	traded_money	6.88e-03
11	max_imbalance_active	1.02e-05

Table 4.3: Feature reconstruction error - validation real dataset above the threshold.

Rank	Feature	Average Error
1	effect_on_price	1.24e-01
2	hidden_nbf	6.88e-02
3	volume_share	6.11e-02
4	time_enter_to_trade	5.70e-02
5	client_imbalance_at_trade	5.30e-02
6	start_position	2.56e-02
7	cancellation_rate	2.38e-02
8	trades_on_genuine	1.81e-02
9	enters_on_nbf	1.03e-02
10	traded_money	9.53e-03
11	max_imbalance_active	1.19e-05

Table 4.4: Feature reconstruction error - validation synthetic dataset above the threshold.

Rank	Feature	Average Error
1	effect_on_price	3.19e-02
2	volume_share	3.15e-02
3	hidden_nbf	2.90e-02
4	traded_money	2.84e-02
5	trades_on_genuine	2.59e-02
6	cancellation_rate	1.39e-02
7	start_position	7.10e-03
8	max_imbalance_active	6.88e-03
9	time_enter_to_trade	3.92e-03
10	client_imbalance_at_trade	1.70e-04
11	enters_on_nbf	1.18e-05

Table 4.5: Feature reconstruction error - normal instances below the threshold.

4.2.3 Summary of Feauture Importance Findings

The DIFFI method showed that the best performance on the *validation real* dataset is achieved when selecting four features, specifically, *hidden_nbf*, *volume_share*, *effect_on_price*, *traded_money*. Therefore, an IForest and an OC-SVM, referred to as IForest-4 and OC-SVM-4, respectively, were trained using only these four features and compared with the other models. The two least important features were *cancellation_rate* and *client_imbalance_at_trade*. The ranking of the features was similar for the *validation real* and *validation synthetic* datasets.

The autoencoder reconstruction error method is more difficult to interpret since it does not directly show a ranking of the most important features. Instead, the most important features are the ones that have high average reconstruction error for the *validation real* positive instances and low average reconstruction error for the normal data instances. Most noticeable from the results is that the feature *effect_on_price* has about twice as big an average error as the second highest ranked feature. The other features that had high errors for the *validation real* positive instances but significantly lower for the normal instances were *volume_share* and *hidden_nbf*. All features have an overall low average error, and the ranking of the features is similar between the *validation real* and *validation synthetic* datasets.

4.3 Model Evaluation

The trained models were evaluated in four ways: by analyzing the anomaly score distribution, ROC curve, precision-recall curve, and different metrics at chosen anomaly score thresholds. Two main test datasets were used during the model evaluation: the *final real* dataset and the *final synthetic* dataset, which is based on the *final real* dataset. The *validation real* dataset is added to some diagrams.

4.3.1 Anomaly Score Distribution

The autoencoder's anomaly score distribution can be seen in figure 4.5. When using an autoencoder as an anomaly detector, a larger reconstruction error means a larger chance for an anomaly. As shown in the figure, both the *final synthetic* dataset and the real instances of spoofing are more inclined to have a higher reconstruction error than the training and the normal data. Furthermore, it can be noticed that the normal data, which was not used during training or as validation, follows the training data reconstruction error distribution almost perfectly.

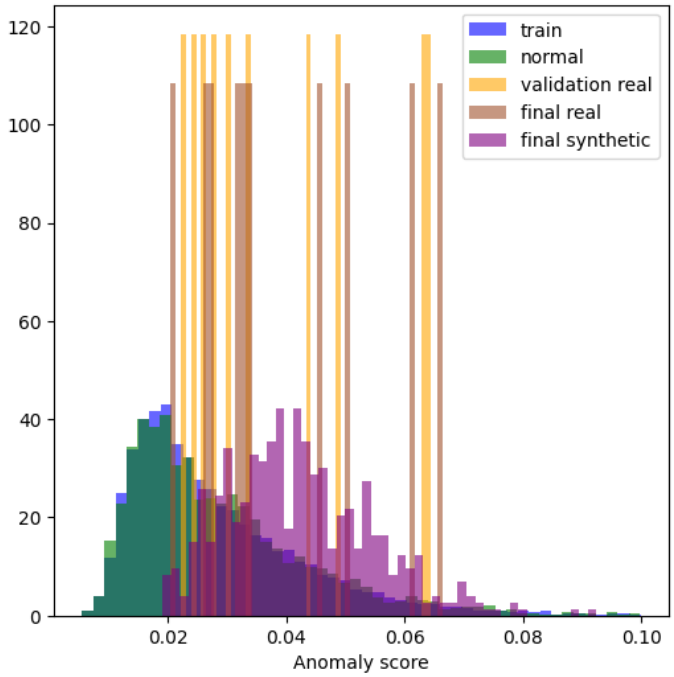
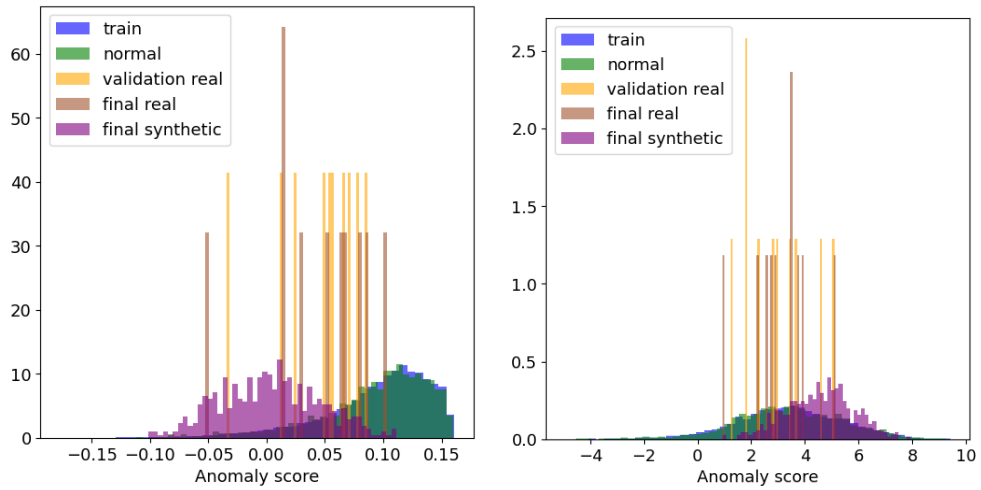


Figure 4.5: Final evaluation - anomaly score distribution AE.

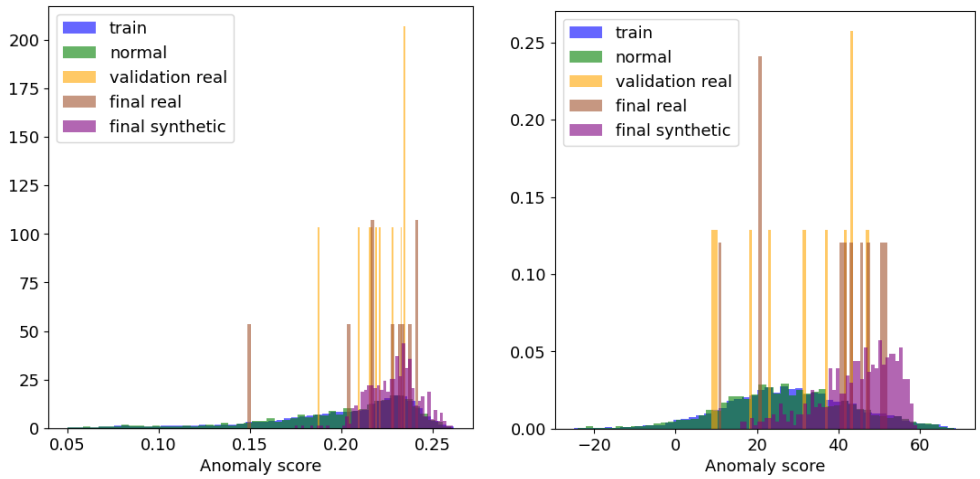
The anomaly score distribution for the IForest and the OC-SVM can be seen in figure 4.6. When implementing IForest or an OC-SVM using the Sklearn library, a lower anomaly score indicates a bigger probability of being an anomaly. Regarding the IForest, it can be seen that both the *final validation* dataset and the real spoofing instances are more inclined to have a lower anomaly score than the training and normal data. However, such a pattern cannot be seen regarding the OC-SVM. Instead, the *final validation* dataset is even classified as more normal than the training and normal datasets.



(a) IForest (b) OC-SVM

Figure 4.6: Final evaluation - anomaly score distribution IForest and OC-SVM.

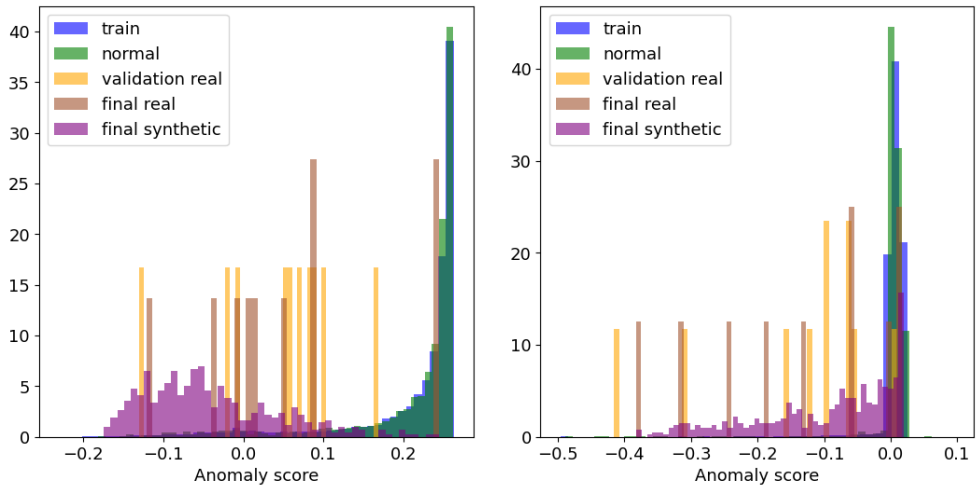
The anomaly score distribution for the hybrid models, the AE + IForest and the AE + OC-SVM, can be seen in figure 4.7. In both diagrams, the *final synthetic* data and the real spoofing instances have a higher anomaly score than the normal data, indicating that the models have not learned how to distinguish the spoofing instances from the normal data.



(a) AE + IForest (b) AE + OC-SVM

Figure 4.7: Final evaluation - anomaly score distribution hybrid models.

The anomaly score distribution for the feature selection models, IForest-4 and OC-SVM-4, can be seen in figure 4.8. In both diagrams, it can be seen that the models have been able to distinguish the *final synthetic* dataset and the spoofing instances from the training and normal datasets.



(a) IForest-4 (b) OC-SVM-4

Figure 4.8: Final evaluation - anomaly score distribution IForest-4 and OC-SVM-4.

4.3.2 ROC-curve and Precision-Recall-curve

The ROC curve for the *final real* dataset can be found in figure 4.9, where it can be seen that IForest-4 has the highest performance and the hybrid models the lowest. The dotted line represents a model with random guessing, serving as a baseline for model performance. However, both the hybrid models and OC-SVM perform no better than this random model. OC-SVM-4 has a slightly lower false positive rate than IForest-4 before achieving 0.7 in true positive rate. The IForest-4 has classified all spoofing instances as an anomaly when the false positive rate is around 0.45.

The ROC curves for the *validation real* and *final synthetic* datasets can be found in figure 4.10. Most models perform higher on the *validation real* than on the *final real* dataset, where IForest-4 achieves 1.0 in true positive rate at only 0.2 in false positive rate. The *final synthetic* dataset performs similarly to the *final real* dataset in figure 4.9 but has a smoother curve since it consists of more data instances.

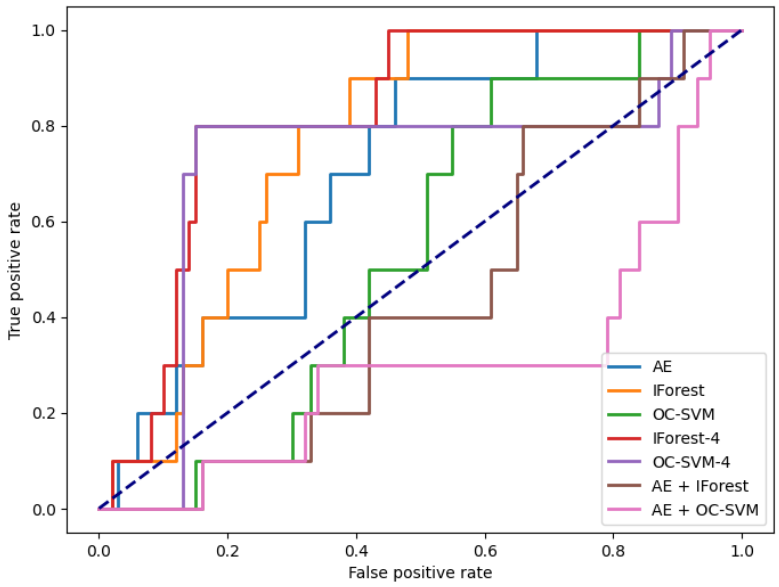
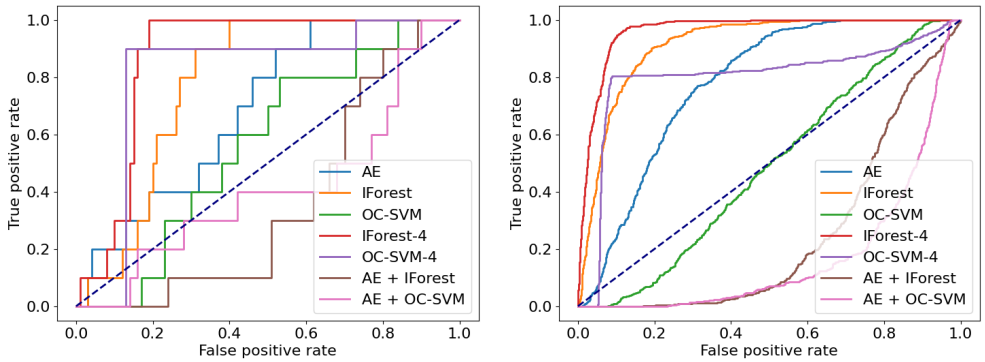


Figure 4.9: Final evaluation - ROC-curve on final real test set.



(a) Validation real dataset.

(b) Final synthetic dataset.

Figure 4.10: Final evaluation - ROC-curve on validation real and final synthetic test set.

The precision-recall curve for the *final real* dataset can be found in figure 4.11. Most noticeable is the precision increases when the recall increases, instead of the precision starting at a high number and then decreasing. This is because the data instances with the highest anomaly scores belong to the normal dataset. At the lowest thresholds, more normal data instances will be classified as anomalies than spoofing instances, resulting in low precision when the recall is low. Additionally, IForest-4 had the highest performance, but OC-SVM-4 had a higher performance when the recall was between 0.5 and 0.7. The precision is quite low overall, mainly due to the few instances in the *final real* dataset. Therefore, a clearer picture of the model's precision-recall trade-off can be seen in figure 4.12b, where the *final synthetic* dataset is used instead. Both IForest models and the OC-SVM-4 exhibit high performance; however, the OC-SVM-4 maintains this only when its recall rate is 0.8 or lower. In figure 4.12a, the precision-recall curve for the *validation real* dataset can be seen, which shows similar performance for the models as when using the *final real* dataset.

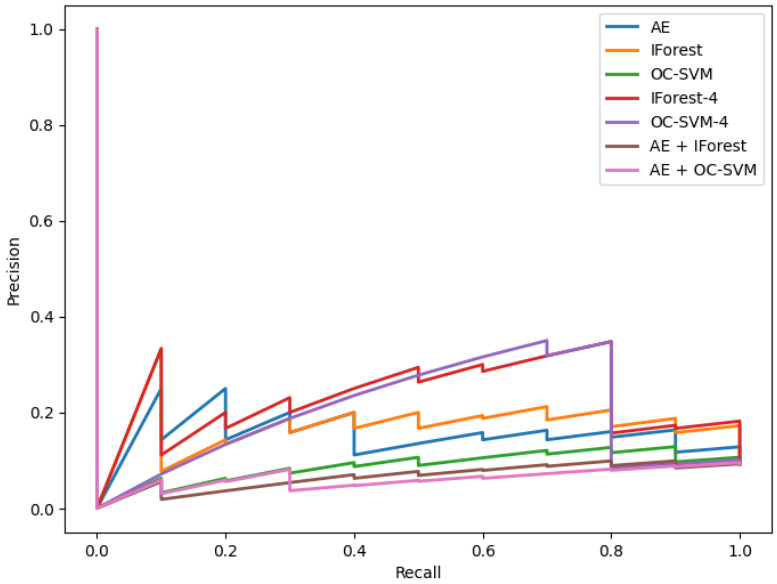
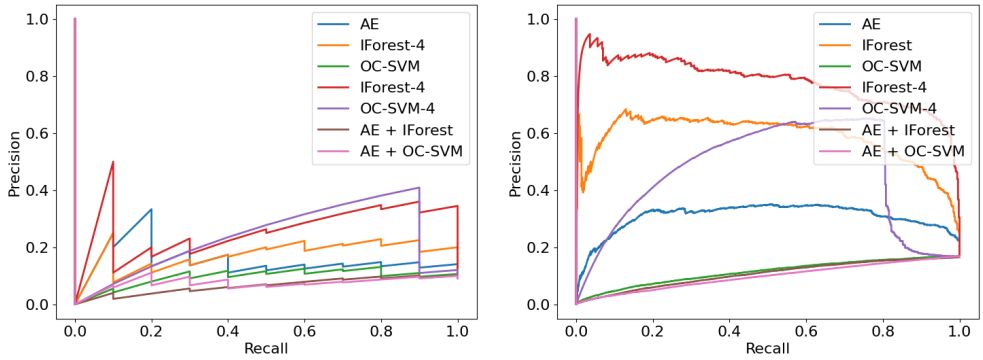


Figure 4.11: Final evaluation - PR-curve on final real test set.



(a) Validation real dataset.

(b) Final synthetic dataset.

Figure 4.12: Final evaluation - PR-curve on validation real and final synthetic.

The AUC ROC and the AUC PR are calculated for each model. For the *final real* dataset, the values can be found in table 4.6; for the *final synthetic* dataset, the values can be found in table 4.7.

Model	AUC ROC	AUC PR
AE	0.71	0.22
IForest	0.77	0.24
OC-SVM	0.54	0.14
AE + IForest	0.37	0.12
AE + OC-SVM	0.31	0.12
IForest-4	**0.82**	**0.30**
OC-SVM-4	0.72	0.26

Table 4.6: Final evaluation - area under curve on final real dataset.

Model	AUC ROC	AUC PR
AE	0.78	0.31
IForest	0.91	0.59
OC-SVM	0.49	0.11
AE + IForest	0.30	0.11
AE + OC-SVM	0.17	0.098
IForest-4	**0.96**	**0.78**
OC-SVM-4	0.80	0.45

Table 4.7: Final evaluation - area under curve on final synthetic dataset.

4.3.3 Benchmark Model

The benchmark model, with the constant c set to 0, correctly identified all instances from the *validation real* dataset. It also maintained a very low false positive rate of 0.039% for the normal instances. However, it only classified 40% correctly as anomalies from the *final real* dataset, resulting in a recall of 0.4 and precision of 0.80. Figure 4.13 shows the classified spoofing ratio for different constant values. Based on these results, the best value of the constant would be 0.6, where the benchmark algorithm would classify 80% instead of 40% of the *final real* dataset correctly. Additionally, the false positive rate remains relatively low at 3.5%.

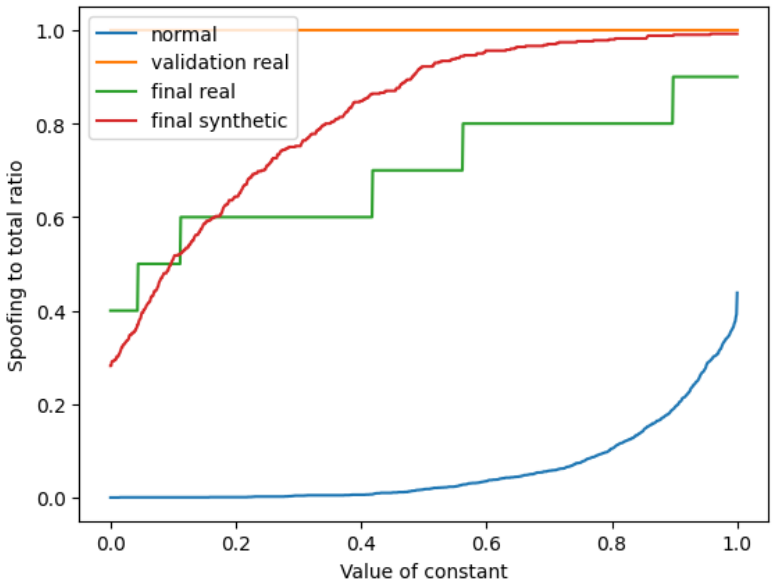


Figure 4.13: Spoofing to total ratio by benchmark algorithm.

4.3.4 Evaluation Metrics at Different Anomaly Score Thresholds

To compare the models, the F1 score, precision, and recall were calculated at different thresholds for each model. The results for the *final real* dataset can be found in table 4.8, and for the *final synthetic* dataset in table 4.9. Several models achieve the same performance at some thresholds for the *final real* data. The reason for this is the low amount of positive instances, making it more likely that models achieve the same true positive rate at the same threshold. Of the machine learning models, IForest-4 achieved the best performance at all thresholds for both the *final real* and *final synthetic* dataset. However, the benchmark (BM) algorithm clearly achieved higher performance than all machine learning models, mainly due to a clearly lower false positive rate at the different thresholds. The tables are missing data for the benchmark algorithm in the recall = 1 row because the benchmark algorithm never reaches a recall of 1 when varying the constant c between 0 and 1.

Threshold percentile	Metric	AE	IF	OC-SVM	AE + IF	AE + OC-SVM	IF-4	OC-SVM-4	BM
recall=1	F1	0.23	0.29	0.19	0.17	0.17	**0.31**	0.18	
	Precision	0.13	0.17	0.11	0.093	0.10	**0.18**	0.10	
	Recall	1.0	1.0	1.0	1.0	1.0	1.0	1.0	
70th	F1	0.19	0.32	0.093	0.047	0.047	**0.37**	**0.37**	**0.42**
	Precision	0.12	0.21	0.061	0.030	0.030	**0.24**	**0.24**	**0.27**
	Recall	0.40	0.70	0.20	0.10	0.10	**0.80**	**0.80**	**0.90**
80th	F1	0.25	0.25	0.063	0.063	0.063	**0.44**	**0.44**	**0.5**
	Precision	0.18	0.18	0.045	0.045	0.045	**0.32**	**0.32**	**0.36**
	Recall	0.40	0.40	0.10	0.10	0.10	**0.70**	**0.70**	**0.80**
90th	F1	**0.19**	0.095	0.00	0.00	0.00	**0.19**	0.00	**0.76**
	Precision	**0.18**	0.091	0.00	0.00	0.00	**0.18**	0.00	**0.73**
	Recall	**0.20**	0.10	0.00	0.00	0.00	**0.20**	0.00	**0.80**

Table 4.8: Metrics at different thresholds for the final real dataset.

Threshold percentile	Metric	AE	IF	OC-SVM	AE + IF	AE + OC-SVM	IF-4	OC-SVM-4	BM
recall=1	F1	0.38	0.44	0.24	0.27	0.23	**0.46**	0.29	
	Precision	0.23	0.29	0.14	0.16	0.13	**0.30**	0.17	
	Recall	0.99	0.99	0.67	0.91	0.73	**1.00**	0.92	
70th	F1	0.45	0.63	0.047	0.011	0.017	**0.70**	0.57	**0.70**
	Precision	0.35	0.49	0.036	0.0088	0.013	**0.54**	0.44	**0.54**
	Recall	0.63	0.89	0.066	0.016	0.024	**0.98**	0.81	**0.99**
80th	F1	0.37	0.64	0.020	0.0036	0.0018	**0.77**	0.71	**0.87**
	Precision	0.34	0.58	0.018	0.0033	0.0017	**0.70**	0.65	**0.80**
	Recall	0.41	0.71	0.022	0.0040	0.0020	**0.85**	0.78	**0.97**
90th	F1	0.25	0.48	0.0050	0.00	0.00	**0.60**	0.39	**0.75**
	Precision	0.34	0.64	0.0066	0.00	0.00	**0.80**	0.52	**0.99**
	Recall	0.41	0.39	0.0040	0.00	0.00	**0.48**	0.31	**0.60**

Table 4.9: Metrics at different thresholds for the final synthetic dataset.

4.3.5 Stationarity of Data

Finally, the stationarity of the data was evaluated. Since IForest-4 performed the highest of the machine learning models, the anomaly score distribution for the data from 2020 to 2024 using IForest-4 can be seen in figure 4.14. The anomaly score distribution is similar for all year's datasets, indicating that

normal trading characteristics have not significantly changed between 2020 and 2024. Additionally, the stationarity of the data was evaluated using the benchmark algorithm, and the results can be seen in figure 4.15. The false positive rates are similar across the datasets; however, it is noticeable that the rate for the 2020 dataset is slightly higher than for the other years, while both 2023 and 2024 have among the lowest. In other words, the results from the benchmark algorithm suggest that the characteristics of normal trading have slightly changed over time.

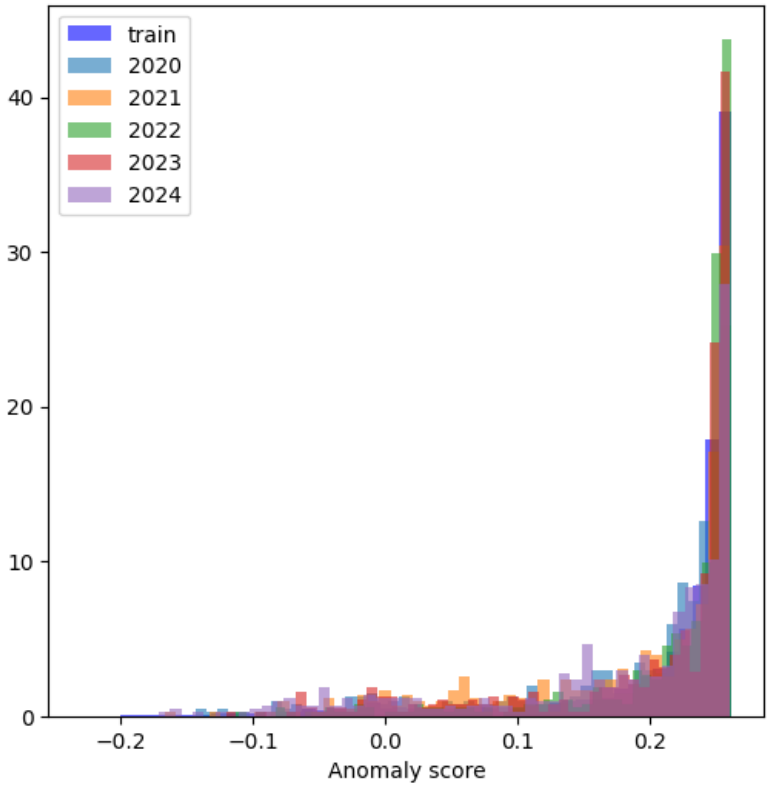


Figure 4.14: Iforest-4 - stationarity of data.

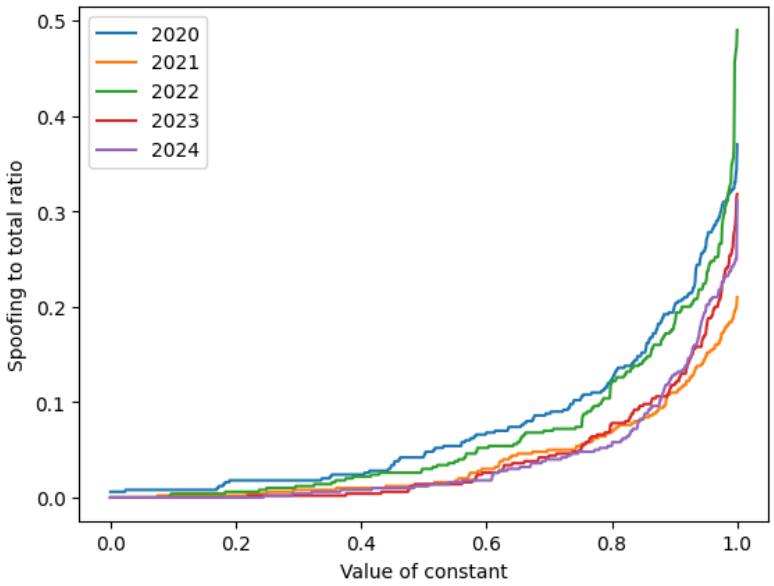


Figure 4.15: Benchmark algorithm - stationarity of data.

4.3.6 Summary of Model Evaluation Findings

The anomaly score distributions, ROC curves, and precision-recall curves all show that IForest-4, OC-SVM-4, IForest, and the Autoencoder are the best models at detecting spoofing, while the hybrid models and OC-SVM have not learned how to distinguish spoofing instances from normal data. The IForest-4, which is the best model, has a relatively high false positive rate; it is around 0.45 when it has classified all spoofing instances of the *final real* dataset as an anomaly and slightly lower than 0.2 when it has classified 80% of the spoofing instances correctly.

The ROC curves show similar performance for the *final real* and *final synthetic* datasets. The PR curves are different for the *final real* and *final synthetic* datasets, mainly due to the low number of positive instances in the *final real* dataset, resulting in a lower precision.

The benchmark algorithm performs poorly on the *final real* dataset when the intervals for the features are set as tight as possible; that is, when the constant c is set to 0. The recall is only 0.40, but the false positive rate is low, giving a precision of 0.80. However, when the constant is increased, thereby enlarging the size of the intervals, recall increases while only slightly raising the false positive rate. As a result, the benchmark algorithm achieves higher results than all other models at all chosen thresholds.

Using the benchmark algorithm to evaluate the stationarity, the results

show that the false positive rate is slightly higher for the 2020 data than for the 2023 and 2024 data, indicating that normal trading characteristics have changed over time. However, this is not shown in the anomaly score distribution of the IForest-4.

Chapter 5

Discussion

5.1 Model Performance

The models with the highest performance were the IForest-4 and OC-SVM-4, which were trained on the four most important features identified by the Depth-based Isolation Forest Feature Importance (DIFFI) method. This high performance can be explained by anomalies often displaying prominent abnormal features or characteristics in low-dimensional spaces [26]. However, they easily become hidden and indiscernible in high-dimensional spaces, a phenomenon known as the curse of dimensionality [26].

Another solution to feature selection to handle the curse of dimensionality is to reduce the dimensionality of the data to a lower, but still informative, space. This was the aim of the hybrid models, where the autoencoder would be used as a dimensionality reducer, followed by applying a one-class classifier on the informative latent space. Surprisingly, contradicting the hypothesis, the hybrid models performed the lowest, even having an AUC ROC of lower than 0.5, which is the AUC ROC of a model that classifies randomly. This indicates that the autoencoder did not learn to transform the data into a more informative latent space. There may be many reasons for the low performance, but one main cause is probably the heterogeneous nature of anomalies. Anomalies are irregular, with different classes of anomalies demonstrating distinct characteristics of abnormality [26]. This variability makes it challenging for the autoencoder to capture and preserve the complex interactions of features in high-dimensional data within the newly reduced latent space [26]. This challenge was better handled by the IForest model, achieving an AUC ROC of 0.77, confirming that isolation forests effectively can handle high-dimensional data [26].

Finally, using only the autoencoder also resulted in higher performance than the hybrid models. Especially in the anomaly score distribution, seen in figure 4.5, it can be seen that all spoofing cases had an anomaly score of at least 0.02, while a large part of the normal instances had a lower score than 0.02. However, if looking at the autoencoder feature importance in table 4.3, we see that the feature *effect_on_price* had a twice as large average reconstruction error as the second highest ranked feature. This indicates that the autoencoder's performance was largely affected by only one feature. Additionally, many features with the highest average error were also the highest ranked for the normal data instances. This suggests that the autoencoder could generalize over anomalies quite well, confirming the findings of related work [46].

The benchmark algorithm, when the constant c was set to 0, achieved a recall of 0.4 and a precision of 0.8. This reflects low performance in terms of recall but high performance in terms of precision. The primary reason for this result is that the fixed ranges set are relatively tight, perfectly adapted to the *validation real* data instances, resulting in a low false positive rate but also a correspondingly low true positive rate. This situation exemplifies the challenges associated with typical rule-based algorithms, especially the difficulty in fully formalizing abnormal trading behavior through predefined patterns and thresholds [5, 6]. While the benchmark algorithm achieved a recall of 1.0 for known spoofing cases, it only managed a recall of 0.4 for unknown cases. This difference highlights the potential advantages of unsupervised anomaly detection in this field, as such methods can identify previously unknown anomalies. However, a clear benefit of rule-based algorithms is their low false positive rate. This is why the benchmark algorithm excelled at all thresholds compared to the machine learning models, as seen in tables 4.8 and 4.9. The benchmark algorithm maintained a false positive rate of just 0.00039, which would enable human analysts in surveillance teams to effectively review all true and false positives.

5.2 Impact of False Positives

An ideal market manipulation detection model should completely eliminate false negatives while a low number of false positives could be tolerated. However, a false positive rate that is too high can render the model impractical and costly for use in trading surveillance contexts. The best model, IForest-4, had a false positive rate of 0.45 when all instances of the *final real* dataset were detected and a false positive rate lower than 0.2 when 80% of

all *final real* instances were found. A false positive rate of 0.2 could be seen as tolerable, but if the total dataset each week consisted of millions of instances, a 0.2 false positive rate would require extensive analyst intervention, increasing operational costs significantly. Therefore, a model useful in a trading surveillance context should probably have a false positive rate lower than 0.1 when the recall is close to 1.0. In any case, choosing the anomaly score threshold value will always be a trade-off between recall and false positive rate.

A potential reason for the high false positive rates observed in all models might be that the training dataset contained more anomalies than initially assumed. When using unsupervised methods, the assumption is that anomalous events are far less frequent than normal events [26]. If not, these methods will have higher false positive rates than expected [26]. A possible solution could, therefore, have been to analyze the false positives with the highest anomaly scores, and if analysts believe they should be seen as positive instances, label them as positive instances instead.

5.3 Feature Importance

The four most important features according to the DIFFI method are *hidden_nbf*, *volume_share*, *effect_on_price*, and *traded_money*. The two least important features were *cancellation_rate* and *client_imbalance _at_trade*. That both *hidden_nbf* and *volume_share* are important features makes sense since a spoofing instance requires that a sufficiently large order is placed to create a false impression of trading interest. Additionally, this large order needs to be non-hidden since otherwise, the large order would not be able to influence the market. As a result, all instances of the spoofing instances in the *validation real* and *final real* datasets have a low *hidden_nbf* value and a significantly high *volume_share* value. That *effect_on_price* and *traded_money* also are important can be explained by the fact that in spoofing instances, the aim is often to buy or sell at prices more favorable than would be possible without the non-bona fide orders, which means that in spoofing instances there has been an effect on the price in the desired direction, and the money traded on the genuine side is of significant size.

The cancellation rate was initially seen as one of the most important characteristics of spoofing, since a spoofing instance requires that a majority of the non-bona fide orders are canceled. The fact that the feature *cancellation_rate* is ranked second lowest could be because the normal instances also have a high cancellation rate. In the filtering process, only the sequences with large orders are selected, and it may be

that the majority of large orders placed in an order book are canceled. Therefore, a high *cancellation_rate* may indicate large orders but does not distinguish spoofing from other large orders placed. Additionally, *client_imbalance_at_trade* was ranked the lowest. In a typical spoofing instance, the *client_imbalance_at_trade* should be high since the client should have a substantially larger volume on the non-bona fide side than on the genuine side when the genuine trade occurs. However, in some spoofing cases, the non-bona fide orders are canceled before the genuine trade happens, resulting in a low *client_imbalance_at_trade* for some spoofing cases. This feature being ranked the lowest indicates that it is irrelevant when describing spoofing.

The ranking of the features using the *validation synthetic* dataset is similar to that of the *validation real* dataset. This indicates that the synthetic dataset successfully represented the real spoofing instances.

Feature importance was also performed using the autoencoder reconstruction error. The features that had the highest reconstruction error in the *validation real* dataset, while significantly lower for the normal dataset below the chosen threshold, were *effect_on_price*, *volume_share* and *hidden_nbf*. All of them were also in the top four important features of the DIFFI method. The features where the normal dataset even had higher average scores were *cancellation_rate*, *traded_money*, and *max_imbalance_active*, indicating that these features are not important when distinguishing spoofing instances from normal data. However, since all average errors were quite low, it is difficult to interpret these results and derive more insights. This is further made challenging by deep learning models, including autoencoders, often operating in a black-box fashion [26].

5.4 Stationarity of Data

The stationarity of the data was evaluated by using the IForest-4 model and the benchmark algorithm. From the anomaly score distribution of the IForest-4 model, no significant difference could be found between the different years. However, the benchmark algorithm achieved an overall lower false positive rate in 2023 and 2024 and an overall higher false positive rate in 2020. This indicates that normal trading behavior is continuously evolving, confirming the hypothesis. The reason why the IForest-4 anomaly score distribution does not show any difference could be that it only uses 4 of the features, while the benchmark algorithm uses all 11 features. Given the continuous evolution of normal trading behavior, it is important to determine the optimal time frame

for data collection for the training dataset. In this thesis, the training dataset was selected from six consecutive months. This time frame was chosen to collect enough data for the training dataset. However, choosing such a long time frame could have made it more difficult for the models to learn what normal trading is, suggesting a shorter time frame could have improved the performance. Therefore, choosing the time frame for the data collection is probably a trade-off between performance and having enough data for the training dataset.

5.5 Discussion of Method

5.5.1 Data Transformation

In this thesis, it was chosen to transform raw order book data into aggregated feature vectors. The main reasons for this were simplifying the detection task and enabling a wider selection of machine learning methodologies. However, a consequence is that important information needed to identify spoofing may be eliminated, such as information about the client's individual order book events.

The order book data was filtered to reduce the amount of data and make it possible for the machine learning models to uncover patterns in the data that are not discernible to humans. A potential issue with this approach is that some spoofing cases may be filtered out, e.g., if the volume of the non-bona fide order is lower than the chosen fixed threshold or if the non-bona fide order is placed at position 0 in the limit order book. Preferably, no fixed thresholds are wanted to be set to ensure that all spoofing instances, including the unknown, can be detected.

Another challenge with the chosen data transformation method is which features should be chosen. There is no right or wrong answer to how to aggregate features and how to choose which ones to include in the final feature set. In this thesis, the features selected were, therefore, a result of iterative training, evaluation, and discussion with domain experts. If a method was chosen where raw data was used instead of aggregated features, the results would probably be easier to compare to related work and would not be as dependent on subjective choices.

5.5.2 Synthetic Dataset

The synthetic dataset, generated by the distribution method, was created to handle the limitation of only having a small number of spoofing instances available. The dataset was mainly used in the validation process but was also evaluated and compared to the real spoofing cases in the final evaluation.

In the validation process, the *validation synthetic* dataset was used to perform the parameter tuning, mainly using 5-fold cross-validation but also by evaluating the anomaly score distribution, ROC curves, and precision-recall curves. One obvious risk with using synthetic data in the validation process is that it is unknown exactly how well the synthetic dataset represents real spoofing strategies, resulting in tuning the models perfectly after the synthetic data but maybe not for spoofing detection. However, already in the validation process, the results from the *validation synthetic* dataset were compared with those from the *validation real* dataset and ensured that it was similar. For example, the anomaly score distributions were compared for both datasets, such as in figure 3.3. Since the synthetic data seemed to represent the real spoofing instances well, it was seen as the best available option to use synthetic data in the validation process.

The final evaluation also showed that the synthetic dataset represented the real instances well. This can be seen in the anomaly score distribution diagrams, such as in figure 4.5, where the *final synthetic* dataset follows the *final real* dataset instances quite well. Additionally, the ROC and precision-recall curves of the *final synthetic* and *final real* datasets have similar performance, with the difference that the *final synthetic* curves are smoother since more instances are included in the dataset. For example, the curves for the OC-SVM-4 model are almost perfectly following each other in figures 4.9 and 4.10b. However, there are also differences, such as the curves for IForest-4 and IForest, which perform better for the *final synthetic* dataset than for the *final real* dataset. The differences are seen even more clearly in the precision-recall curves seen in figures 4.11 and 4.12b, where the precision is clearly higher for the *final synthetic* dataset than the *final real* dataset. The main reason for this is that there is a lower rate of positive instances in the *final real* dataset, making it a more imbalanced dataset and, therefore, making the false positive rate affect the precision to a higher extent. Despite these differences, many similarities between the curves are still apparent, such as the hybrid models and OC-SVM having low performance and the OC-SVM-4 having a growing precision until reaching a recall of 0.8. Nonetheless, both the IForest-4 and IForest models seem to perform better on the *final synthetic* dataset than on

the *final real* dataset.

So overall, the synthetic dataset based on the distribution method represents typical spoofing behavior quite well. However, knowing how well a synthetic dataset represents real spoofing is still difficult since there are few real spoofing instances with which to evaluate the result. In this thesis, the synthetic dataset was chosen based on the final aggregated feature distribution, but another possibility would be to synthesize parts of raw limit order book data with spoofing instances and then add the filtering and feature aggregation to this data. However, that would be a much more time-consuming and complex task since, for every spoofing instance, it would be required to synthesize a complete limit order book for a certain time frame.

5.5.3 Parameter Tuning

Parameter tuning was performed in this thesis by using 5-fold cross-validation. The complete dataset of 25,252 normal instances was used for this parameter tuning and divided into five folds. This approach meant the same normal data instances were used in the 5-fold cross-validation as in the training, validation, and final data sets. This is not optimal, as it implies that the parameter-tuned models may already be optimized for the testing data, achieving higher performance than they would on unknown data. In other words, the generalizability may be lower than the results indicate. The reason why all available normal data was used for the cross-validation was to ensure that the combined instances of four folds were enough to train the models. Additionally, it was not seen as a significant issue since this issue only regarded the normal data instances. The only spoofing data instances used during the parameter tuning were the ones belonging to the validation data sets, meaning that the spoofing data instances belonging to the final data sets were never used during validation or parameter tuning. However, the impact of this issue could be reduced by at least not using the normal instances belonging to the final data sets during the parameter tuning.

5.6 Limitations

There are two main limitations to this thesis. Firstly, spoofing, as a form of market manipulation, can vary over time and appear differently depending on stock and market conditions. Additionally, as indicated by the evaluation of data stationarity, normal trading behavior is continuously evolving. This phenomenon, known as concept drift [26], must be accounted

for when developing spoofing detection models to ensure their usefulness. Consequently, even if a model performs well on the stock data selected for this thesis, it may not perform as well on datasets from different periods or stocks.

Secondly, the scarcity of real spoofing instances reduces the reliability of the results. To address this, a synthetic dataset was created that achieved a similar performance as the dataset with real spoofing cases; however, since it is derived from the limited real spoofing data, it may not fully represent the range of spoofing activities.

5.7 Comparison to Related Work

Poutré et al. [6] introduced a hybrid model consisting of a transformer autoencoder combined with an OC-SVM to detect trade-based market manipulation, including spoofing/layering. This inspired the hybrid models in this paper. However, a basic autoencoder was chosen instead of the transformer autoencoder since it was chosen to transform the raw order book data into aggregated features, enabling the use of simpler models. Poutré et al. achieved an AUC ROC of 0.900 and 0.847, a higher performance than the resulting models in this thesis. However, they do not specifically document their results on layering; instead, they combine them for all trade-based market manipulation. Therefore, it is difficult to compare the results.

Zhai et al. [9] proposed a k-nearest neighbor and one-class support vector machine, where the OC-SVM achieved impressive results with a precision of 1.0 and recall of 0.9933. However, their test set with real spoofing cases consisted only of high-frequency trading, suggesting that it is easier to detect high-frequency spoofing than non-high-frequency spoofing.

Unlike the other related work, Do et al. [8] used supervised learning to detect spoofing. This was enabled by them having access to a global sample of prosecuted spoofing cases. They developed a random forest and a boosted tree classification model and achieved AUC ROC scores of 0.96 and 0.97, respectively, which are higher results than the models in this thesis achieved. While supervised methods typically yield better results, they also have the disadvantage of being unable to detect unknown spoofing cases.

To summarize, the found related work has achieved higher performance in detecting spoofing. However, all methods have limitations, resulting in that detecting trade-based market manipulation, such as spoofing, still remains an open problem [26].

5.8 Ethics and Sustainability

This project does not explicitly focus on sustainability. However, as it involves machine learning, there are indirect implications for environmental sustainability due to the energy consumed during the training and usage of the models. In this thesis, the absence of large-scale machine learning systems means minimal energy costs associated with training and usage. Should future iterations of this project use more complex models and larger datasets, and if these models are deployed extensively, the energy consumption could increase but still be low since no further external resources would be needed. The impact on sustainability would, therefore, still be close to irrelevant. However, it is always important with machine learning to be aware of energy consumption and use models that are as efficient as possible.

Furthermore, this project addresses ethics. The insights gained from this project can increase the chances of detecting trade-based market manipulation on the stock exchanges. This would result in a more fair playing field on the stock exchange and increase the chance of traders performing illegal trades being accounted for. Additionally, using effective trading surveillance methods is essential for maintaining market integrity and protecting trader interests.

Chapter 6

Conclusions and Future work

6.1 Conclusions

The objective of this master's thesis was to evaluate the performance of machine learning techniques in detecting spoofing activities within limit order book data. A hybrid model was proposed and evaluated alongside common unsupervised anomaly detection methods. Although the hybrid model did not perform effectively in detecting spoofing, the Isolation Forest and the One-Class Support Vector Machine when trained on the four most significant features, yielded more promising results. Specifically, they achieved AUC ROC scores of 0.82 and 0.72 and AUC PR scores of 0.30 and 0.26, respectively, on the final real dataset. However, the high false positive rate, particularly at thresholds where recall is maximized, indicates that further improvements are necessary.

Additionally, when these models were benchmarked against a rule-based algorithm, the latter achieved a recall of 0.4 and a precision of 0.8, confirming the challenge of rule-based algorithms to find unknown spoofing instances. However, the false positive rate was low, only 0.00039, which is the clear advantage of a rule-based algorithm over machine learning methods. The four most critical features identified were *hidden_nbf*, *volume_share*, *effect_on_price*, and *traded_money*, indicating that spoofing instances can be distinguished from normal trading to a large extent based only on these features.

6.2 Future Work

Firstly, active learning could be employed to enhance the generalizability of models and address the challenge posed by the limited amount of spoofing data. While commonly used in supervised settings, active learning can also be adapted for unsupervised contexts. In such cases, a human expert would be involved, reviewing the most anomalous instances and iteratively retraining the model. The more frequently the models are used, the more data they will be trained on, continuously improving their performance. The primary challenge with active learning is achieving sufficient initial performance to justify its use from the start.

Secondly, rather than striving to make a model 100% generalizable, an alternative approach would be to develop a specific model for each type of stock. This method would allow for the bypassing of aggregated features in favor of utilizing raw features from the limit order book data, such as price, volume, and size of the order book. This strategy would enable the models to learn more specific patterns, potentially making it easier to detect spoofing. However, a significant drawback is that multiple models would need to be developed and maintained, requiring a trade-off between performance and the number of models employed.

Finally, to improve the results, especially those of the autoencoder, the models should have been trained on more data. The quality and amount of data required by deep learning methods, such as an autoencoder, is generally high [26]. Therefore, additional training could potentially enhance the autoencoder's effectiveness in detecting anomalies. Additionally, deep learning models are more sensitive to the choice of hyperparameters, such as loss function, and the architecture structure, than simpler models, such as OC-SVM and IForest [26]. It would, therefore, be interesting to see if the performance of the hybrid models was improved if this was explored to a greater depth for the autoencoder.

References

[1] S. Khodabandehlou and S. Alireza Hashemi Golpayegani, "Market manipulation detection: A systematic literature review," *Expert Systems with Applications*, vol. 210, p. 118330, 2022. doi: https://doi.org/10.1016/j.eswa.2022.118330. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422014555 [Pages 1, 2, and 6.]

[2] Nasdaq, "Financial market definition," 2024, accessed: 2024-04-30. [Online]. Available: https://www.nasdaq.com/glossary/f/financial-market [Page 1.]

[3] ——, "Trading definition," 2024, accessed: 2024-04-30. [Online]. Available: https://www.nasdaq.com/glossary/t/trading [Page 1.]

[4] R. James, H. Leung, and A. Prokhorov, "A machine learning attack on illegal trading," *Journal of Banking & Finance*, vol. 148, p. 106735, 2023. doi: https://doi.org/10.1016/j.jbankfin.2022.106735. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378426622003156 [Page 1.]

[5] K. Golmohammadi, O. R. Zaiane, and D. Díaz, "Detecting stock market manipulation using supervised learning algorithms," in *2014 International Conference on Data Science and Advanced Analytics (DSAA)*, 2014. doi: 10.1109/DSAA.2014.7058109 pp. 435–441. [Pages 1 and 61.]

[6] C. Poutré, D. Chételat, and M. Morales, "Deep unsupervised anomaly detection in high-frequency markets," Available at SSRN: http://dx.doi.org/10.2139/ssrn.4502662, 2023. [Pages 1, 14, 19, 61, and 67.]

[7] E. Kobre, "Spoofing market manipulation cases set stage for more enforcement," 2023, available: https://news.bloomberglaw.com/us-law-week/spoofing-market-manipulation-cases-set-stage-for-more-enforcement. [Page 2.]

[8] B. L. Do and T. J. Putnins, "Detecting layering and spoofing in markets," Available at SSRN: https://ssrn.com/abstract=4525036 or http://dx.doi.org/10.2139/ssrn.4525036, 2023. [Pages 2, 4, 7, 8, 19, and 67.]

[9] J. Zhai, Y. Cao, and X. Ding, "Data analytic approach for manipulation detection in stock market," *Rev Quant Finan Acc*, vol. 50, pp. 897–932, 2018. doi: https://doi.org/10.1007/s11156-017-0650-0 [Pages 2, 10, 20, and 67.]

[10] X. Tao, A. Day, L. Ling, and S. Drapeau, "On detecting spoofing strategies in high frequency trading," 2020. [Pages 4, 9, 10, and 19.]

[11] Nasdaq, "Market order definition," 2024, accessed: 2024-04-30. [Online]. Available: https://www.nasdaq.com/glossary/m/market-order [Page 5.]

[12] J. A. Sirignano, "Deep learning for limit order books," *Quantitative Finance*, vol. 19, no. 4, pp. 549–570, 2019. doi: 10.1080/14697688.2018.1546053. [Online]. Available: https://doi.org/10.1080/14697688.2018.1546053 [Page 5.]

[13] Nasdaq, "Nasdaq nordic market model 2023:06," 2023. [Online]. Available: https://www.nasdaq.com/docs/2023/07/03/Nasdaq-Nordic-Market-Model-2023_06_.pdf [Page 5.]

[14] Y. Wu, M. Mahfouz, D. Magazzeni, and M. Veloso, "Towards robust representation of limit orders books for deep learning models," 2022. [Page 5.]

[15] E. J. Lee, K. S. Eom, and K. S. Park, "Microstructure-based manipulation: Strategic behavior and performance of spoofing traders," *Journal of Financial Markets*, vol. 16, no. 2, pp. 227–252, 2013. doi: https://doi.org/10.1016/j.finmar.2012.05.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1386418112000377 [Pages 8 and 9.]

[16] A. Bloomenthal, "Market maker definition: What it means and how they make money," 2023. [Online]. Available: https://www.investopedia.com/terms/m/marketmaker.asp [Page 10.]

[17] Nasdaq, "Quote stuffing definition," 2024. [Online]. Available: https://www.nasdaq.com/glossary/q/quote-stuffing [Page 10.]

[18] J. Chen, "What is high-frequency trading (hft)? how it works and example," 2023. [Online]. Available: https://www.investopedia.com/terms/h/high-frequency-trading.asp [Page 10.]

[19] G. Shorter and R. S. Miller, "High-frequency trading: Background, concerns, and regulatory developments," Report crediting UNT Libraries Government Documents Department, University of North Texas Libraries, UNT Digital Library, Tech. Rep., June 2014. [Online]. Available: https://digital.library.unt.edu/ark:/67531/metadc332977/ [Page 10.]

[20] C. Aliferis and G. Simon, *Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI*. Cham: Springer International Publishing, 2024, pp. 477–524. ISBN 978-3-031-39355-6. [Online]. Available: https://doi.org/10.1007/978-3-031-39355-6_10 [Page 11.]

[21] X. Ying, "An overview of overfitting and its solutions," *Journal of Physics: Conference Series*, vol. 1168, no. 2, p. 022022, feb 2019. doi: 10.1088/1742-6596/1168/2/022022. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1168/2/022022 [Page 11.]

[22] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *J. Mach. Learn. Res.*, vol. 11, p. 2079–2107, 8 2010. [Page 12.]

[23] M. Mcmullen, "What are features in machine learning and why it is important?" 2019, available: https://cogitotech.medium.com/what-are-features-in-machine-learning-and-why-it-is-important-e72f9905b54d. [Page 13.]

[24] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, dec 2017. doi: 10.1145/3136625. [Online]. Available: https://doi.org/10.1145/3136625 [Page 13.]

[25] S. Ozdemir and D. Susarla, *Feature Engineering Made Easy: Identify unique features from your dataset in order to build powerful machine learning systems*. Packt Publishing, 2018. ISBN 9781787286474. [Online]. Available: https://books.google.se/books?id=rNRJDwAAQBAJ [Page 13.]

[26] W. Hilal, S. A. Gadsden, and J. Yawney, "Financial fraud: A review of anomaly detection techniques and recent advances," *Expert Systems with Applications*, vol. 193, p. 116429, 2022. doi: https://doi.org/10.1016/j.eswa.2021.116429. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417421017164 [Pages 14, 15, 36, 37, 60, 62, 63, 66, 67, and 70.]

[27] S. Mazzanti, "Isolation forest: The anomaly detection algorithm any data scientist should know," 2021, available: https://towardsdatascience.com/isolation-forest-the-anomaly-detection-algorithm-any-data-scientist-should-know-1a99622eec2d. [Page 16.]

[28] M. Mohammed and M. Telek, "Anomaly detection using combination of autoencoder and isolation forest," pp. 25 – 30, 02 2023. doi: 10.3311/WINS2023-005 [Pages 16, 21, and 33.]

[29] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008. doi: 10.1109/ICDM.2008.17 pp. 413–422. [Pages 17 and 38.]

[30] H. J. Shin, D.-H. Eom, and S.-S. Kim, "One-class support vector machines—an application in machine fault detection and classification," *Computers & Industrial Engineering*, vol. 48, no. 2, pp. 395–408, 2005. doi: https://doi.org/10.1016/j.cie.2005.01.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835205000100 [Page 17.]

[31] V. R. Khazaie, "Solving challenges in deep unsupervised methods for anomaly detection," *Electronic Thesis and Dissertation Repository*, 2022. [Online]. Available: https://ir.lib.uwo.ca/etd/8564 [Page 17.]

[32] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *CoRR*, vol. abs/1802.06360, 2018. [Online]. Available: http://arxiv.org/abs/1802.06360 [Page 17.]

[33] C. Gomes, Z. Jin, and H. Yang, "Insurance fraud detection with unsupervised deep learning," *Journal of Risk and Insurance*, vol. 88, no. 3, pp. 591–624, 2021. doi: https://doi.org/10.1111/jori.12359. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/jori.12359 [Pages 18 and 40.]

[34] M. Carletti, M. Terzi, and G. A. Susto, "Interpretable anomaly detection with diffi: Depth-based feature importance of isolation forest," *Engineering Applications of Artificial Intelligence*, vol. 119, p. 105730, 2023. doi: https://doi.org/10.1016/j.engappai.2022.105730. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197622007205 [Pages 18, 32, and 40.]

[35] A. Tellaeche Iglesias, M. . Campos Anaya, G. Pajares Martinsanz, and I. Pastor-López, "On combining convolutional autoencoders and support vector machines for fault detection in industrial textures," *Sensors*, vol. 21, no. 10, 2021. doi: 10.3390/s21103339. [Online]. Available: https://www.mdpi.com/1424-8220/21/10/3339 [Page 20.]

[36] X. Zhang, P. Wei, and Q. Wang, "A hybrid anomaly detection method for high dimensional data," *PeerJ Comput Sci*, vol. 21, no. 10, 2023. doi: 10.7717/peerj-cs.1199 [Page 20.]

[37] M. Jeragh and M. AlSulaimi, "Combining auto encoders and one class support vectors machine for fraudulant credit card transactions detection," in *2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2018. doi: 10.1109/WorldS4.2018.8611624 pp. 178–184. [Page 20.]

[38] A. Bhandari, "Feature scaling: Engineering, normalization, and standardization," 2024, available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/. [Page 28.]

[39] Scikit-learn, "Comparing anomaly detection algorithms for outlier detection on toy datasets." [Online]. Available: https://scikit-learn.org/0.20/auto_examples/plot_anomaly_comparison.html [Page 33.]

[40] TensorFlow, "The sequential model." [Online]. Available: https://www.tensorflow.org/guide/keras/sequential_model [Page 33.]

[41] Scikit-learn. [Online]. Available: https://scikit-learn.org/stable/ [Page 33.]

[42] H. Torabi, S. L. Mirtaheri, and S. Greco, "Practical autoencoder based anomaly detection by using vector reconstruction error," *Cybersecurity*, vol. 9, 2023. doi: https://doi.org/10.1186/s42400-022-00134-9 [Page 33.]

[43] S. Givnan, C. Chalmers, P. Fergus, S. Ortega-Martorell, and T. Whalley, "Anomaly detection using autoencoder reconstruction upon industrial motors," *Sensors*, 2022. doi: 10.3390/s22093166 [Page 33.]

[44] Google Developers, "Classification: Accuracy." [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy [Page 35.]

[45] J. Bohutska, "Anomaly detection - how to tell good performance from bad," 2021, available: https://towardsdatascience.com/anomaly-detection-how-to-tell-good-performance-from-bad-b57116d71a10. [Page 35.]

[46] F. Angiulli, F. Fassetti, and L. Ferragina, "Latentout: an unsupervised deep anomaly detection approach exploiting latent space distribution," *Mach Learn*, vol. 112, pp. 4323–4349, 2023. doi: 10.1007/s10994-022-06153-4 [Page 61.]

TRITA-EECS-EX-2024:285