# Universiteit Leiden

# Computer Science

## Supervised Outlier Detection in Financial Regulatory Data


Name: Bernard van den Boom
Date: 06/07/2018

1st supervisor: dr. C.J. Veenman
2nd supervisor: dr. F.W. Takes
External supervisor: dr. I.P.P. van Lelyveld


MASTER'S THESIS


Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

# Abstract

The surge in financial regulatory data over the last decade has led to a situation where it has become impossible to look at the data by hand. Therefore, it is important to be able to process the data in a structural manner, to facilitate central banks in their goal of ensuring financial stability. One aspect of this, is finding entities that show out of the ordinary trading behavior. We propose a supervised outlier detection method that uses probabilities from multiple One-vs-Rest gradient boosted models to detect outlying parties. To do this, we aggregate the transaction-level data first. Then, we add both non-network based features and propose using the relationships between parties in the data to produce network-based features. We find that in our case using network-based features only slightly increases performance. At the same time, using both types of features, we are able to process the data set to generate a list of outliers, which regulators can then take a closer look at.

# Contents

# 1 Introduction

The financial world is a complex world generating huge streams of data ranging from trading data to market data to regulatory data. Among other things, this financial world is complex because of its interwoven structure and numerous products. Banks, for example, have increasingly complicated products to sell to their customers. The years following the global financial crisis of 2008 have facilitated an influx of new regulatory data as financial institutions are now required to report their over-the-counter (OTC) trading activities. One of these specific type of contracts, where there is no need for an exchange, but where parties can directly agree on a contract, is an interest rate swap contract. In this agreement, two parties agree to exchange interest rate payments between the two. This might be beneficial if one party can get a better offer for an interest rate than the other and/or one party might be more willing to speculate on interest rates. The interest rate swaps (IRS) market, just as other over-the-counter markets, now needs to comply with regulation that requires them to report 26 fields of specific information about themselves and 59 more fields of common data about the transaction (ESMA, 2018). The size and details of the reported transactions provide opportunities to systematically analyze the data set to get insights into the market that has been a historically opaque market.

### 1.0.1 Goal and Problem

It is not easy to maintain and preserve financial stability. Although financial stability encompasses many aspects and is a complex issue, the over-the-counter markets can have a significant impact on financial stability because of the sheer size of the markets. Traditional methods such as looking at the data by hand are impossible and regulators struggle to get a grip on reporting data, with new data coming in every day. It is clear, there is a need for ways to look at the regulatory dataset in a more structural way.

This thesis will give insight into the unique data set, which has only been introduced for a number of years and is still being updated and improved. Out of the many interesting research questions that can emerge from the data set, one of these questions involves whether the sector that is reported for a counterparty matches up with their trading behavior. For example, do counterparties that are labeled a pension fund behave similar to other counterparties labeled as pension funds? Finding parties that behave other than the sector they have been assigned to, might be interesting parties to take a better look at.

### 1.0.2 Contributions

In essence, we are searching for counterparties assigned to one sector that report trading behavior more consistent with another sector as to advise regulators to take a closer look at these deviating counterparties. We propose a new way of detecting this specific type of *outliers*. At the same time, we solve the problem of not being able to handle the large amounts of regulatory data by hand. In addition, we evaluate to what extent the trading relationships between counterparties can be used as features to improve classification models.

### 1.0.3 Structure

The rest of this thesis will be structured as follows. In Section 2, we will introduce the context of the data set and present related works. Section 3 will elaborate on our methods including the dataset and our proposed outlier detection method. The following section, Section 4, deals with aggregating data and using network information for our models. In Section 5, we run our experiments and discuss our results. Lastly, Section 6 will put forward our conclusions, where we combine our key points and recommend opportunities for future research.

# 2 Background

Some background information about the roots of the financial data and its nature will be presented in this section. We will give a short history of the derivatives market first. Then, we will go into more detail about the specific European regulation that was introduced after the 2008 credit crisis. Finally, the data will be related to the particular problem that is being investigated.

## 2.1 The Over-The-Counter Derivatives Market

Today, the over-the-counter (OTC) derivatives market is almost a $550.000 billion market (Bank for International Settlements, 2017). To put that into perspective, that is roughly seven times the world GDP (The World Bank, 2017). An OTC derivative is a contract between two parties of which its value is derived from some underlying asset and which is not traded on a stock exchange. The two parties usually privately and bilaterally negotiate the contract. The underlying asset can be anything from stocks and equities to currencies and interest rates. The goal of such a private contract is usually to hedge, speculate or shift risks. Accounting for 95% of the derivatives markets (European Commission, 2013), it is no wonder it played a role in the financial crisis of 2008 and continues to play a role in the global economy, although how big a role they played in the crisis remains a fierce topic of debate.

Pre-crisis, the OTC derivatives market was largely unregulated and nontransparent. Trades were not reported to any outside entity and initial margin – property or assets that a party posts to cover losses in case it is in trouble – was often not required. This led to accumulating counterparty exposures. In other words, an OTC derivative participant would often have a number trades with different parties that could get them in trouble in case many of those counterparties would not be able to meet their payment obligations. After all, they deal directly with each other and therefore rely on each other's performance. Formally, they are said to be subject to counterparty risk: "the risk that the other party to an agreement will default" (Wilde, 2004). In sum, the lack of transparency and regulation in the OTC derivatives market led to irresponsible risk exposure.

After the crisis, regulators decided the OTC derivatives market needed regulation, not the least for the role the credit default swap (CDS) market had in the 2008 crisis. Goals were to decrease counterparty risk and reduce the probability a similar crisis could emerge. In the U.S., financial reform legislation was introduced in the form of the Dodd-Frank Wall Street Reform and Consumer Protection Act, an elaborate act aimed at promoting financial stability and improving accountability

and transparency (FINCAD, 2018). In Europe, the European Commission issued the European Market Infrastructure Regulation (EMIR) in late 2012 to increase transparency and reduce operational and credit risk. In essence, the three main aspects of EMIR are the following (Hillis et al., 2013):

- **Clearing** – a trade needs to be done through a central counterparty (CCP) that acts as an intermediary and functions as the buyer and seller of the trade to be able to reconcile all the different trades where possible. In theory, this should allow for more efficient markets.
- **Risk mitigation** – if a OTC derivative is not traded via a CCP, it is required that they meet specific regulations to alleviate their risks.
- **Transaction reporting** – all parties involved in derivative contracts are required to report their trades to designated trade repositories, according to a set standard of data fields, ranging from the name of the counterparty to the size of their positions to collateral.

## 2.2 Interest Rate Swaps

One particular OTC derivative is the interest rate swap (IRS). An interest rate swap is “an agreement between two parties to exchange one stream of interest payments for another, over a set period of time" (*Understanding Interest Rate Swaps*). It is a derivative contract traded over-the-counter, meaning traded without going through a formal exchange such as the the New York Stock Exchange (NYSE) or Euronext. There are different types of interest rate swaps, of which the most common one is what is called the “vanilla" swap. This type of swap consists of exchanging fixed-rate payments for floating-rate payments. A fixed rate is an interest rate that remains the same for the entire contract period. This contract period can typically range from one to thirty years. A floating rate is an interest rate that is dependent on an interest rate that varies over time, usually an interest rate benchmark such as the London Inter-bank Offered Rate (LIBOR). Interest rate swaps were initially undertaken by parties to hedge risks: pay a fixed rate and receive floating-rate payments, and manage portfolios. Nowadays, however, they also act as a market for speculating on future interest rates. For more background information on interest rate swaps and its details, see for example the book by Corb.

A number of terms that will be used throughout this thesis and occur in our data set as important data fields are summarized below:

**Counterparty** An entity that participates in an interest rate swap.

**LIBOR and Euribor** The London Interbank Offered Rate is “the rate of interest at which banks offer to lend money to one another in the wholesale money markets in London" (Bankrate, 2018). Similarly, the Euribor is the Euro Interbank Offered Rate, the European equivalent of the LIBOR. Both benchmark rates have different rates for different periods, e.g. 1 month, 3 months, 6 months and 12 months.

**Legal Entity Identifier (LEI)** An LEI is an alpha-numeric code of length 20, unique to each legal entity/counterparty. It is used to identify entities uniquely and serves as an index in our dataset.

**Fixed Rate** Side or 'leg' of the transaction where the interest rate is fixed. The side of the swap that pays this rate, pays the fixed rate the two counterparties agreed upon when the contract was undertaken, e.g. 3%.

**Floating Rate** Payment leg that is based on some underlying interest rate which might change over time, e.g. it might be 1% in one month and 2% a couple of months later.

**Notional Amount** The value in some currency on which the interest rate payments which are exchanged are based. It stays fixed for the entire contract period.

**Trading Repository** An entity that centrally collects and maintains the records of derivatives (ESMA, 2018).

## 2.3 Related Works

A significant amount of data in the financial domain consists of transactional (granular) data. Our dataset is also of that type, with detailed information on individual interest swap contracts. As we are interested in looking at trading behavior per counterparty and because of the transactional nature of this dataset we need to aggregate the data losing as little as possible information contained in the details. We first present an overview of the research that been done on the dataset. We then proceed by covering how relationships or links in datasets can be exploited. Interest rate swaps, for example, have relationships between the two counterparties that have an agreement. Research proposes methods to utilize this link information in classification models. Lastly, having aggregated the data and having used the network information, we develop classifiers to detect outliers and therefore give an overview of the broad field of outlier detection.

### 2.3.1 Dataset

On the European Market Infrastructure Regulation (EMIR), some basic analysis of the market has been performed, taking either a look at transaction-level data or from the perspective of networks, systemic risk and central clearing. Most of these analysis have been performed by economists and financial experts. Abad et al., for one, provides an elaborate overview of the IRS market (Abad et al., 2016). In the paper, the authors try to improve the understanding of the market and find out possible systemic risks. Fiedor et al., on the other hand, perform network analysis on the centrally cleared IRS derivatives and find that a crucial role is played by the 16 biggest dealers in the world (G16) (Fiedor, Lapschies, and Országhová, 2017). A recent paper by Levels et al. takes a first look at the Dutch credit default swap (CDS) market, specifically examining the flow-of-risk and the impact of Brexit (2018). It also provides a nice overview of the market.

### 2.3.2 Link-based Classification

In computer science, learning from structured data has seen a significant growth in interest. Link-based classification and the use of social network analysis in general has seen an increase in research in different domains, especially in the domain of the world wide web and hypertext mining, but also in criminology (Qin et al., 2005), bio-informatics (Ma'ayan, 2011) and financial analysis (European Central Bank, 2010). Interpreting data in a structured way is especially useful considering many data sets have instances that are not independent, instead have an underlying network structure. Using statistical inference procedures would be naïve in such a case and can lead to incorrect conclusions (Jensen, 1999). One example of a link analysis algorithm is the one proposed in the paper of Yang (Yang, 2002), which is called HITS (Kleinberg, 1998), also known as hubs and authorities, is well known. A similar noteworthy iterative algorithm is Google's PageRank algorithm, which is a model based on the links between web pages (Brin and Page, 1998).

Just as Yang and Kleinberg, Bhagat et al. (Bhagat, Cormode, and Rozenbaum, 2009) and Getoor (Getoor, 2005) also introduce ways of exploiting the link structure of data sets to improve classification performance. Neville and Jensen (Neville and Jensen, 2000), as opposed to using link-based features, use a relational classification technique using simple Bayesian classifiers. Getoor also mentions in her paper the important possible issue that can result from using link-based features for training and test set. Because we use network related features, this issue will be addressed later on in thesis.

### 2.3.3 Outlier Detection

Although research has come up with multiple definitions of outliers, we hold onto the definition of an outlier for the remainder of thesis as "a data point that is significantly different from the remaining data" (Aggarwal and Heights, 2016). Furthermore, we use anomalies and outliers interchangeably, as also suggested by Aggarwal and Heights.

The literature on outlier detection is rich, ranging from unsupervised outlier detection to supervised classification and semi-supervised recognition or detection (Hagberg, Schult, and Swart, 2008). Which approach to use depends on a variety of factors such as the type of data (univariate or multivariate), whether you can fit a distribution to your features (parametric or non-parametric), scalability and speed, and whether you let the model classify outliers or decide on it yourself by using outlier scores. Interpretability is a key aspect in outlier detection and some models have higher interpretability than others.

Often an issue with traditional unsupervised outlier detection, as with many unsupervised learning methods, is how to evaluate the performance of your model. For that reason, literature often resorts to case studies and qualitative evaluation of the detected outliers (Aggarwal and Heights, 2016). In other cases, the ground truth can be derived from an original classification problem. These ground truths allow the researcher to use different evaluation metrics that use threshold values or outlier scores to calculate precision and recall values, draw ROC AUC or precision-recall curves, important evaluation terms we will explain in depth in a later section.

Some of the most popular outlier detection methods include proximity- or distance-based methods (k-nearest neighbors (Altman, 1992) and Local Outlier Factor (Breunig et al., 2000)), neural networks (Hawkins et al., 2002), parametric methods (extreme value analysis) and density-based methods (KDE, DBSCAN (Ester et al., 1996)). Although there are clustering algorithms that can be used to detect outliers, such as DBSCAN (Ester et al., 1996), clustering is generally focused on finding sets or collections of points that are similar, not on finding points that fall outside these collections. Consequently, using clustering to find outliers can be considered equivalent to capturing background noise (Charu C. Aggarwal and Yu, 2001).

Lastly, many of these outlier methods have connections with supervised learning algorithms. For example, isolation forests are decision trees and random forests in their unsupervised analogue, just as replicate neural networks are a special form of neural networks used for outlier analysis (Aggarwal and Heights, 2016). Isolation forests build decision trees by selecting features randomly and splitting on the max and min of the features. Using these algorithms is different from our proposed method but at the same time provide an alternative.

# 3 Methods

In this chapter, we will first discuss how the data was obtained and preprocessed. We also provide some descriptive statistics of the dataset and present the network of counterparties in a network graph. We then summarize our approach and go over the main algorithms, three supervised learning algorithms.

## 3.1 Data

This section will go into detail about where the data is from, how it was preprocessed, how data was aggregated and what its limitations are. The section also includes descriptive statistics of the data and the market as a whole, presenting, for example, the distribution of the type of contracts and the average size of contracts.

### 3.1.1 Obtaining and Preprocessing the Data

The EMIR data that is used in this thesis has been provided by De Nederlandsche Bank N.V. (DNB). As the data is confidential, it cannot be shared and is not publicly accessible. The data was accessed on location at DNB, where it is directly obtained from the authorized trade repositories that counterparties report to. The consequence is that the data contains abundant and inconsistent information that needs to be preprocessed and cleaned. The details of the reporting of trades is established by the European Commission and consists of a maximum of 85 fields (columns). Not all fields are relevant for this research, as they are not applicable to the specific asset class of interest rate swaps.

As can be read from Abad et al., most interest rate derivatives are reported to two trade repositories: DTCC and UnaVista. Recall that trade repositories centrally collect reported trades. UnaVista, has some severe data quality issues making it unfit to include in the final dataset. Fortunately, the Depository Trust & Clearing Corporation (DTCC) and Regis, another trade repository, are more consistent with respect to data quality and the template they use for reporting[1]. So, these two datasets have been merged together and the other trade repositories have not been taken into account, either for data quality issues or their minimal market share. All the results will be based

[1]For information on the trade repositories, see: DTCC: http://www.dtcc.com/, UnaVista: https://www.lseg.com/areas-expertise/post-trade-services/matching-and-reconciliation, Regis: http://www.regis-tr.com/regis-tr/

on the data from these two repositories, which together make up the majority with respect to market share of trade repositories in OTC interest rate derivatives.

For the most part, the cleaning process, by which the raw data from the two trade repositories (DTCC and Regis) is transformed into a ready-to-use dataset, has been adapted from the paper by Levels et al. (2018) and Abad et al. (2016). In summary, important cleaning steps include removing observations with invalid LEI's for either counterparty side, as LEI's allow us to identify parties uniquely. The LEI's that are invalid often seem a wrongly copied identification number (likely a human error). Rows with notional values that are highly unlikely are also removed, just as expired trade deals and inconsistent trades, based on the reporting of both counterparties. Because both counterparties have a reporting obligation, their trade ID might occur twice in the data set. It has been decided only to use one randomly chosen unique report per unique transaction. After cleaning, the dataset does not include duplicate trades.

Apart from cleaning the raw data and getting the data in the right format, the data has been enriched with data from several other public sources, such as the publicly available GLEIF [2] database, based on the unique counterparty identifier, the Legal Entity Identifier (LEI), present in the reporting data. This data consists of counterparty specific information such as the counterparty's home country and official company name, among much more information. Crucial to our task is to get the right sector of a counterparty for as many counterparties present in the data set: more training examples allow for better model training. The counterparty sector field has not been filled in rigorously by a majority of the reporting counterparties. Fortunately, we can retrieve much of the sector information on counterparties from the variety of other sources.

### 3.1.2 Descriptive Statistics

Table 3.1 summarizes some of the main statistics of the EMIR data set. The network consists of 11,105 nodes and 14,149 edges. The degree distribution can be seen in Figure 3.1.

By quite a large margin, most transactions are done by banks, although by only a relatively small number of banks. On the other hand, there are many unique Corporates that enter into only a relatively small number of agreements, as also demonstrated by the degree distribution. Most nodes only have a small number of trading partners. Insurance companies and pension funds (ICPFs) hold the longest contracts with some of the biggest notionals, in line with what is expected of these type of parties: they are often focused on the long-term. The reason the number of unique counterparties does not add up to 11,105 is that the training set only takes unique counterparties from the side of the reporting counterparty. After all, the column values are submitted by that specific counterparty, not by the other counterparty. We call the counterparty reporting the trade 'counterparty a' (cpa) and the other counterparty side 'counterparty b'. A consequence of having information only from one side is that cpb's do not occur in our data set used for training, if they are not also the cpa in at least one of the reported trades. The network in Figure 3.2 shows all unique cpa's and cpb's as nodes; the edges are trades weighted by the sum of notional.

[2] Accessible at `https://www.gleif.org/en`

| Sector | No. of transactions | No. of uniq. counterparties | Average notional in mln. € | Average contract length in years | Sum of notional in bln. € |
|---|---|---|---|---|---|
| Bank | 202,973 | 529 | 40.6 | 11.1 | 8,249 |
| Corporate | 9,255 | 3,806 | 58.7 | 11.3 | 543 |
| Financial | 8,933 | 642 | 31.6 | 11.8 | 283 |
| ICPF | 11,203 | 583 | 50.4 | 19.3 | 564 |
| Other | 1,200 | 230 | 14.7 | 16.5 | 17 |
| Total | 233,564 | 5,790 | - | - | 9,656 |

TABLE 3.1: Descriptives on the used data set

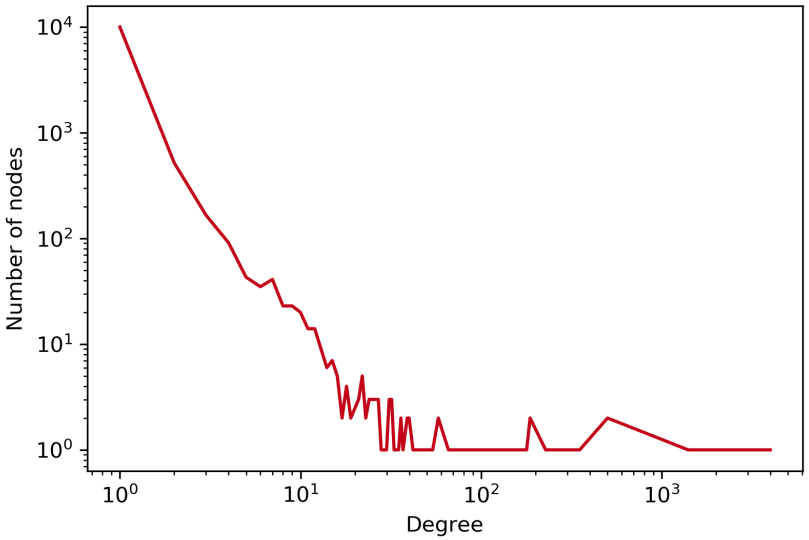


FIGURE 3.1: Degree distribution of the EMIR dataset

The size and color are also based on this notional. We can include cpb's as nodes, because the notional data field is shared between the counterparties. Because of the confidentiality of the data, we cannot go into detail which counterparties occur where in the network graph. Some general things you can see is several big clusters of counterparties, with some seemingly important counterparties in the middle of these clusters. In addition, the center of the network shows some large counterparties grouped together.

## 3.2 Outlier Detection

Outlier detection is about finding points or patterns in data that do not conform to the overall pattern in your data. As mentioned earlier, we define an outlier – or anomaly – as a data point that is considerably different from the other remaining data points. Our goal is to find counterparties that show trading behavior that is more consistent with parties in a different sector from their own. This sector information is available for all counterparties we include in our data set. We are trying to find a counterparty of a sector that is different from remaining counterparties in its sector.

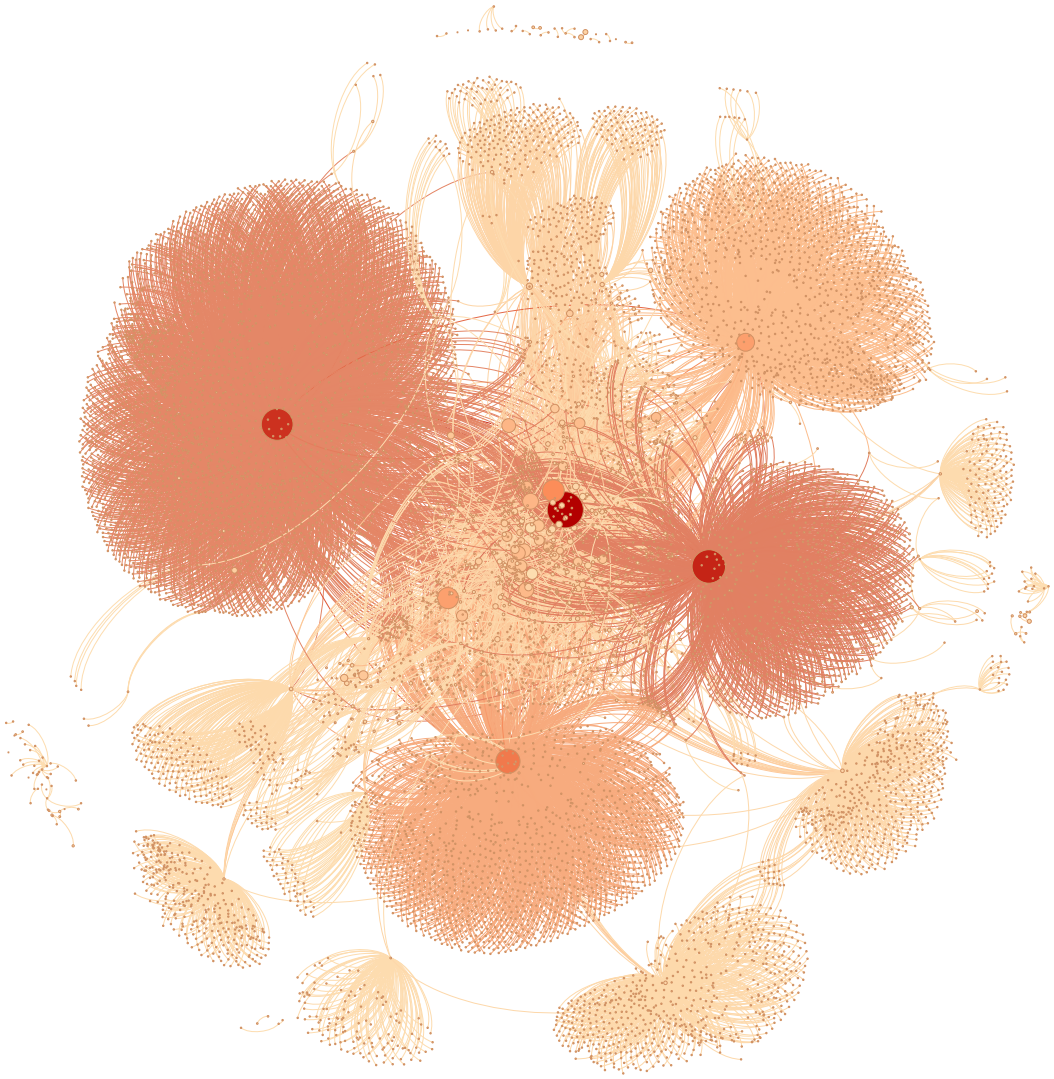

FIGURE 3.2: Network of transactions. Nodes are counterparties, edges are transactions. Nodes and edges are coloured according to (sum of) notionals, just as the node sizes. The graph layout used is Fruchterman-Reingold (Fruchterman and Reingold, 1991).

We propose a *supervised method of finding outliers* to find this subgroup of outliers, instead of using a conventional outlier detection method. One reason we do this supervised is that we know the sector of all counterparties, which allows us to develop a model that trains to characterize a particular sector, such that we can find parties that do not conform to the usual instance of that class. With unsupervised learning methods, we have no information about the labels making it harder to evaluate and validate the outcome of such algorithms.

Another related reason is that many popular and conventional outlier detection methods are often based on proximity. When working in high dimensions and without labels such as in our case (50 features), proximity-based methods quickly lose their meaningfulness (Charu C. Aggarwal and Yu, 2001). Besides, many of the outlier detection methods that have been developed have not been designed to deal with the curse of dimensionality (Bellman, 1961). Put in a basic way, calculated distances in proximity-based methods are going to be very similar, seem more random and most points are going to be away from the center. We need exponentially more data as the number of features increases to be able to accurately generalize and estimate some function $f$ (Samet, 2006).

The supervised learning algorithms we have picked, as opposed to ordinary outlier detection methods, suffer less from high-dimensional data or have the functionality to deal with the issue. There are many specialized ways of dealing with high-dimensional data. One algorithm, the random forest algorithm (Breiman, 2001), for example, decreases the dimensionality by repeatedly using subsets of features. For linear models, regularization can be used to combat issues. The models that we will be experimenting with are described in the next section.

Finally, consider Figure 3.3. Conventional outlier detection methods capture both the blue and red instances as outliers for sector $C_i$. Our method uses the probabilities an instance is $\neg C_i$ given the feature vector, for samples that are in $C_i$. We are not interested in instances that are assigned label $C_i$, but in reality are $\neg C_i$. Instead, we are interested in counterparties that do not look like other counterparties in its sector.

We now propose a different method where we develop multiple classification models and use its resulting probability estimates to decide on outliers. More specifically, we create $n$ One-versus-Rest (OvR) classifiers, one for each unique sector: Bank-vs-Rest, Corporate-vs-Rest, Financial-vs-Rest, ICPF-vs-Rest, Other-vs-Rest. Each of these classifiers tries to learn the best possible separation between one sector and all other sectors.

In our case, we can obtain probability estimates for each pair of the $n$ classifiers: probability $P(C_i|X)$, the probability an instance belongs to class $C_i$, and $P(\neg C_i|X)$, the probability an instance does not belong to class $C_i$, where we call $X$ the feature vector. The $P(\neg C_i|X)$, sorted in descending order, is used to decide on outliers by using a set threshold, such that we have a candidate outlier where:

$$P(\neg C_i|X) \geq threshold$$

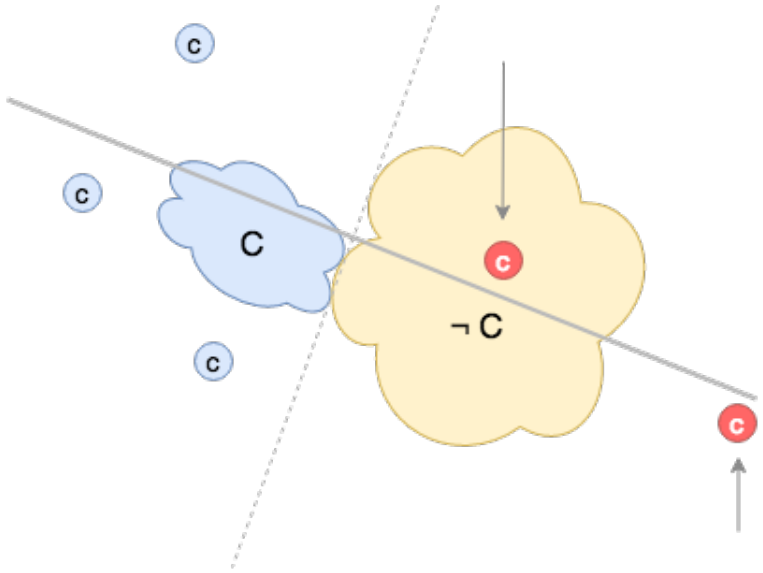


FIGURE 3.3: Linear discriminant. The red instances (circles) are outliers. The blue instances will not be considered in our method, while ordinary outlier detection algorithms tend to include them as outliers.

We keep the list of probabilities $P(\neg C_i|X)$. The threshold can be chosen by the regulator and depends on how well a model is performing and how well probabilities are calibrated.

To summarize, in this section we have proposed using a supervised outlier detection approach to detect counterparties that trade more consistent with a counterparty assigned to a sector other than their own. We do this supervised because (1) we can take advantage of sector labels (2) we avoid introducing an unnecessary variety of problems and complexities that often need to be dealt with when using unsupervised learning methods and (3) because we are interested in a particular type of outliers – not all – as demonstrated by a visualisation.

## 3.3 Supervised Learning Algorithms

There are countless supervised learning algorithms. We experiment with three different algorithms which have shown to be effective in many practical applications: Logistic Regression, Random Forest and Gradient Boosting. We will briefly go over these algorithms in more detail.

### 3.3.1 Logistic Regression

Logistic regression is a popular and fast linear model, which produces well-calibrated probabilities and supports easy regularization. Binary logistic regression requires the dependent variable to be binary, although it can be generalized to multi-class problems, for example by means of multinomial logistic regression. Logistic regression is part of the broader family of generalized linear models (GLMs), just as linear regression does. A GLM consists of three components (Turner, 2008; *Introduction to Generalized Linear Models | STAT 504*):

- Random component – describes the probability distribution of the dependent variable
- Systematic component – or *linear predictor*, written as $\eta_i = \beta_0 + \beta_1 x_{1i} + ... + \beta_p X_{pi}$, a linear function of the independent variables
- Link function – describes the relationship between the first two components, that is, it describes how the expected value of dependent variable $Y$ relates to the linear predictor $\eta_i$.

For binary logistic regression, the random component is the binomial distribution, the linear predictor are the either discrete of continuous independent variables and the link function is the logit function: $\log(p/(1-p))$. As for our specific implementation, we include L2 regularization minimizing the following cost function. This formula is optimized by Pedregosa et al., 2011. It maximizes the log-likelihood and uses the sigmoid function, the inverse of the logit function: $f(x) = \frac{1}{1+e^{-x}}$:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1)$$

with $w$ and $c$ the weights and intercept of the model and $C$ the inverse of regularization strength. $\frac{1}{2} w^T w$ is the regularization term, that helps prevent overfitting by preventing the weights to perfectly fit to the training data. The $C$ parameter is an important parameter that will be tuned while conducting our experiments.

### 3.3.2 Random Forest

Random forests fall under the category of ensemble learning methods, where multiple randomized models are combined into one model. For random forests, this is done by creating multiple randomized decision trees, or a forest of randomized decision trees (Louppe, 2014). Although variations exist, one of the original random forest algorithms was introduced by Ho ("Random decision forests") then extended by Breiman (Breiman, 2001).

Random forests work using a technique called bootstrap aggregating or bagging. With bagging, one takes multiple random samples with replacement from the dataset and use each bootstrap sample as training data for a learning algorithm. This causes some instances to occur multiple times, while others do not occur at all (these are called 'out of bag'). To produce one prediction per instance, the ensemble of algorithms are combined by taking the mean of the predictions of the models (regression) or using a majority vote (classification). Bagging is particularly useful for reducing variance and preventing overfitting. A shortcoming of bagging, however, is that it can lead to decision trees that are highly correlated, i.e. trees can end up looking very similar. This happens in case some or one of the features is a strong predictor for the target.

To overcome issues with correlated trees, it was proposed by Ho and then Breiman to add the step of randomly selecting features. From each bootstrapped sample, we randomly select features – usually a fixed number of features – such that seemingly highly predictive features no longer

dominate the trees. A best split is then calculated for each node on all the selected features. In sum, the random forest algorithm does the following:

For each of the number of trees in the forest:

1. Select a bootstrap sample from the training set
2. Create a decision tree
    (a) For each internal node, randomly select a subset of features
    (b) Split on the best feature

One advantage of using methods built on decision trees and using methods to decide on the best features such as Information Gain and the Gini index is that we can extract variables importances with relative ease. For a random forest, this is slightly more difficult than for a single decision tree, but can, for example, be done by averaging the sum decrease in Gini index for each variable over all the trees (James et al., 2013). For a more elaborate account of the research on Random Forests, refer to for example Hastie, Tibshirani, and Friedman (2009).

### 3.3.3 Gradient Boosting: XGBoost

One of the first gradient boosting algorithm was developed by Friedman (Friedman, 2001). As its name suggests, gradient boosting consists of gradient descent and boosting, as opposed to bagging as used by the Random Forest algorithm. The former technique is an iterative optimization technique that minimizes some cost function by taking steps in the direction of the negative gradient (Google Developers, 2018). The latter term, boosting, generally refers to the idea of using multiple 'weak' learners to create a stronger learner, i.e. learners with high bias and low variance (underfitting).

Gradient boosting is an ensemble learning algorithm, where multiple models are combined to form a strong predictor. Contrary to bagging, boosting does not produce independent trees, instead using the errors of previous trees. For that reason, new trees strongly depend on the previously created trees. Originally, gradient boosting uses regression trees and is designed for regression problems. The regression trees are created sequentially and predictions are made by summing up the scores in each tree. However, in case of a classification problem, we can easily use the logistic function (binary class problem) or softmax (multiclass problem) – similar to logistic regression – to come up with our loss function which can be used to calculate gradients and produce probabilities that an instance belongs to a certain class.

More specifically, the probabilities are computed using the logistic function (James et al., 2013, p. 132):

$$P(C = 1|X) = logistic(x)$$
$$= \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

$X$ contain the features, one column for each leaf node; the betas represent the weights of nodes as computed by the gradient boosting algorithm and is of the same size as $X$. The sum of the weights of the leaf nodes are thus fed into the logistic function to calculate the probability a sample belongs to class $C_i$.

Two popular implementations of gradient boosting are XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017). These have been particularly successful in predictive machine learning competitions and commercial applications (Adam-Bourdarios et al., 2015). We choose XGBoost as our implementation of choice, but early experiments do not show significant differences between the results of several top gradient boosting implementations.

# 4 Aggregation and Feature Creation

Because we are not interested in specific transactions or contracts, but rather in counterparties and their aggregate positions, we need to somehow group the dataset by counterparty (LEI). This has to be done rigorously. For numerical columns, this procedure is different from columns with non-numerical values. For the first, we insert the values into bins to get an accurate feature vector describing the distribution of the values per counterparty. Although the original dataset consists of many columns, the majority contain data that cannot be used as features, because they either contain identification numbers and names, or have too many missing values.

## 4.1 Non-Network Based Features

Some of the columns in the dataset are not numerical. For these columns, we aggregate and take the majority of all the values put together. This is done for cleared (yes/no), transaction type, but also for the country of counterparty a and b, their names and the sector of counterparty b in case they have been labeled differently over multiple transaction. So, for example, if a counterparty with three reported interest rate swap transactions has name “A" in one row and “B" in the other two rows, we report “B" as its name, because it occurs most frequently.

For each counterparty, columns with numerical values such as the notional amounts have to be aggregated as well. We do this by binning all the values. The bin edges are taken logarithmically because of the large differences in notional amounts. To give a simple example, for values between 1 and 100,000 and five bins, we create the following bin edges:

```
np.logspace(0,5,6) = array([10^0, 10^1, 10^2, 10^3, 10^4, 10^5])
```

This allows us to create an equal length feature vector of length the number of bins (five, in our example), for every counterparty. As for columns for which the values do not differ as much, we use simple linearly spaced bin edges.

There are many counterparties that only have one reported trade. As a result, there is only a single entry in the feature vector that describes the size of notional amounts. In addition, these counterparties hold relatively small interest rate swap positions, which puts most instances in the first bin.

## 4.2 Network Based Features

As an important sub-question, we ask ourselves whether we can find that network-based features improve model performance. The idea is that using the trade relationships that counterparties have might help in building a model distinguishing their sectors. It is not unrealistic to think some type of counterparties trade more often with other types of counterparties.

To proceed with some of the *network based* or *link-based features*, we introduce the following notation. Consider a network (graph) $G = (V, E)$, with a set of nodes $V$ and a set of edges $E$, where nodes may be connected via an edge from $u$ to $v$. The number of nodes is $|V|$, the number of edges is $|E|$ and we denote the degree of a node in an undirected graphs as $deg(v)$. The neighborhood of a node $v$ is defined as $N(v) = \{w \in V : (v, w) \in E\}$.

Although the EMIR dataset can be considered a directed graph according to whether the counterparty is on the buy side of the contract – i.e. paying the fixed leg – or on the sell side, we feel there is no need to complicate matters for our purpose by making the graph directed. Besides, the type of transaction feature already captures some of the directionality of the trades.

### 4.2.1 Average Neighbor Degree

One of the network features we add is the weighted average neighbor degree. The weighted average neighbor degree is the weighted average degree of the neighborhood of each node, that is $k^w_{nn,u} = \frac{1}{s_u} \sum_{v \in N(u)} w_{uv} k_v$ (Barrat et al., 2004), with $s_u$ the weighted degree of node $u$, $w_{uv}$ the weight of the edge that links $u$ and $v$ and $N(u)$ the neighbors of node $u$ (Hagberg, Schult, and Swart, 2008). The average neighbor degree is often used to determine the dependencies between the degrees of neighboring nodes (Yao, van der Hoorn, and Litvak, 2017).

### 4.2.2 Clustering Coefficient

To be able to give a numerical value to what extent nodes in the network of counterparties are clustered together, we use the local clustering coefficient. More specifically the local clustering coefficient of a node expresses how close its neighbors are to forming a complete graph. Formally, the local clustering coefficient $C(v)$ for an undirected graph $G$ is defined as:

$$C(v) = \frac{2 \cdot |\{(u, w) \in E : (u, v) \in E \wedge (v, w) \in E\})|}{deg(v) \cdot (deg(v) - 1)} \tag{4.1}$$

In words, this divides two times the edges between the neighbors of $v$ by the maximum number of possible edges between these neighbors. Some example graphs to get an intuition behind the (local) clustering coefficient can be seen in Figure 4.1.

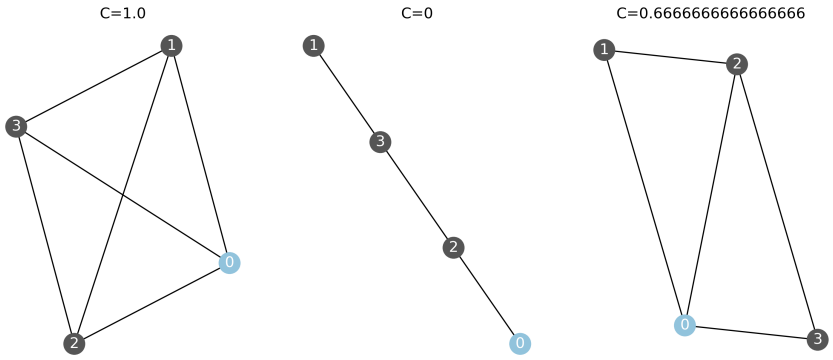


FIGURE 4.1: The clustering coefficient for three different (randomly generated) graphs. Node 0 is connected to all other nodes in the left graph, so C = 1; it is connected to only one other node in the middle graph C=0; in the right graph only 3 and 1 so C = 2/3.

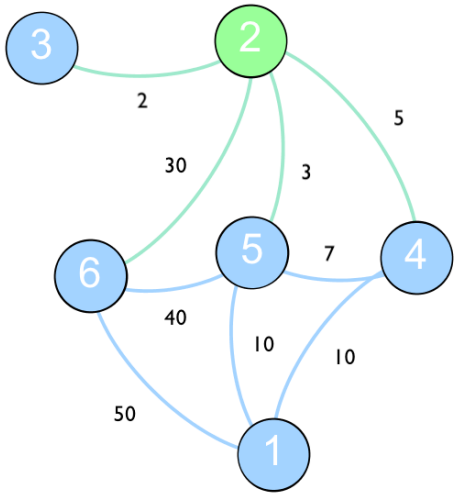


FIGURE 4.2: We apply our short algorithm on node 2. Its neighbors are nodes {3,4,5,6}. For node 6, we get: $30/(30+40+50) = 0.25$. Similarly, for node 3, 4, 5 we obtain 1, $\frac{5}{22}$, $\frac{3}{57}$. The resulting list of values are inputs to a histogram.

### 4.2.3 Relative Weights of Neighbor Edges

Another network feature we include that tries to capture the importance of nodes, is the set of relative weights of neighboring nodes. More specifically, for every node $u$, we iterate over its neighbors $N(u)$ and for every neighbor $v$, divide the edge weight of the edge $u$ to $v$ by the sum of all edge weights of $v$. See Figure 4.2 for a simplified example. Obviously, every neighbor's relative weight is a number between 0 and 1. Just as with other numerical variables, the variable length set of values are binned (10 bins) as to create fixed length feature vectors. Before binning, this *vector of relative weights* for a node describes how important its transactions with another node are for those neighboring nodes, i.e. is the transaction that one node has with a neighboring node important for that neighbor node considering its remaining transactions? If the vector of relative weights contains many values close to 1, you can conclude the concerned node is important for

most of its neighbors in terms of its share in that neighboring nodes' weighted edges.

### 4.2.4 Possible Issues with Using Network-Related Features

As mentioned by Getoor 2005, there might be problems associated with introducing link-based features. For example, when training a classifier using the features, the nodes in the training set might include information about nodes in the test set, because they are neighbors and this neighbor information is used as features. How much of an influence this has, will be briefly tested by taking a sample that is maximally connected. This sample is compared with a 'normal' split.

The exact procedure is as follows: we sample a fixed number of nodes randomly and put them into our test set. We then, for each node in this test set, gather its neighbors and add it to a set that we will use for training. Neighbors that are already in the test set will not be added to this training set. In the end, we will have a test set of size $s_{test}$ and a training set of size $s_{training}$ where all nodes in the training set must be connected to at least one node in the test set. Early experiments fitting multiple classifiers quickly showed there was no siginificant difference in performance. For that reason, we continue to use the network-related features in our further experiments in a later section. We find that in our case predictive performance does not suffer from possible leakage.

# 5 Experiments

## 5.1 Experimental Design

We experiment with different types of algorithms as outlined in Section 3.3. Besides, we also use one training set with network features included and another without the network features. To tune important hyperparameters and evaluate the model's generalization error, we first use nested 5-fold cross-validation, using average precision as a scoring function. Then, we continue using the best classifier to run 10-fold cross-validation on the entire data set and evaluate performance using both average precision as well as ROC AUC. The posterior probabilities that result from this final step, will be interpreted to decide on outlying counterparties. All experiments were run on a laptop with minimal specs. This was easy because the size of the data is manageable. Finally, we continue by explaining our choice of (hyper)parameters, our nested cross-validation procedure and discuss performance evaluation.

### 5.1.1 Choice of Parameters

Because we put instances of one sector against all instances not part of that sector, we end up with imbalanced training and test sets. Fortunately, the Logistic Regression and Random Forest classifiers can take into account class imbalance by using Scikit-learn's `class_weight` variable which uses as weight `n_samples / (n_classes * np.bincount(y))`. For XGBoost, we scale the positive $y$ in a similar fashion using the `scale_pos_weight` parameter, the ratio of positive to negative instances.

Additionally, for Logistic Regression, we tune hyperparameter $C$, its regularization term; for Random Forest we tune the number of trees in the forest; for XGBoost, we search for appropriate values for the maximum depth of trees, the learning rate and the number of boosted trees.

### 5.1.2 Nested Cross-validation

We have used nested cross-validation to tune model parameters, because we cannot use a test set for both evaluating parameter settings and evaluating model performance at the same time. This could lead to overestimating generalization performance of the model (Cawley and Talbot, 2010), i.e. it is no longer a true estimate. Our procedure is summarized in Algorithm 1 and visualised in Figure 5.1.

**Algorithm 1** Nested K-Fold Cross-validation

```
p ← parameter grid
Split dataset S into k sets                                  ▷ Outer CV
for i ← 1, k do
    for each parameter setting in p do                       ▷ Inner CV
        Perform Inner k-fold cross-validation
    end for
    Select best parameter setting p_opt
    Train classifier
    Evaluate performance on test set
end for
Evaluate average performance on test sets
```

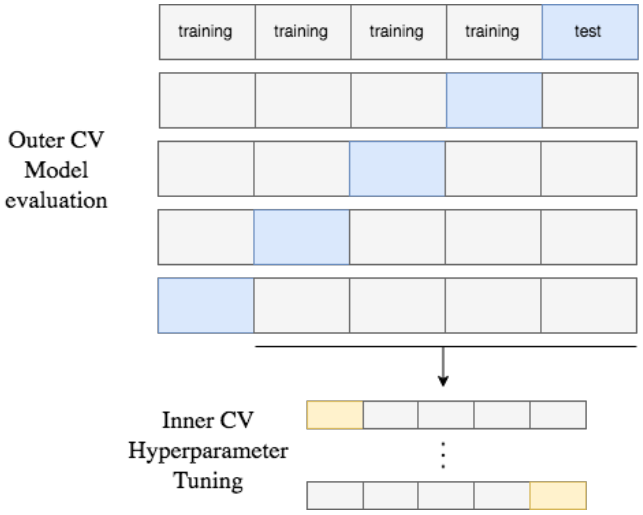


FIGURE 5.1: Nested (5-fold) cross-validation procedure for hyperparameter tuning in the inner loop and model evaluation in the outer loop.

### 5.1.3 ROC and Precision-Recall Curves

Two popular methods to evaluate the performance of a binary classifier, particularly in cases where classes are imbalanced and accuracy is too simplistic, are the Receiver Operating Characteristic (ROC) and its Area Under the Curve (AUC), and Precision-Recall curves. We will discuss both evaluation metrics and plots and describe the relationship between the two.

ROC curves stem from signal detection theory which surfaced during WW2 (Green and Swets, 1974). After a variety of other applications, such as in medicine, Spackman first introduced it for evaluating binary classification algorithms (Spackman, 1989). One way of presenting the results of a classifier is by constructing a confusion matrix, where we present in table form the true positives (TP), false negatives (FN), false positives (FP) and true negatives (TN), comparing the actual labels with the algorithm's predicted labels. See Table 5.1 for a typical confusion matrix. The ROC curve plots two metrics derived from the confusion matrix on its axis: the False Positive Rate ($x$-axis) and the True Positive Rate ($y$-axis). By varying the classifier threshold, we calculate new points to eventually draw the curve. These and other important evaluation metrics are summarized in Table 5.2.

| | | Predicted | |
|---|---|---|---|
| | | 1 | 0 |
| True | 1 | TP | FN |
| | 0 | FP | TN |

TABLE 5.1: Structure of a Confusion Matrix

| Term | Explanation |
|---|---|
| TP | # correctly predicted positive class |
| FP | # incorrectly predicted positive class |
| TN | # correctly predicted negative class |
| FN | # incorrectly predicted negative class |
| Precision | TP/(TP+FP) |
| Recall | TP/(TP+FN) |
| TPR | TP/(TP+FN) |
| FPR | FP/(FP+TN) |

TABLE 5.2: Table of important evaluation terms

The FPR and TPR in a ROC curve are calculated for different values of thresholds, such that it allows you to control both rates and decide on the threshold that fits the application. The ROC curve can be summarized using the Area Under the Curve (AUC), which is a value (between 0 and 1) that summarizes how well a classifier is able to discriminate between a positive and negative observation. Ideally, the ROC curve would be close to the left upper corner (James et al., 2013), i.e. have an AUC close to 1. The metric is particularly suitable for comparing different models, measuring the quality of a model regardless of the decision threshold. A relatively intuitive way to interpret the ROC AUC is to read it as the probability that a random positive instance is ranked higher than a random negative instance (Fawcett, 2006).

An alternative way of evaluating model performance is using the Precision-Recall curve (PR curve). A PR curve plots precision on the $y$-axis and recall on the $x$-axis. Looking at the formulas, note that for constructing a PR curve True Negatives are never used. This has important implications for its use with imbalanced data sets such as where the positive class is severely in the minority. More specifically, Davis and Goadrich (Davis and Goadrich, 2006) state that in an ROC curve "a large change in the number of false positives can lead to a small change in the false positive rate". Because the PR curve uses precision, which uses true positives, not true negatives, it is able to incorporate the imbalance. Just as with ROC curves, the area under the PR curve summarizes the curve and the model's discriminative power for varying threshold. This area is also known as Average Precision (AP). We are using both of the curves and area's under the curves to determine how well each OvR model is able to discriminate one sector from all others. We define the AP as follows (Pedregosa et al., 2011):

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n$$

$P_n$ and $R_n$ are the precision and recall at the $n$-th threshold.

### 5.1.4 Probability Calibration

Before using the posterior probabilities to draw conclusions and use them as some sort of confidence on a prediction, we need to make sure the estimates that our models produce are well-calibrated. This is certainly not always the case. For that reason, we need to first check whether they are, if we want to use the probabilities as some sort of confidence on the prediction.

Classifiers might not produce calibrated probabilities for a variety of reasons. A simple Bayesian classifier, relying on conditional independence assumptions, for example, is proven to produce inaccurate probability estimates (Domingos and Pazzani, 1997). Its probabilities are pushed towards 0 and 1. Other classifiers, such as logistic regression and decision trees, have less biased probability estimates Niculescu-Mizil and Caruana, 2005.

A typical way of checking how well the posterior probabilities of a model are calibrated is by using probability calibration curves, also known as reliability diagrams, where the fraction of positives are plotted against the mean predicted value. Ideally, the curve should be as close to the diagonal as possible.

Another way of measuring the accuracy of our posterior probabilities is to calculate Brier score. We will be using this scoring function to evaluate our probabilities. The Brier score was first introduced in 1950 by Glenn Brier who came up with it to verify weather forecasts (Brier, 1950). Nowadays, it is more broadly applied and often defined as (Upton and Cook, 2014):

$$BS = \frac{1}{n}\sum_{i=1}^{n}(p_i - o_i)^2$$

$p_i$ the probability for instance $i$ and $o_i$ the actual outcome of instance $i$, 0 or 1 in our case. It is equivalent to see it as the mean squared error (MSE) of our estimate. Smaller errors are better, so a smaller Brier score means better calibrated probabilities.

## 5.2 Results & Discussion

In our Resuls & Discussion section, we first discuss how all supervised algorithms performed with regard to ROC AUC and AP. We will proceed by using the probability estimates and showing its distributions from our best performing model to detect outliers. Finally, we extract some important (combination of) features from the model.

### 5.2.1 Model Performance

For all our OvR classifiers, we find the best performance in terms of ROC AUC and AP using the gradient boosting classifier XGBoost, as compared to logistic regression classifiers and random forest classifiers. We also find that different classifiers only improve slightly when network features are

| *i*-vs-Rest | ROC AUC | AP |
|---|---|---|
| Bank | **0.96** | 0.82 |
| Corporate | 0.83 | **0.87** |
| Financial | 0.64 | 0.21 |
| ICPF | 0.88 | 0.71 |
| Other | 0.89 | 0.32 |

TABLE 5.3: Results for each of the OvR classifiers (ROC AUC and Average Precision)

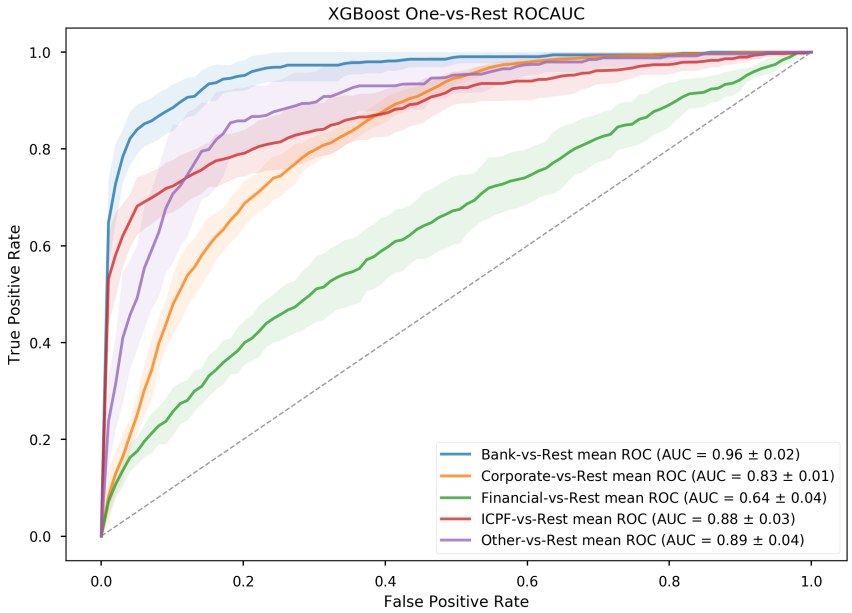


FIGURE 5.2: ROC curves for each XGBoost OvR classifier scored on ROC AUC. Standard deviations come from the results from the different k-folds.

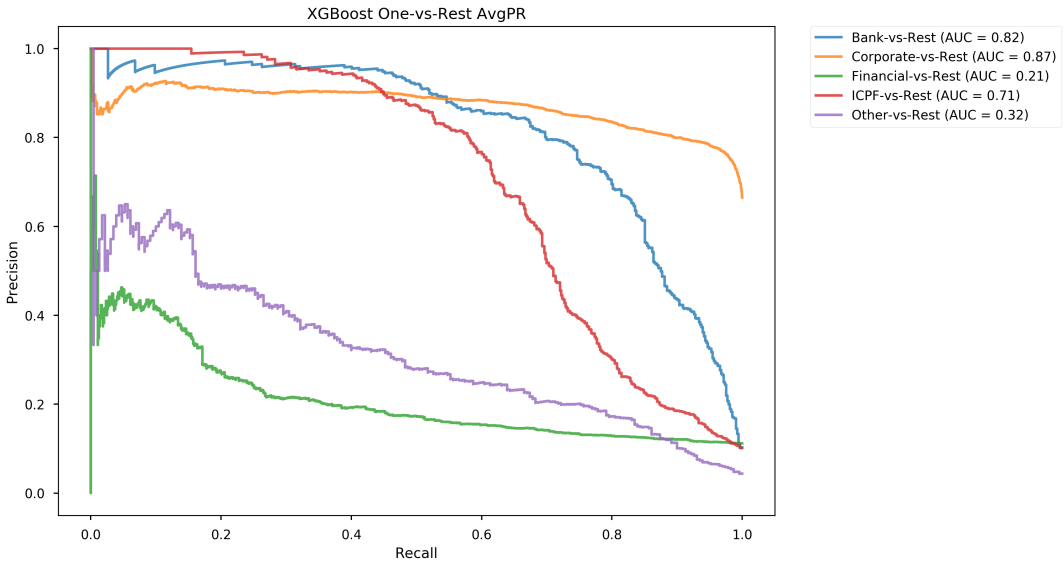


FIGURE 5.3: Precision-recall curves for each XGBoost OvR classifier scored on Average Precision.

added. This suggests other features have a stronger impact on the model than the network features, an aspect we will take a closer look at in Section 5.2.3. Thus, in our case, adding network features only has a modest positive effect on performance.

Figure 5.2 shows the ROC curves for each of the OvR classifiers. Recall that every sector is put against all other sectors. The boundaries show one standard deviation calculated from the ten folds. The best results in terms of ROC AUC are from the Bank-vs-Rest classifier, but ICPF-vs-Rest, Other-vs-Rest and Corporate-vs-Rest show decent ROC AUC values too. The PR curve and AP values, shown in Figure 5.3, indicate the best performance for Corporate-vs-Rest and Bank-vs-Rest. The Other-vs-Rest and Finacial-vs-Rest have low scores. So, Bank-vs-Rest, Corporate-vs-Rest and ICPF-vs-Rest score well on both ROC AUC and AP, while Other-vs-Rest shows high ROC AUC, but low AP and Financial-vs-Rest shows mediocre performance on both metrics. The combined results are also conveniently put into Table 5.3. It is not entirely clear to us why the ROC AUC and AP show such a large difference for the Other-vs-Rest and Financial-vs-Rest classifiers. We suspect this has to do with the imbalance in the resulting data sets and the fact that the ROC takes into account true negatives while PR does not.

### 5.2.2 Using Probabilities for Outlier Detection

Our results demonstrate the different models show varying results with regards to discriminatory power. However, we only use the posterior probabilities, not the predictions, of our OvR classifiers to find outliers, i.e. find counterparties that are more similar to a sector other than their assigned one. For an institution such as a central bank, committed to financial stability, it is interesting to find these outlying parties. These are parties that regulators needs to examine in more detail.

To use the probabilities, we use the Brier score to measure calibration and find that, in general, the classifiers seem reasonably well-calibrated, hence why we decide not to apply techniques such as Isotonic Regression and Platt scaling (Platt, 2000) to improve calibration. Of all the models, the Bank-vs-Rest, ICPF-vs-Rest and Other-vs-Rest show the best Brier scores: 0.041, 0.071, 0.097, respectively. Financial-vs-Rest and Corporate-vs-Rest have Brier scores of 0.207 and 0.160, respectively. We can conclude that, in general, our gradient boosted classifiers turn out to give reasonably well-calibrated probabilities.

The probability distributions for each of the sectors are visualized in Figure 5.4. $P(C_i)$, the probability an instance belongs to a class according to our model, is plotted against the density (KDE). Differences between the histograms are apparent: some histograms almost entirely overlap, such as the Financial-vs-Rest classifier, while others, such as ICPF-versus-Rest and Other-vs-Rest have less overlap. Their shape is unsurprising considering the ROC AUC scores in Table 5.3. For example, the Financial-vs-Rest classifier shows a low ROC AUC score and a low AP score, as well as almost entire overlap. The classifier is simply not able to distinguish the Financials from all others very well. We can, however, still use the probabilities and set a threshold to get counterparties of class $C_i$ for which the probability estimate $P(\neg C_i|X)$ is highest.

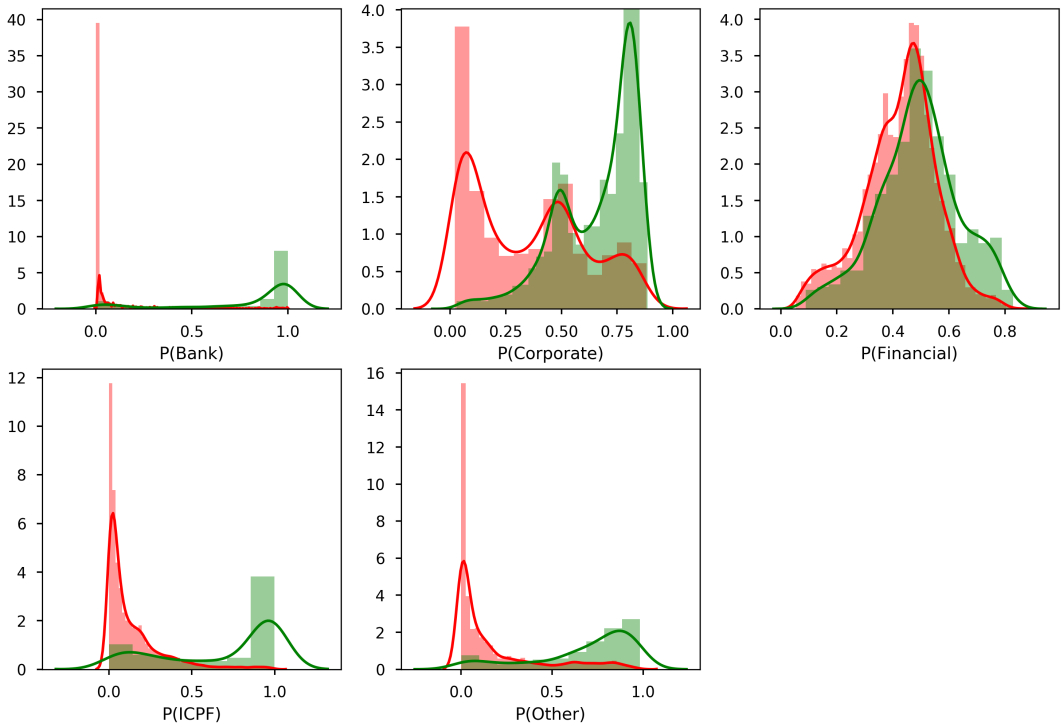


FIGURE 5.4: $P(C_i)$ for samples from $C_i$ (green) and $\neg C_i$ (red). Density, instead of count, is on the $y$-axis.

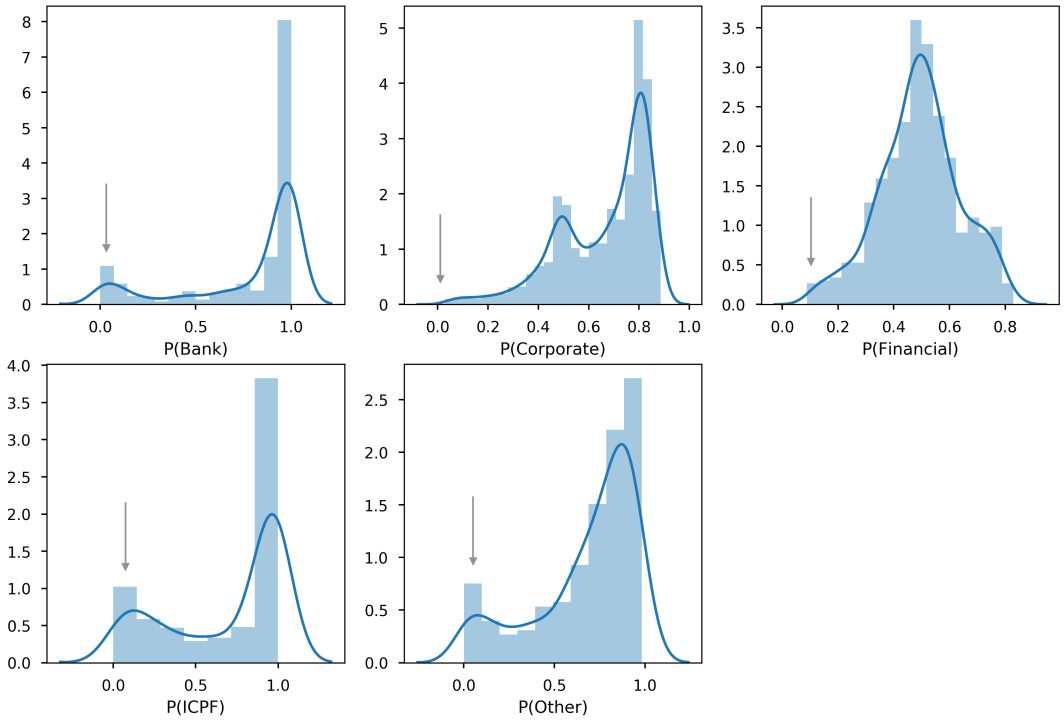


FIGURE 5.5: $P(C_i)$ for samples from $C_i$. Density, instead of count, is on the $y$-axis.

Figure 5.5 shows the probability distribution of instances that are part of $C_i$: $P(C_i|X)$. We include this figure to clarify where we are looking for outliers, instead of showing discriminatory power. We are interested in the left tail (see the arrows) where we find a high probability an instance is not part of a specific sector, while we know it is. In other words, where an instance with label $C_i$

has a $P(\neg C_i|X)$ that is high, or larger than a certain threshold.

To be able to advise regulators which parties to take a closer look at, we proposed ranking the probability estimates and using a threshold to select outliers. The choice of threshold is up to the regulator, because it is often hard to generalize this threshold. After all, we have seen that probabilities can be not as well calibrated and discriminatory power is different for every OvR classifier. In addition, in the specific case of the Dutch central bank, we might filter out non-Dutch counterparties. Additionally, we could put an emphasis on counterparties that have the highest sum of notionals. The idea here is that parties that have a larger outstanding notional amount play a more important role in the financial system.

### 5.2.3 Feature Importances

The gradient boosting model we use allows us to take a look at feature importances based on the individual decision trees (Section 3.3.3). Because we create five models, one for each sector against all other sectors, the feature importances might be different for each model. Before drawing conclusions based on these feature importances, it is important to realize the highest ranked feature importances are not necessarily the most important. A combination of features might in fact be more important to a model, i.e. the combination of features is good at separating two classes.

Overall, we notice that feature importances indeed vary quite a bit for each OvR-classifier. Still, there are some features that show up as top features consistently. Some of these are network-related features, such as some of the vector elements of our relative weight of neighbor edges (Section 4.2.3) and the average neighbor degree weighted by notional. The network-related features seem important, although their inclusion does not improve performance significantly. Not surprisingly, how many trades a counterparty undertakes plays an important role as well: parties with more transactions are more often part of one sector than all other. Find the top ten features for each model in Appendix A. In sum, most models make use from a variety of network-related features, while some also find the country or length of the contract important. In that sense, using network-related features, although not directly having a large impact on performance, seems to be a welcome addition.

To summarize, we found gradient boosting to be performing best in terms of ROC AUC and AP, although performance varies over the different One-vs-Rest classifiers. Having decided on the algorithm, the resulting posteriors – of which we argue the Brier scores are generally sufficiently high and thus sufficiently calibrated – are used to find outlying counterparties. Our proposal is to do this using a threshold to select a number of counterparties that are most likely to be an outlier. Conveniently, gradient boosting also allows us to look at feature importances and we find that some of the network-related features consistently show in the top scored features.

# 6 Conclusion

Analyzing thousands of financial transaction reports coming in on an every day basis cannot be done by hand. We can, however, analyze the granular data in a more systematic way. One important task of monetary authorities, who receive all kinds of trade reporting data, is to ensure financial stability. The entities that report these trades might be of a size that can endanger stability, making it important to monitor their behavior. In this thesis, we have proposed a way of finding the entities in IRS reporting data that do not conform to the behavior you expect from them based on their trading behavior. If a large bank takes on positions more consistent with a hedge fund, this might need a closer look by a regulator. More specifically, we have been looking for counterparties that enter into IRS agreements more typical of counterparties in a sector other than their own and call these outliers. Because we have the sector information and we are looking for a specific type of outliers, we propose a supervised outlier detection method, where we use posterior probabilities to find these outliers.

As features for our supervised method, we have decided to experiment with link-based features, i.e. features that exploit in some way the relationship between counterparties based on their trades. Using these network-based features we have seen that performance improves slightly and overall have high feature importance scores. Our results do not show a strong added value of the network-related features, but future research might be able to provide more convincing examples.

Another interesting aspect to look at in feature research would be to look at trading behavior over time, although you would have to look at a long period of time, since IRS contracts frequently span over multiple years or even decades. This, however, introduces a new challenge: the reporting format has changed quite a bit over time, if the reporting obligation existed at all at that time. On top of that, the IRS data we have used included many fields that were not filled in correctly or only filled in partially, including many potentially valuable fields such as the current market value of the contract. Another opportunity of future research, apart from improving general data quality, lies in enriching the data with more sources, possibly including counterparty specific data such as turnover or number of employees.

With regards to the field of outlier detection, there is room for more research, particularly research into scalable supervised and unsupervised high-dimensional outlier detection methods. From what we have found, the research on supervised outlier detection is scarce, but can prove very useful for a variety of problems if more research is put into the field.

# A Feature Importances XGBoost OvR Classifiers

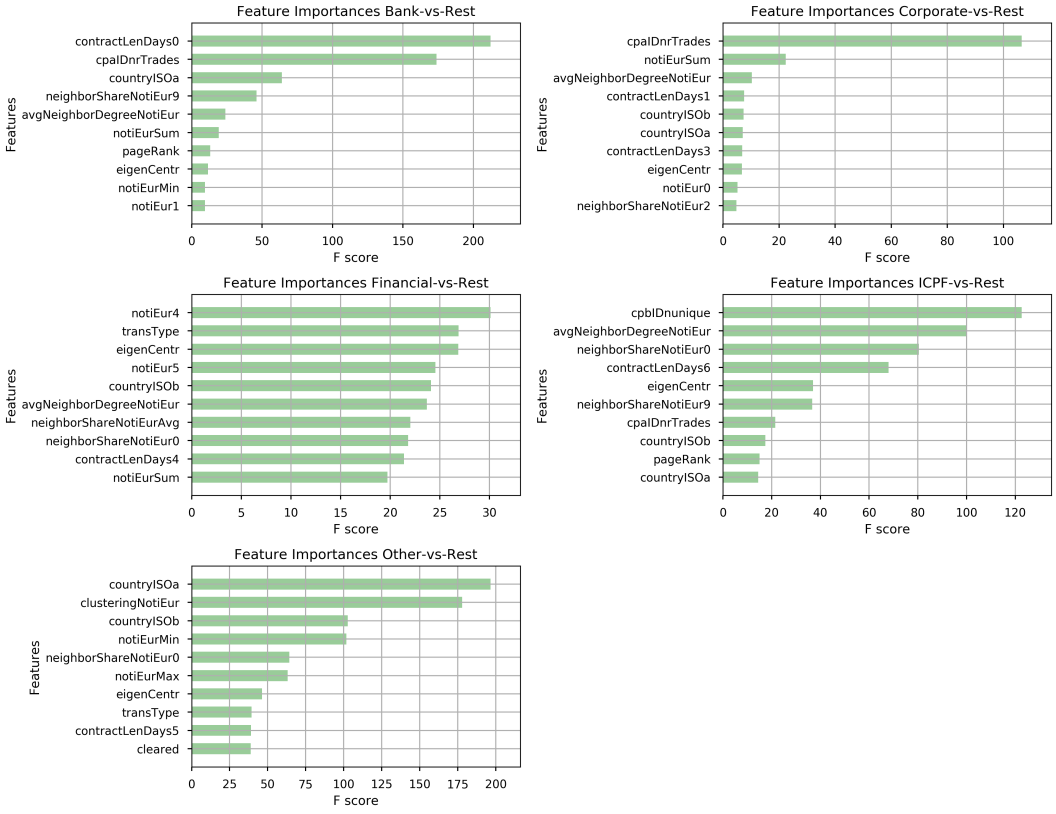


FIGURE A.1: Feature importances for each of the OvR classifiers.

# Bibliography


Abad, Jorge et al. (2016). "Occasional Paper Series Shedding light on dark markets: First insights from the new EU-wide OTC derivatives dataset". In: DOI: 10.2849/878675. URL: https://www.esrb.europa.eu/pub/pdf/occasional/20160922{_}occasional{_}paper{_}11.en.pdf.

Adam-Bourdarios, Claire et al. (2015). "The Higgs boson machine learning challenge". In: 42, pp. 19–55. URL: http://proceedings.mlr.press/v42/cowa14.pdf.

Aggarwal, Charu C and Yorktown Heights (2016). *Outlier Analysis Second Edition*. ISBN: 9783319475783. DOI: 10.1007/978-3-319-47578-3. URL: http://rd.springer.com/book/10.1007/978-3-319-47578-3.

Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". In: *American Statistician* 46.3, pp. 175–185. ISSN: 15372731. DOI: 10.1080/00031305.1992.10475879.

Bank for International Settlements (2017). *Semiannual OTC derivatives statistics at end-December 2011*. URL: https://www.bis.org/statistics/derstats.htm?m=6{\%}7C32{\%}7C71http://www.bis.org/statistics/derstats.htm (visited on 01/31/2018).

Bankrate (2018). *1 Month LIBOR | 30 Day Libor Rate Current Interest Rates Index One*. URL: https://www.bankrate.com/rates/interest-rates/1-month-libor.aspx (visited on 05/20/2018).

Barrat, A et al. (2004). "The architecture of complex weighted networks." In: *Proceedings of the National Academy of Sciences of the United States of America* 101.11, pp. 3747–52. ISSN: 0027-8424. DOI: 10.1073/pnas.0400087101. URL: http://www.ncbi.nlm.nih.gov/pubmed/15007165http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC374315.

Bellman, R.E. (1961). "Adaptive control processes: A guided tour". In: *Princeton University Press* 28, pp. 1–19. arXiv: arXiv:1302.6677v1. URL: http://arxiv.org/abs/1302.6677.

Bhagat, Smriti, Graham Cormode, and Irina Rozenbaum (2009). "Applying Link-Based Classification to Label Blogs". In: Springer, Berlin, Heidelberg, pp. 97–117. DOI: 10.1007/978-3-642-00528-2_6. URL: http://link.springer.com/10.1007/978-3-642-00528-2{_}6.

Breiman, Leo (2001). "Random Forests". In: *Machine Learning* 45.1, pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324.

Breunig, Markus M. et al. (2000). "LOF: Identifying Density-Based Local Outliers". In: *Proceedings of the 2000 Acm Sigmod International Conference on Management of Data*, pp. 1–12. ISSN: 01635808. DOI: 10.1145/335191.335388. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.8948.

Brier, Glenn W. (1950). "VERIFICATION OF FORECASTS EXPRESSED IN TERMS OF PROBABILITY". In: *Monthly Weather Review* 78.1, pp. 1–3. ISSN: 0027-0644. DOI: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2. URL: http://journals.ametsoc.org/doi/abs/10.1175/1520-0493{\%}281950{\%}29078{\%}3C0001{\%}3AVOFEIT{\%}3E2.0.CO{\%}3B2.

Brin, Sergey and Lawrence Page (1998). "The anatomy of a large scale hypertextual Web search engine". In: *Computer Networks and ISDN Systems* 30.1/7, pp. 107–17. ISSN: 01697552. DOI: 10.1.1.109.4049. arXiv: 1111.6189v1. URL: http://infolab.stanford.edu/pub/papers/google.pdf.

Cawley, Gavin C and Nicola L C Talbot (2010). "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation". In: *Journal of Machine Learning Research* 11, pp. 2079–2107. URL: http://jmlr.org/papers/volume11/cawley10a/cawley10a.pdf.

Charu C. Aggarwal and Philip S Yu (2001). "Outlier detection for high dimensional data". In: *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pp. 37–46. ISSN: 01635808. DOI: http://doi.acm.org/10.1145/375663.375668. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.895.53{\&}rep=rep1{\&}type=pdf.

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost : Reliable Large-scale Tree Boosting System". In: *arXiv*, pp. 1–6. ISSN: 0146-4833. DOI: 10.1145/2939672.2939785. arXiv: 1603.02754. URL: http://arxiv.org/abs/1603.02754http://dx.doi.org/10.1145/2939672.2939785.

Corb, Howard. (2012). *Interest rate swaps and other derivatives*. Columbia Business School, p. 599. ISBN: 978-0-231-53036-1. URL: http://opac.ub.tum.de/InfoGuideClient.tumsis/start.do?Login=wotum{\&}Query=540={\%}22978-0-231-53036-1{\%}22.

Davis, Jesse and Mark Goadrich (2006). "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pp. 233–240. ISBN: 1595933832. DOI: 10.1145/1143844.1143874. arXiv: 1609.07195. URL: https://www.biostat.wisc.edu/{~}page/rocpr.pdfhttp://portal.acm.org/citation.cfm?doid=1143844.1143874.

Domingos, Pedro and Michael Pazzani (1997). "On the Optimality of the Simple Bayesian Classifier under Zero-One Los". In: *Machine Learning* 29.1, pp. 103–130. ISSN: 08856125. DOI: 10.1023/A:1007413511361. arXiv: 05218657199780521865715. URL: http://link.springer.com/article/10.1023/A:1007413511361.

ESMA (2018). *Trade Repositories*. https://www.esma.europa.eu/supervision/trade-repositories. (Accessed on 06/04/2018).

Ester, Martin et al. (1996). "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231. ISSN: 09758887. DOI: 10.1.1.71.1980. arXiv: 10.1.1.71.1980. URL: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf.

European Central Bank (2010). *Recent advances in modelling systemic risk using network analysis*, p. 32. ISBN: 9789289906111. DOI: 10.2866/57570. URL: https://www.ecb.europa.eu/pub/pdf/other/modellingsystemicrisk012010en.pdf?ee625137cfa8445525242272e4dab584.

European Commission (2013). “Regulations”. In: *Official Journal of the European Union* L52, pp. 1–10.

Fawcett, Tom (2006). “An introduction to ROC analysis”. In: *Pattern Recognition Letters* 27.8. ROC Analysis in Pattern Recognition, pp. 861 –874. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.10.010. URL: http://www.sciencedirect.com/science/article/pii/S016786550500303X.

Fiedor, Pawel, Sarah Lapschies, and Lucia Országhová (2017). “Networks of counterparties in the centrally cleared EU-wide interest rate derivatives market”. In: *ESRB Working Paper* 54. DOI: 10.2849/86362. URL: https://www.esrb.europa.eu/pub/pdf/wp/esrb.wp54.en.pdf?52afac2c22258efce835a6d07e19d606.

FINCAD (2018). *Dodd-Frank and EMIR | Derivatives Risk Management Software & Pricing Analytics | FINCAD*. URL: http://www.fincad.com/resources/learning-resources/regulations/dodd-frank-and-emir (visited on 02/01/2018).

Friedman, Jerome H. (2001). “Greedy function approximation: A gradient boosting machine”. In: *Annals of Statistics* 29.5, pp. 1189–1232. ISSN: 00905364. DOI: DOI10.1214/aos/1013203451. arXiv: arXiv:1011.1669v3.

Fruchterman, Thomas M J and Edward M Reingold (1991). “Graph Drawing by Force-directed Placement”. In: 21.1, pp. 1129–1164. URL: http://citeseer.ist.psu.edu/viewdoc/download;jsessionid=19A8857540E8C9C26397650BBACD5311?doi=10.1.1.13.8444{\&}rep=rep1{\&}type=pdf.

Getoor, Lise (2005). “Link-based Classification”. In: *Advanced Methods for Knowledge Discovery from Complex Data*. New York: Springer-Verlag, pp. 189–207. DOI: 10.1007/1-84628-284-5_7. URL: http://link.springer.com/10.1007/1-84628-284-5{_}7.

Google Developers (2018). *Reducing Loss: Gradient Descent | Machine Learning Crash Course |*. URL: https://developers.google.com/machine-learning/crash-course/reducing-loss/gradient-descent (visited on 05/23/2018).

Green, David Marvin and John A. Swets (1974). *Signal detection theory and psychophysics*. R.E. Krieger Pub. Co, p. 479. ISBN: 9780882751399. URL: https://trove.nla.gov.au/work/10637047?selectedversion=NBD655908.

Hagberg, Aric a., Daniel a. Schult, and Pieter J. Swart (2008). “Exploring network structure, dynamics, and function using NetworkX”. In: *Proceedings of the 7th Python in Science Conference (SciPy2008)*. Vol. 836, pp. 11–15. ISBN: 3333333333. URL: http://www.osti.gov/energycitations/product.biblio.jsp?osti{_}id=960616.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). “Random Forests”. In: Springer, New York, NY, pp. 587–604. DOI: 10.1007/978-0-387-84858-7_15. URL: http://link.springer.com/10.1007/978-0-387-84858-7{_}15.

Hawkins, Simon et al. (2002). “Outlier Detection Using Replicator Neural Networks”. In: *Data Warehousing and …* Pp. 170–180. ISSN: 0302-9743. DOI: 10.1007/978-3-540-74553-2. URL: http://link.springer.com/chapter/10.1007/3-540-46145-0{_}17.

Hillis, Brett et al. (2013). *EMIR – Summary of Key Requirements*. URL: https://www.reedsmith.com/en/perspectives/2013/06/emir--summary-of-key-requirements (visited on 02/05/2018).

James, Gareth et al. (2013). *An Introduction to Statistical Learning*. Vol. 103, p. 441. ISBN: 978-1-4614-7137-0. DOI: 10.1007/978-1-4614-7138-7. arXiv: arXiv:1011.1669v3. URL: http://www-bcf.usc.edu/{~}gareth/ISL/ISLRFirstPrinting.pdfhttp://link.springer.com/10.1007/978-1-4614-7138-7.

Jensen, David (1999). "Statistical challenges to inductive inference in linked data". In: *Seventh International Workshop on Artificial Intelligence and Statistics*, pp. 98–101. URL: https://pdfs.semanticscholar.org/6732/8fa41b5048e8efd1335a025c9a2b0beeff03.pdfhttp://www.aaai.org/Papers/Symposia/Fall/1998/FS-98-01/FS98-01-010.pdf{\%}5Cnhttp://scholar.google.com/scholar?hl=en{\&}btnG=Search{\&}q=intitle:Statistical+Challenges+to+Inductive+Infe.

Ke, Guolin et al. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. URL: https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.

Kleinberg, Jon M (1998). "Authoritative Sources in a Hyperlinked Environment *". In: URL: http://www.cs.cornell.edu/home/kleinber/auth.pdf.

Levels, Anouk et al. (2018). "CDS market structure and risk flows: the Dutch case". In: URL: https://www.dnb.nl/en/binaries/WorkingPaperNo.592{_}tcm47-375866.pdf.

Louppe, Gilles (2014). "Understanding Random Forests: From Theory to Practice". In: ISSN: 00219606. DOI: 10.13140/2.1.1570.5928. arXiv: 1407.7502. URL: http://arxiv.org/abs/1407.7502.

Ma'ayan, Avi (2011). "Introduction to network analysis in systems biology." In: *Science signaling* 4.190, tr5. ISSN: 1937-9145. DOI: 10.1126/scisignal.2001965. URL: http://www.ncbi.nlm.nih.gov/pubmed/21917719http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3196357.

Neville, Jennifer and David Jensen (2000). "Iterative classification in relational data". In: *Learning Statistical Models from Relational Data*, pp. 42–49. DOI: 10.1.1.23.2875. URL: http://www.aaai.org/Papers/Workshops/2000/WS-00-06/WS00-06-007.pdfhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.23.2875{\&}rep=rep1{\&}type=pdf{\%}5Cnhttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.23.2875{\%}5Cnhttp://www.aaai.org/Papers/Workshop.

Niculescu-Mizil, Alexandru and Rich Caruana (2005). "Predicting good probabilities with supervised learning". In: *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pp. 625–632. ISBN: 1595931805. DOI: 10.1145/1102351.1102430. URL: http://portal.acm.org/citation.cfm?doid=1102351.1102430.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Penn State. *Introduction to Generalized Linear Models | STAT 504*. URL: https://onlinecourses.science.psu.edu/stat504/node/216/ (visited on 05/09/2018).

PIMCO. *Understanding Interest Rate Swaps*. URL: https://nl.pimco.com/en-nl/resources/education/understanding-interest-rate-swaps.

Platt, John (2000). "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods". In: 10.

Qin, Jialun et al. (2005). "Analyzing Terrorist Networks : A Case Study of the Global Salafi Jihad Network". In: *Lecture Notes in Computer Science* 3495, pp. 287–304. ISSN: 0009921X. DOI: 10.1007/11427995_24.

Samet, By Hanan (2006). "Foundations of Multidimensional and Metric Data Structures". In: *Order A Journal On The Theory Of Ordered Sets And Its Applications* di.August, pp. 0–1. ISSN: 0033-295X.

Spackman, Kent A. (1989). "Signal detection theory: valuable tools for evaluating inductive learning". In: *Proceedings of the Sixth International Workshop on Machine Learning*. Elsevier, pp. 160–163. ISBN: 1558600361. DOI: 10.1016/B978-1-55860-036-2.50047-3. URL: http://linkinghub.elsevier.com/retrieve/pii/B9781558600362500473http://portal.acm.org/citation.cfm?id=102118.102172{\&}coll=GUIDE{\&}dl=GUIDE{\&}CFID=8689407{\&}CFTOKEN=49107713.

The World Bank (2017). *GDP (current US$) | Data*. URL: https://data.worldbank.org/indicator/NY.GDP.MKTP.CD (visited on 01/31/2018).

Tin Kam Ho. "Random decision forests". In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. IEEE Comput. Soc. Press, pp. 278–282. ISBN: 0-8186-7128-9. DOI: 10.1109/ICDAR.1995.598994. URL: http://ieeexplore.ieee.org/document/598994/.

Turner, Heather (2008). "Introduction to Generalized Linear Models". In: URL: http://statmath.wu.ac.at/courses/heather{_}turner/glmCourse{_}001.pdf.

Upton, Graham and Ian Cook (2014). *A Dictionary of Statistics*. Oxford University Press. ISBN: 9780199679188. DOI: 10.1093/acref/9780199679188.001.0001. URL: http://www.oxfordreference.com/view/10.1093/acref/9780199679188.001.0001/acref-9780199679188.

Wilde, Tom (2004). "Counterparty Risk". In: URL: http://www.nasdaq.com/investing/glossary/c/counterparty-risk.

Yang, Kiduk (2002). "Combining text- and link-based retrieval methods for Web IR". In: *NIST SPECIAL PUBLICATION SP*, pp. 609–618. ISSN: 1048-776X. URL: https://pdfs.semanticscholar.org/85f7/7e6b547facd91dcaa261024ddd0ed3bb5d01.pdfhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.7314{\&}rep=rep1{\&}type=pdf.

Yao, D., P. van der Hoorn, and N. Litvak (2017). "Average nearest neighbor degrees in scale-free networks". In: *ArXiv e-prints*. arXiv: 1704.05707 [math.PR].