CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Stock Prediction from Unlabeled Press Releases using Machine Learning and Weak Supervision

Master's thesis in Computer Science and Engineering

MARKUS INGVARSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Stock Prediction from Unlabeled Press Releases using Machine Learning and Weak Supervision

Markus Ingvarsson

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Stock Prediction from Unlabeled Press Releases using Machine Learning and Weak Supervision
MARKUS INGVARSSON

Supervisor: Carl-Johan Seger, Department of Computer Science and Engineering
Examiner: Moa Johansson, Department of Computer Science and Engineering

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2021

Stock Prediction from Unlabeled Press Releases using Machine Learning and Weak Supervision

MARKUS INGVARSSON
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This thesis examines the effect of press releases on the Nordic stock market. A weak supervision approach is utilized to estimate the short-term effect on stock returns given press releases of different categories. By utilizing the data programming framework as implemented in the Snorkel library, approximately 24% of all press releases are categorized into a set of 10 distinct categories. Further, a collection of machine learning models for stock price prediction is developed, where simulation is conducted to determine how press releases may be used to forecast stock price movement. Stock price prediction is performed for large stock price movements and for stock price direction, where the result shows that the best performing model achieves a 53% F1-score and 54% accuracy respectively for the tasks. Finally, it appears that the labeled press releases can be used to increase the predictability of stock movements in the Nordic stock market.

Keywords: stock prediction, press releases, weak supervision, machine learning, nordic stock market

Acknowledgements

First and foremost, I want to thank my supervisor, Carl-Johan Seger, whose guidance and patience have been invaluable throughout the thesis. I also want to thank my examiner, Moa Johansson, who has been a great mentor to me throughout my master's thesis and stay at Chalmers.
Special thanks to Rasmus Holm and the MFN and Modular Finance team for providing me with press release data for this thesis. Also, a special thanks to George Bol and Henrik Bergström for offering the exceptional Börsdata API and for never hesitating to explain the available features.
I also want to show my sincerest appreciation to my lecturers Selpi Selpi and Richard Johansson, Nasdaq Stockholm's Ludvig Westerdahl, and my great friend Daniel, who has been of great support when determining design choices in the practical work.
I would like to thank my Canadian host mom Liz, my peer review partner Rahwa Araya, my opponent Elmar Aliyev and my soon-to-be colleagues Niclas, Torbjörn, Julie, Joakim, Zenobia, and Lars for providing me with insightful feedback on the thesis report.
Last but not least, I want to thank my family and friends for their continuous encouragement throughout this year.
Thank you.

Markus Ingvarsson, Gothenburg, June 2021

Contents

List of Figures **xi**

List of Tables **xiii**

List of Acronyms **xv**

1 Introduction **1**
1.1 Purpose . . . 1
1.2 Goal . . . 2
1.3 Limitation . . . 2

2 Theory **3**
2.1 Stock Markets . . . 3
2.1.1 Efficient Market Hypothesis . . . 4
2.1.2 Other Market Hypotheses . . . 4
2.2 Machine Learning . . . 5
2.2.1 Support Vector Machine . . . 6
2.2.2 Extreme Gradient Boosting . . . 8
2.2.3 Supervised Learning without Labeled Data . . . 9
2.2.4 Model Selection and Optimization . . . 10
2.2.4.1 Hyperparameter Tuning . . . 11
2.2.4.2 Regularization and Pruning . . . 11
2.2.5 Evaluation of ML Models . . . 12

3 Literature Review **15**
3.1 Stock Market Prediction Through News Events and NLP . . . 15
3.2 Snorkel . . . 16

4 Method **19**
4.1 Data Collection . . . 19
4.1.1 Press Releases . . . 19
4.1.2 Market Data . . . 21
4.2 Press Release Labeler . . . 22
4.2.1 Model Selection . . . 23
4.2.2 Optimization and Evaluation . . . 24
4.3 Stock Price Predictors . . . 24
4.3.1 Model Selection . . . 25

4.3.2 Data Preprocessing . . . 27
4.3.3 Optimization and Evaluation . . . 28
4.4 Tools . . . 28
4.4.1 Python . . . 28
4.4.2 Scikit-learn and XG-Boost . . . 28
4.4.3 Snorkel and NLP Libraries . . . 29

5 Results **31**
5.1 Press Release Labeler . . . 31
5.2 Press Release Impact Analysis . . . 32
5.3 Stock Price Predictors . . . 36

6 Discussion **39**
6.1 Press Release Labeler . . . 39
6.2 Press Release Impact Analysis . . . 40
6.3 Stock Price Predictors . . . 41
6.4 Future Work . . . 42

7 Conclusion **43**

Bibliography **45**

A Labeling Functions **I**
A.1 Helper Functions . . . I
A.2 Share Buyback Labeling Function . . . VI
A.3 Acquisition Labeling Function . . . VIII
A.4 New Deal Labeling Function . . . X
A.5 Earnings Report Labeling Function . . . XII
A.6 New Key Person Labeling Function . . . XIII
A.7 Lost Key Person Labeling Function . . . XV
A.8 Inside Buy Labeling Function . . . XVIII
A.9 Inside Sell Labeling Function . . . XXI
A.10 Award Labeling Function . . . XXIV
A.11 Divestment Labeling Function . . . XXVI
A.12 The Label Matrix . . . XXVIII

B Stock Market Prediction **XXXI**
B.1 Stock Exchanges . . . XXXI
B.2 Countries . . . XXXI
B.3 Sectors . . . XXXI
B.4 Lines of Businesses . . . XXXII

List of Figures

2.1 The fundamental workflow for a machine learning classifier. A training set of data is used to calibrate the parameters of a machine learning system. Additionally, a collection of test data is utilized to evaluate the model's performance. 5
2.2 A hyperplane with a margin for an SVM trained on a binary classification tasks. The data points that support the margin are called support vectors. 6
2.3 Linear separability in the new feature space is feasible by applying a kernel function ϕ to the preexisting input space, thus creating a new, higher-dimensional feature space. 7
2.4 A decision tree to predict whether or not a person survived the Titanic disaster. The `sibsp` feature represents the number of siblings and spouses that the person had on board. 8

4.1 The number of available companies over time. Data from roughly 500 companies were available in 2011, increasing as more companies have entered the stock market. 19
4.2 The number of companies over the number of available press releases (PR). Roughly 300 companies have issued between 201-500 press releases, and about 150 companies have issued less than six press releases. 20
4.3 The distribution of the press releases (PR) over time. 20
4.4 The number of press releases (PR) per size group, where the size group is based on the press releases' content length in words (w) and pages (p). 21
4.5 The distribution of companies per country. 22
4.6 The distribution of companies per sector. 22
4.7 The distribution of returns given by stocks provided a press release. . 25

5.1 The average returns per day over 20 days before issued press releases of some certain category. 33
5.2 The pre-event trade for the set of different press release categories. The white dot represents the category's average stock return. 34
5.3 The post-event trade for the set of different press release categories. The white dot represents the category's average stock return. 34

6.1 The number of labeled press releases (PR) for the ten defined press release categories. 39

6.2 The SHAP plot of the $XGB_{\Phi,pd}$ model. 41
6.3 The SHAP plot of the $XGB_{\Phi,lp}$ model. 41

List of Tables

4.1 The press release categories that were labeled through weak supervision. Only one category can be assigned for each press release. . . . 23
4.2 The XG-Boost models' hyperparameters for the Stock Price Direction Predictor (pd) and Large Price Movement Predictor (lm). 26
4.3 The SVM models' hyperparameters for the Stock Price Direction Predictor (pd) and Large Price Movement Predictor (lm). 26
4.4 The input features of the XG-Boost and SVM stock price predictor models. Stock Returns of N Days is the return that the stock has given during the N previous trading days before the press release. The Stock Press Release Reaction feature is $p_{o,j,t}/p_{c,j,t-1}$, assuming that the press release was issued in the morning before opening hours. The volatility features are measured over the previous 32 days, where the standard deviation measures volatility. R^2 measures how well the stock that issued the press release is following the benchmark index. 27

5.1 The number of labeled press releases per category. 52293 out of 218757 press releases were actively labeled as a category, where the remaining 76% were `unknown` (UN). 31
5.2 A set of performance metrics for the Press Release Labeler. 32
5.3 Financial metrics of press releases of different categories for a pre-event trade. 35
5.4 Financial metrics of press releases of different categories for a post-event trade. 36
5.5 The evaluation metrics for the Stock Price Predictor models for price direction movement. XGB_{Φ} reaches the accuracy for predicting stock price direction out of all the classifiers. 36
5.6 The evaluation metrics for the Stock Price Predictor models of large price movements. XGB_{Φ} reaches the highest F1-score for predicting large stock price movements out of all the classifiers. 37
5.7 The accuracy (denoted as A) of the $XGB_{pd,\Phi}$ and $D_{pd,s}$ model for stock price direction prediction. 38
5.8 The F1-score for the $XGB_{lm,\Phi}$ and $D_{lm,s}$ model for large stock price prediction. 38

A.1 The label matrix which shows the coverage, overlap and conflict of each LF. XXVIII

B.1 Number of stocks per stock exchange. XXXI
B.2 Number of stocks per country. XXXI
B.3 Number of stocks per sector. XXXI
B.4 Number of stocks per line of business. XXXII

List of Acronyms

ANN Artificial neural network.
API Application programming interface.

CNN Convolutional neural network.
CV Cross-validation.

EMH Efficient market hypothesis.

FN False negative.
FP False positive.

LF Labeling function.

ML Machine learning.

NLP Natural language processing.

P/E Price-to-earnings ratio.

RegEx Regular expression.

SHAP Shapley additive explanations.
SVC Support vector classifiers.
SVM Support vector machines.
SVR Support vector regression.

TN True negative.
TP True positive.

XG-Boost Extreme gradient boosting.

1
Introduction

Forecasting is an intriguing task practiced throughout many disciplines, where qualified predictions are made based on historical data. The field of economics actively works with forecasting, and the stock market is one area where predictions of future returns of stocks are made regularly. A stock is a fractional ownership in a company that entitles the owner to a portion of the business. One may value a stock based on the company's business performance, where the financial track record and future earnings will determine what price most investors are willing to pay for a stock. Others may only evaluate the price trend and trading volume of a stock when considering how much they are prepared to pay for it.
The pricing of stocks in the market is said to be efficient [1]. However, stock movements were studied in the 1970s, where it was found that a correlation exists between price changes in stocks and significant news events in the American stock market [2].
Publicly traded companies use press releases to interact with investors and the general public. These press releases often come in textual format and are read mainly by investors who understand their business and the overall stock market. Recent advances in the field of natural language processing (NLP) have demonstrated that text can be categorized using machine learning [3]. However, text categorization often requires a large amount of labeled data, which can be a daunting task as the amount of available information steadily increases in the world. It is not uncommon for domain knowledge expertise to be required for data labeling, further complicating the task.
Weak supervision has been proposed as a solution to the limited access to labeled data [4]. Weak supervision is a technique where data is labeled by an ensemble of different, often noisy, and imprecise labeling techniques. This thesis will examine how this technique can be used to label press releases by category, such that one can further analyze the impact that they may have on stock returns. Further, the impact that these categories have on stock predictability will also be tested. The majority of research in this field has focused on the American and Chinese stock markets [5], but how do the Nordic stock markets' stock prices correlate with press releases?

1.1 Purpose

This thesis investigates how press releases affect a stock's share price in the Nordic stock market. The questions investigated are:

- How do different types of news events affect future stock returns?
- Can stock price movements be predicted in the stock market given a press release? In the affirmative case, can various types of press releases be utilized to increase the predictability of a stock's future price movement?

The research questions will be resolved through weak supervision and data analysis methods, as well as the application of machine learning techniques that have been shown to be effective in previous research.

1.2 Goal

The thesis' objective is to develop a Press Release Labeler capable of automatically assigning a set of categories to an unlabeled set of press releases without access to previously labeled data. The relationship between stock prices and various types of press releases is further examined.
Two stock price predictors are developed, which attempt to: (i) predict the binary outcome of whether a stock's price will rise or fall given a press release; and (ii) predict large price movements of stocks given a press release. Any increased predictability given a certain kind of press release is also analyzed and presented.

1.3 Limitation

Only textual data in English will be considered in this thesis. Companies that do not publish their press releases in English will therefore not be considered. As a result of only using data from the Nordic stock market, the conclusions drawn in this thesis may be limited to this market. Finally, because the models will not be deployed in the open market, this thesis will evaluate their performance solely through simulation. The stock predictor models will not attempt to outperform existing trading systems or methods but will instead focus on the predictability of stock price movements in response to news events.

2
Theory

Stock markets are an active area of research, where market efficiency is actively studied. Significant time and effort have also been spent on research in the ML community during recent years. The theory behind the fundamentals used in this report is presented in the upcoming sections.

2.1 Stock Markets

In the modern financial world, stock markets serve as platforms for people to buy and sell stocks, and a primary function of this market system is to reduce the complications associated with stock allocation. A stock share represents ownership of the company that issued the share, and companies often issue shares to raise capital. A company may take a loan from a bank to guarantee sufficient means for an ongoing expansion of its business. However, banks will often charge interest for these loans, given that the loans are approved in the first place. The loans may be substantial in size and used to develop products or solutions that are new to the market, where proof of concepts may yet be unavailable. By issuing shares to a group of investors, a company can raise capital to expand its business, add knowledge and experience to its management team, and avoid bank loan interest payments that could halt its growth. The day the company decides to enter the public stock market, it will ideally have a case that can interest the general public. Sought-after shares will likely sell for a higher price, which will allow the company to raise more capital from fewer issued shares.
A stock's price fluctuates in response to the price of the shares at which buyers are ready to pay and the price at which sellers are prepared to sell. However, the notion is that this price will automatically represent the company's intrinsic worth.
A return on investment from a stock trade will be given once the stock shares are sold. The return of some trade for a company j will be defined by the percentage change of the original investment, more formally as:

$$\bar{x}_{j,t} = \frac{p_{j,t} - p_{j,t-k}}{p_{j,t-k}} \times 100$$

where $p_{j,t}$ is the price of the stock at time t, and $p_{j,t-k}$ was the price at time $t-k$. New information is brought to the market through press releases. We define the information delivered through a press release from a company j at time t to be $\phi_{j,t}$. A press release can be seen as an *event*, which updates the information corpus Φ_T,

defined as:

$$\Phi_T = \{\phi_{a,0}, \ldots, \phi_{k,T}\}$$

which is the set representing all the available information that reflects the companies' $C = a \ldots k$ prices in the stock market up until a time T.
A few different hypotheses regarding how the market reacts to new information are discussed in the following subsection.

2.1.1 Efficient Market Hypothesis

An efficient market is a market where the price of an asset fully reflects the available information of that asset [1]. The efficient market hypothesis (EMH) proposes that the existing market environment is a *fair game*. This theory suggests that an information corpus Φ_t adjusts the affected stock prices directly, suggesting that an updated information corpus Φ_{t+1} is not of interest when predicting stock prices since the fair price is always the current price.
The EMH has been divided into three forms: the so-called *weak*, *semi-strong*, and *strong* form [1]. The different forms provide different levels of the strictness of the EMH, where the idea is that an investor can not outperform the market on a risk-adjusted basis over time.
The weak form suggests that the price of a stock reflects all the available market data. This form implies that using market data, such as historical stock prices, to forecast future price movements is not advantageous. The method to use market data for stock prediction is often referred to as technical analysis.
The semi-strong form proposes that all publicly available information regarding the company will quickly adjust and set the fair price. This form implies that forecasting for non-insiders would be useless since stocks are always priced fairly, with all earnings estimates in mind.
Lastly, the strong form of the EMH suggests that the current fair price of a stock is based not only on publicly available information but also on all private insider information about the company and its operations. This form suggests that even insider trading would not outperform the market on a risk-adjusted basis.

2.1.2 Other Market Hypotheses

In 1977, Sanjoy Basu found that companies with a meager price-to-earnings ratio (commonly known as the P/E ratio) usually give higher returns on average than the high P/E stocks on a risk-adjusted basis [6]. The price-ratio hypothesis was presented, where low P/E stocks are believed to trade at low valuations in relation to their earnings due to consecutive years of poor earnings.
Research in experimental psychology has proposed another hypothesis, the so-called overreaction hypothesis, where it is thought that people may overreact to unanticipated news events [7]. Recent news has in the past tended to overshadow older ones. Historically, stock price movements have shown to be significantly correlated with the following year's earning changes. This behavior has resulted in the suggestion that previous winners and losers (with respect to returns) are thought to underperform respectively overperform the average market returns.

A systematical overshooting of stock prices would indicate a violation of the weak-form efficient market hypothesis since it would imply that one could predict stock prices by looking at market data alone, without using any earnings figures [7]. The empirical study done over a 36-month period concluded that stock portfolios of prior "losers" tended to outperform prior "winners" by roughly 25%, even given the fact that the stock portfolio of the prior "winners" was regarded as notably riskier.

2.2 Machine Learning

Since the stock market is significantly data-driven, it has attracted the use of machine learning (ML) [5]. ML classification is a supervised learning task where the objective is to learn $f : \mathcal{X} \mapsto \mathcal{Y}$ from a training set $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\}$, where $\mathcal{X}$ is defined as the feature space, $\mathcal{Y}$ as our set of available target labels, and D as our training data that consist of input-output data pairs. The end goal of most classifiers is to make accurate predictions of some unseen data where the labels are unknown.

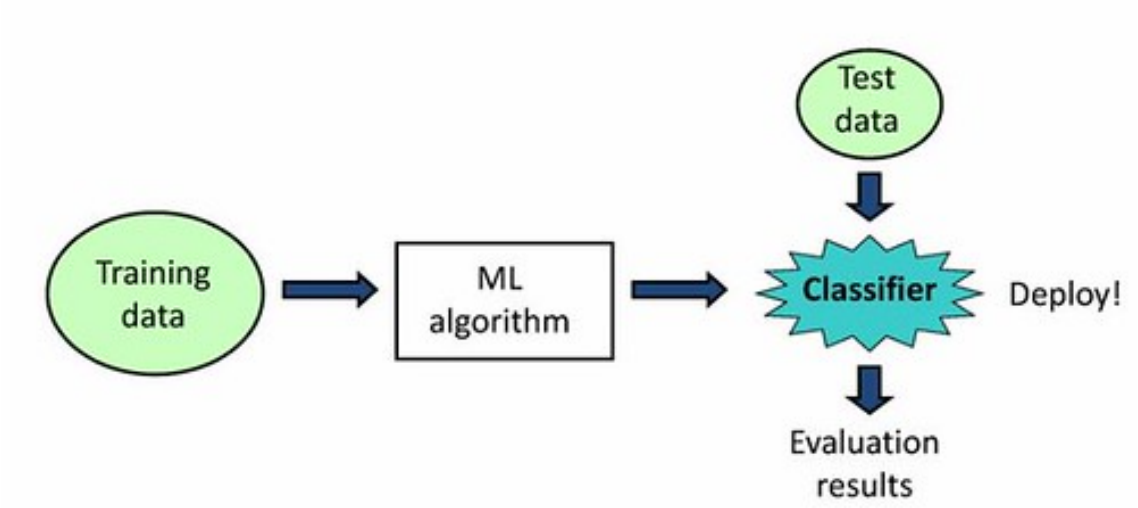

Figure 2.1: The fundamental workflow for a machine learning classifier[1]. A training set of data is used to calibrate the parameters of a machine learning system. Additionally, a collection of test data is utilized to evaluate the model's performance.

ML algorithms partition or group data according to a set of predefined rules to create their final model, where its hyperparameters govern the algorithm's training process. The training process is the process where the ML algorithm tries to model a dataset through the corresponding ML algorithm by optimizing some *objective function*. As the algorithm models the dataset, it adjusts the parameters of the final model to make the most accurate predictions. One of the primary functions of an objective function is to calculate the loss used to evaluate the classifier's performance by counting the number of correct labels predicted. The objective is to minimize

[1]https://search.creativecommons.org/photos/d6c00a36-f028-4500-bcb1-c041e8d69a83

the loss, which can be accomplished by using an optimization function to improve the final model's prediction accuracy or some other valid evaluation metric.
Classification can be done in many settings, but binary classification will be in focus when reviewing the theory in the following subsections.

2.2.1 Support Vector Machine

Support Vector Machine (SVM) is a widely used supervised learning algorithm that can be used for classification tasks [8]. The SVM model uses a non-probabilistic binary linear classifier, which attempts to classify a dataset by splitting the two groups of data in half by some separator, often referred to as the hyperplane.
A hyperplane can also be referred to as Support Vector Classifiers (SVC), where the name comes from the fact that it will use some data points on each side of the margin as support vectors.

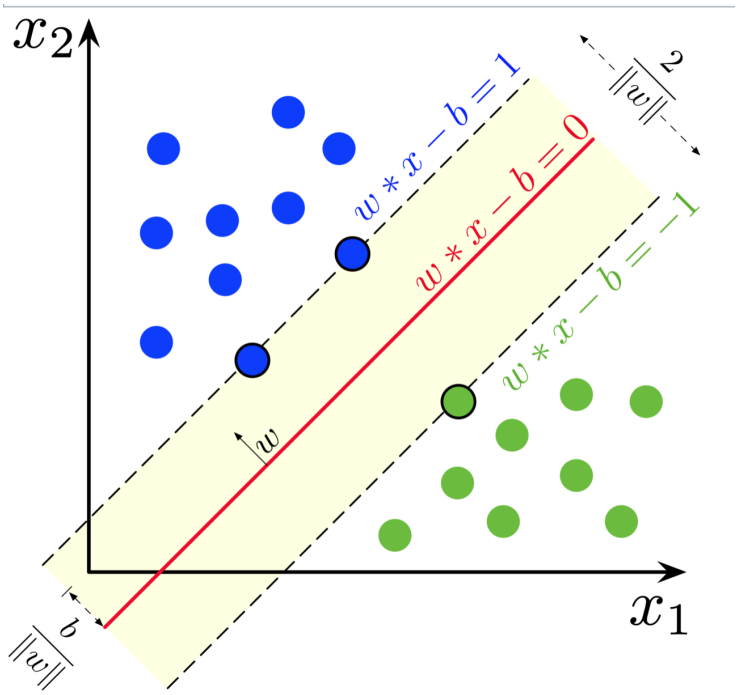

Figure 2.2: A hyperplane with a margin for an SVM trained on a binary classification tasks[1]. The data points that support the margin are called support vectors.

The SVC is selected based on how well it manages to maximize the margin and the separability between the two classes. Therefore, we can state our problem for finding the set of parameters that can minimize the objective function as:

$$\begin{aligned} \underset{\mathbf{w}}{\text{argmin}} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & t_n\left(\mathbf{w}^\top \mathbf{x}_n + b\right) \geq 1, \text{ for all } n \end{aligned}$$

[1] https://commons.wikimedia.org/wiki/File:SVM_margin.png

where t_n is defined as the target label for x_n, and w represents the coefficients (or weights) for the input variables x_1 and x_2. However, in most settings, the data will overlap, so a soft margin is commonly used. We will thus soften our constraint by adding a slack variable ξ_n:

$$t_n \left(\mathbf{w}^\top \mathbf{x}_n + b \right) \geq 1 - \xi_n$$

where the new optimization task becomes:

$$\begin{aligned} \underset{\mathbf{w}}{\text{argmin}} \quad & \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^{N} \xi_n \\ \text{subject to } \xi_n \geq 0 \quad \text{and} \quad & t_n \left(\mathbf{w}^\top \mathbf{x}_n + b \right) \geq 1 - \xi_n \text{ for all } n \end{aligned}$$

Misclassification is thus allowed given that $\xi_n > 1$. One idea to use a soft margin is to allow misclassification in order to improve generalization. The regularization parameter C is used to adjust the width of the margin, which is a topic that will be discussed more in detail in subsection 2.2.4.
A natural shortcoming with linear classifiers is that they can only classify linearly separable data, and linear separability can not always be assumed from data in every setting. However, SVM can get around this problem by using a so-called *kernel function*. A kernel function will rigorously attempt to find an SVC in a higher-dimensional setting by transforming the preexisting input features to a higher dimensional and linearly separable feature space. Three commonly used kernel functions are the *polynomial*, the *linear*, and the *radial basis function* kernel [9].

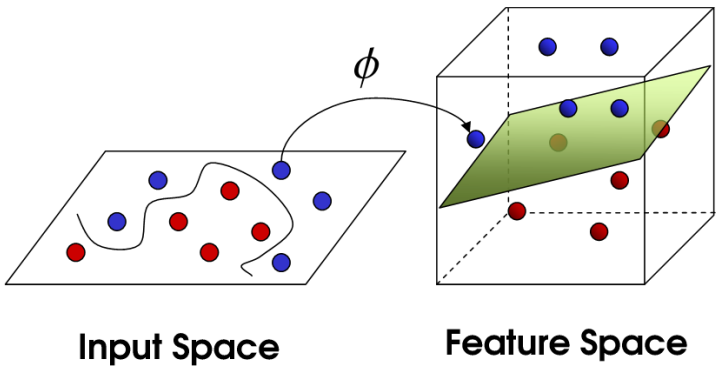

Figure 2.3: Linear separability in the new feature space is feasible by applying a kernel function ϕ to the preexisting input space, thus creating a new, higher-dimensional feature space[1].

[1]https://commons.wikimedia.org/wiki/File:Kernel_yontemi_ile_veriyi_daha_fazla_dimensiyonlu_uzaya_tasima_islemi.png

2.2.2 Extreme Gradient Boosting

The Extreme Gradient Boosting algorithm (XG-Boost) is a computationally efficient and robust ML algorithm based on the gradient boosting algorithm and is practical for classification tasks [10]. The gradient boosting algorithm uses an *ensemble* of *weak classifiers*, where an ensemble is a group of models that cooperatively predicts the target label of some unknown data [11]. A weak classifier is a classifier that can make predictions better than a coin toss but not well enough to be used alone. The weak classifiers often consist of *decision trees* in gradient boosting. Decision trees attempt to split datasets by selecting features that create two distinct partitions, much like the popular spoken parlor game "Twenty Questions." An example of a decision tree is shown in Figure 2.4, where the gender and ages older than nine and a half are the two most determining features on whether a person survived the Titanic disaster.

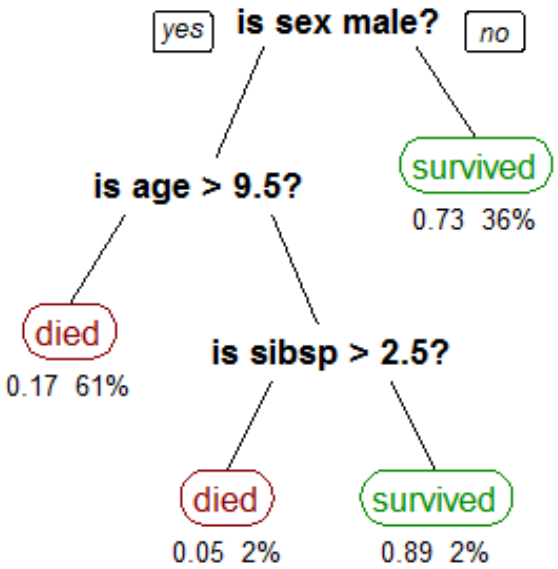

Figure 2.4: A decision tree to predict whether or not a person survived the Titanic disaster[1]. The `sibsp` feature represents the number of siblings and spouses that the person had on board.

After some given number of partitions, the goal is that one will end up with a homogenous partition of some target label. The partitioning of the data is done by some splitting criterion, where XG-Boost utilizes the gain and similarity score. The objective function can be stated as:

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta)$$

[1] https://commons.wikimedia.org/wiki/File:CART_tree_titanic_survivors.png

where the negative log-likelihood is a commonly used loss function $L(\theta)$ for binary classification, which can be defined as:

$$L\left(y_i, p_i\right) = -\left[y_i \log\left(p_i\right) + \left(1 - y_i\right) \log\left(1 - p_i\right)\right]$$

where y_i represents the ground truth label and p_i the predicted target label. The $\Omega(\theta)$ term is the regularization term, which is a topic that will be discussed in greater detail in subsubsection 2.2.4.2.
In XG-Boost and Gradient Boosting, the leaves (i.e., the terminal nodes) of the trees are not filled with target labels as seen in Figure 2.4. Instead, the leaves are filled with pseudo residuals, which measures the difference between the true label and the predicted label. The boosting process simplified can be described as the following: The algorithm initially starts with a single leaf that will predict every target label as 0.5, i.e., the middle ground between the classes 0 and 1. Next, it will create a tree to improve the previously predicted labels. The newly created tree will not predict the target labels as we saw in Figure 2.4. Instead, it will predict the pseudo residuals from the last ensemble, which is (again) not the true target labels but the difference between the trees' predicted labels and the ground truth. The reason for this may not be obvious at first, but the general idea is that each created tree is supposed to contribute to a slightly better prediction than what the previous ensemble of trees did. Thus, we optimize our classifier in a greedy manner by improving it slightly for every tree we create. The predictions of the trees are added together, which is intended to result in a solid end classifier. The contributions from each tree are multiplied by a so-called *learning rate* (LR). The LR is a common hyperparameter present in many ML algorithms, and it is intended to ease the navigation in the loss landscape of an ML algorithm. See subsubsection 2.2.4.1 for further details on hyperparameter tuning.

2.2.3 Supervised Learning without Labeled Data

A common dilemma in creating ML models for supervised learning is that labeled training data is not always available. To manually label an extensive dataset can take a significant amount of time in a project and may even be considered unfeasible in many situations, given the high human cost. Domain knowledge might also be required, which restricts the number of available annotators [12]. There are a few different methods that may be employed to alleviate some of the burden associated with hand labeling data.
One can apply *transfer learning*, which is a common technique where one can reuse previously trained ML models for related tasks [13]. Many similarities may exist in, e.g., the classification task of cars and buses. A bus classification model may be created from a previous car classification model by readjusting the model's parameters. This readjustment is done by training the reused car classifier on a limited set of labeled bus images. The requirements of a large labeled dataset can be drastically reduced given the pre-trained car model, and the time it takes to train the new bus model would naturally decrease as well.
One may attempt to apply *semi-supervised learning* [14], which is a form of incomplete supervision that deals with partly labeled datasets. The unlabeled data is

used to leverage the smaller subset of labeled data by modifying or reprioritizing assumptions derived from the labeled data.

Another approach to incomplete supervision is *active learning* [15]. The initial smaller subset of labeled data is used to create an initial ML model. Afterward, the model classifies unknown data, and if the classifier labels a data point with a relatively low probability, it queries some oracle. This oracle, commonly a human annotator, can help the model to label this critical data point since it is a data point that it struggles to classify. The labeled datapoint is added to the training set to recreate a more robust model. This approach requires one to label a small subset of the original dataset, followed by the outliers that the ML model may not have seen previously.

Finally, *weakly supervised learning* uses a set of potentially inexact and inaccurate weak supervision sources to label unlabeled data [12]. These weak supervision sources consist of labels of varying quality, which may be derived from:

1. *Distant supervision* - an external knowledge base is heuristically projected onto an unlabeled dataset [16].
2. Domain heuristics - e.g., utilizing the expected label distribution to determine the target labels.
3. Pattern matching - data is labeled by its common reoccurring sequences.
4. Other classifiers - predicted labels from other models determine the target labels.

While these noisy sources alone may provide labels of varying quality, their combination may be adequate to identify the labels for the unlabeled dataset. Methods that calculate target labels through heuristic rules will be discussed in greater detail in section 3.2.

2.2.4 Model Selection and Optimization

When building an ML model, one strives to fit a set of qualitative features to a set of target labels. A model needs to be complex enough to represent and store the provided features. An ML model is said to *underfit* if it cannot represent the underlying features of a dataset. Most ML algorithms can fit a dataset and create exceptionally complex models without significant difficulty since parameters to explain the data can often be added.

To fit a dataset tightly can come at a price, however. If a model is complex enough, it can fit the training set perfectly. Eventually, the model makes adjustments to fit every single data point, including the minor and more negligible features in the training set. When a model fits the individual data points rather than the general features of a dataset, we say that the model is *overfitted*. Thus, it cannot generalize and categorize unseen data since it has adapted too much to its training data.

A significant part of the optimization process of ML classifiers consists of finding the balance between a model which is complex enough yet general such that it can classify unseen data.

2.2.4.1 Hyperparameter Tuning

An ML algorithm's hyperparameters are parameters that are used to control the training process. Some common hyperparameters involve the number of training iterations and the *learning rate* (LR). The LR determines the magnitude of the parameter updates during training, where the model's parameters are updated to reduce the loss and make better predictions. A higher LR may cause the algorithm to fail to converge in the loss landscape (or even diverge). Meanwhile, a low LR may cause the algorithm to get stuck in a local minimum. Both scenarios are an issue, so careful hyperparameter tuning is often required when building an ML model.
A common technique to decide a model's hyperparameters is Grid Search Cross-Validation, which consists of Grid Search and Cross-Validation (CV). Grid Search is an exhaustive search through a set of defined hyperparameters. Cross-validation is the process of dividing a sample of data into complementary subsets, training an ML algorithm on one of the subsets, and validating the model on the other subset. This process is done repeatedly to see how the model can perform on unknown data, where the hyperparameters that result in the most significant generalization are selected. Hyperparameters are also in charge of *regularization* and *pruning*, which is discussed in subsubsection 2.2.4.2.

2.2.4.2 Regularization and Pruning

Regularization and pruning are two techniques that are commonly used to prevent overfitting. Regularization can be seen as a form of penalty that is often added to the loss function of an ML algorithm to prevent it from fitting its parameters too tightly to the training data. The regularization term is added to provide better long-term predictions for unknown data. Pruning is a method to drop noisy parameters that do not help the overall generalization of an ML model, which helps avoid overfitting. In subsection 2.2.1, the object function included the regularization constant C, which was used to decide the tolerated misclassifications when utilizing a soft margin.

$$\underset{\mathbf{w}}{\operatorname{argmin}} \quad \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{n=1}^{N} \xi_n$$

The significance of the ξ_n term is reduced as one steadily decreases the value of C. This adjustment results in a wider margin, where misclassifications may be tolerated. If C is relatively large, the margin becomes more narrow as misclassifications will be penalized by a greater factor. There is not some specific value of C that works well with any data. Instead, one has to know how the data is structured to know to what extent it can be separated. This insight determines which kind of margin is preferable.
XG-Boost may also utilize regularization techniques:

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta)\Omega(f) = \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2 + \gamma T$$

The first term is the *L2 regularization* term. The term consists of the tree's parameter weights and the hyperparameter λ. It is used to decrease the significance of individual data points in the leaves of the boosting trees to restrain the trees' depth. The second term is also another parameter meant to limit the depth of the boosting trees, and it does so through pruning. The factor T is the number of leaves in a tree, and the parameter γ is a user-defined hyperparameter. The product between the two values will be used as a threshold to determine whether a branch of a tree (a pair of terminal nodes) should be contracted to its predecessor leaf.

An increased depth of a tree often results in more accurate predictions, but the improvement may not be generalizable to unseen data. Thus, the added depth increases the risk of obtaining an overfitted model, which is biased to the data it has seen. It is preferable to consider regularization and pruning techniques in some settings to improve long-term predictions of the model.

Validation sets, briefly mentioned in subsubsection 2.2.4.1, can also be used to apply *early stopping*, which is a form of regularization. Early stopping is used to notify the ML algorithm to stop training due to an absence of a further reduced validation loss. The final model is not generalizing beyond the seen data if the validation loss is not further reduced, and any further improvements may be limited to the training data.

2.2.5 Evaluation of ML Models

One can evaluate the final performance of the model by its *accuracy*, *precision* and *recall* in classification tasks. These metrics can be described through the number of correct and incorrect predictions for both classes, commonly referred to as true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). TP and TN refer to the outcome where the model *correctly* predicts to positive and negative class respectively. FP and FN refer to the outcome where the model *incorrectly* predicts the positive and negative class, respectively. Accuracy, precision, and recall can thus easily be defined by the following equations:

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

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

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

Accuracy is a standard metric that measures how often the model makes a correct prediction. There are situations where it can give a skewed view of reality, e.g., when the class distribution of the dataset is imbalanced. Precision measures how often one's predictions were correct out of all predictions of a specific class. On the other hand, the recall measures how many of the true labels were classified correctly of a particular class. In other words, precision uses the predictions as its base, and recall uses the ground truth as its foundation.

Finally, there is also the F1-score metric, which is the harmonic mean between the

precision and recall, defined as:

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

All these metrics are often used, and one may choose a metric over another based on what task one faces. It is common to evaluate the performance of an ML model through a secluded *test set* by measuring how well it classifies the unknown data.
Significance tests may also be performed to evaluate if a model performs significantly better compared with another one. Significance tests often consist of a null hypothesis H_0, which is considered as the de-facto standard. The alternative hypothesis H_a is the negated version of H_0. An example of a null hypothesis H_0 could be:

H_0: All swans are black

where the alternative hypothesis would state:

H_a: All swans are not black

Thus, a test is conducted to test if H_0 can be rejected by some confidence interval (CI). A p-value of 0.05 allows one to reject H_0 with a CI of 95%, which is said to be an appropriate rate to use [17].
The Wilcoxon signed-rank test has been proposed as a fitting test for comparing ML models [17]. This significance test can be done by comparing the models' performances for a set of test sets and compare the two models' final metrics for each test set.

3

Literature Review

A literature survey of research in the financial markets and machine learning is discussed in the following section. The chapter is concluded with a review of the Stanford-based Snorkel project [18].

3.1 Stock Market Prediction Through News Events and NLP

Numerous articles on financial market forecasting were published in 2009, shortly after the United States housing bubble collapse. Kogan et al. predicted the volatility of stock returns in the American stock market from financial reports [19]. Support Vector Regression (SVR) was used in conjunction with bags of unigrams and bigrams, which resulted in an improvement over a baseline model that did not include textual information. Schumaker et al. studied stock prediction from financial news [20]. SVR was also used in this paper, and it was combined with text representation models such as bag of words (a statistical representation method of text where the unordered frequencies of unigrams are analyzed), as well as more complex representations such as noun phrases and named entities. The results showed that the model that used article terms and the stock price at the time of publication could predict the direction of a stock's price 57% of the time in the S&P 500 index. The model outperformed the variant that lacked input from news articles, highlighting that stock prediction through news events has been feasible in a stock market. Finally, Luss and D'Aspremont found that financial news can be used to predict abnormal stock returns [21].

In 2020, Zhou et al. studied seven stocks in the China A-share market over an 18 month period [22]. Multiple traditional and non-traditional news sources were combined to see how they affected the predictable power of stock price direction prediction. An SVM model was used, and the different news combinations resulted in a set of models with varying test accuracies. Four stocks were deemed active (frequently traded) and were grouped as a single group. The remaining three stocks were regarded as inactive (less frequently traded) and were grouped as a separate set of stocks. A majority of the models for both the inactive and active stocks resulted in test accuracies above 60%. However, the active stock prediction models tended to go well above this average, where the best-performing model reached an impressive 76% accuracy.

3.2 Snorkel

Snorkel is a Stanford-based project that began in 2016 and is best known for inventing the data programming framework and the Snorkel library[1] that implements and leverages this framework [18]. Data programming is a method to build training data sets programmatically [4]. The data programming paradigm employs weak supervision to programmatically label data points using so-called labeling functions (LF). An LF $l_i(x)$ is a user-defined program that assesses whether or not a data point x_j belongs to a certain category c based on some criterion.

$$\mathrm{l_i}(x_j) = \tilde{y_j} = \begin{cases} c & h(x_j) = True \\ abstain & \text{otherwise} \end{cases}$$

where the target label $\tilde{y_j}$ is labeled either as any of the categories from $c \in C$ or as *abstain*. Abstain is the alternative when conclusions cannot be drawn from the datapoints characteristics. LFs can provide a fast, dynamic and cheap approach to data labeling. Labels may be derived from distant supervision, domain heuristics, pattern matching, or even by other classifiers. To illustrate, two hypothetical LFs are shown in which the goal is to decide if a YouTube comment should be classified as `SPAM` or not spam (denoted as `HAM`)[2]:

```
@labeling_function()
def regex_check_out(x):
    if re.search(r"check.*out", x.text, flags=re.I):
        return SPAM
    else:
        return ABSTAIN


@labeling_function(pre=[spacy])
def has_person(x):
    if len(x.doc) < 20 and
                        any([ent.label_ == "PERSON"
                        for ent in x.doc.ents]):
        return HAM
    else:
        return ABSTAIN
```

The LF `regex_check_out` uses *regular expressions* (RegEx) to match the string `'check'`, eventually followed by `'out'`. This assumption can be drawn from domain knowledge of how `SPAM` messages tend to look like or from a small labeled development set that can be used for insight. The development set may show that `HAM` comments are often short and mentions a person in the video. We thus have

[1] https://www.snorkel.org/

[2] https://github.com/snorkel-team/snorkel-tutorials/blob/master/spam/01_spam_tutorial.ipynb

an LF `has_person`, which checks the length of the comment and utilizes a part-of-speech tagging library to determine if the comment mentions a person. Granted, it will label the comment as `HAM`.
A mentioned drawback of LFs is that they are noisy. Further, they will vary in *coverage* (the fraction of data points that were labeled as non-abstain), they will *overlap*, and thus also introducing the possibility of *conflicts*. Two LFs may categorize the same data point as two unique categories.
One may construct a labeling model by letting the LFs vote individually for each data point, with a majority vote determining the final label. However, an unweighted majority vote ignores the latent accuracy of each LF. The data programming framework, on the other hand, can take advantage of these conflicts and use them to estimate the relevance of each LF. Snorkel estimates the target labels by solving a set of optimization problems for a set of statistical parameters through optimization methods such as stochastic gradient descent [18]. The Snorkel algorithm will thus produce a generative model consisting of all the LFs. The algorithm will train the final model using the training set and the LFs in order to estimate the unknown latent accuracies of each LF. Data programming assumes that each LF makes independent labeling decisions. However, Snorkel can estimate the correlations between LFs such that their vote can be given a reasonable weight of independence. Highly correlated LFs will thus not be weighted twofold, and LFs with a low coverage will not be excluded.
The generative model produces a set of probabilistic, noise-aware training labels that may further train an end model to improve generalization. An LF might match a set of words through RegEx but miss a set of synonyms. The matched words are assigned a set of weights that connects the words to their corresponding label. Deep learning can further be utilized, where the synonyms of the matched strings can be recognized through, e.g., words embeddings [23] [24]. These related word synonyms are thus also recognized as belonging to the particular class, resulting in a final model that can generalize beyond the LFs and increase the model's recall.

4
Method

This chapter introduces the approaches and design choices to create the Press Release Labeler and Stock Price Predictors. The datasets used are also presented. The section is concluded with the main tools utilized in this thesis.

4.1 Data Collection

Press releases from companies traded at the Nordic stock markets were studied in this thesis. Please see Appendix B for further information on the included stock markets. The available companies over the studied period can be seen in Figure 4.1.

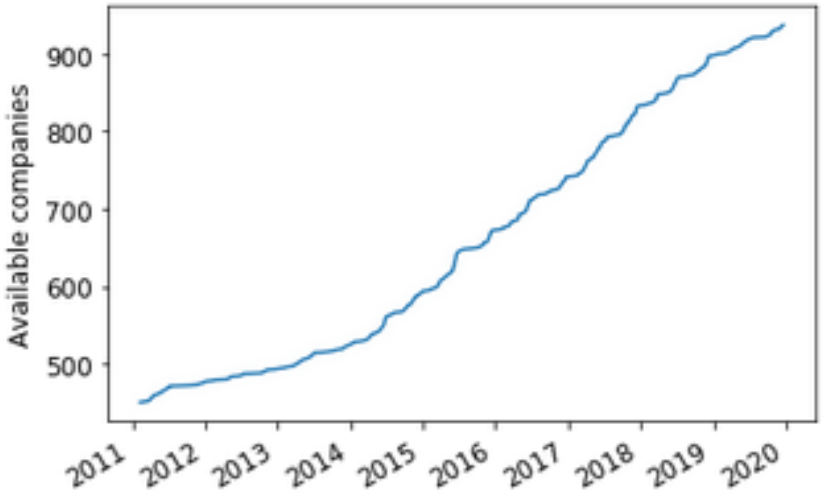

Figure 4.1: The number of available companies over time. Data from roughly 500 companies were available in 2011, increasing as more companies have entered the stock market.

4.1.1 Press Releases

The dataset of press releases consisted of 219, 389 entries and was provided by *MFN*, which provides a service that allows companies to communicate to the public through press releases. The press releases were issued between February 10, 2011, until February 10, 2020, where the following features describe each entry:

- `isins` - A list of the company's International Securities Identification Number.
- `name` - The name of the company.

- `publishing_date` - A timestamp of when the press release was communicated.
- `title` - The title of the press release.
- `content` - The text from the press release body.

Relevant metadata regarding the press releases can be seen in Figure 4.2, Figure 4.3, and Figure 4.4.

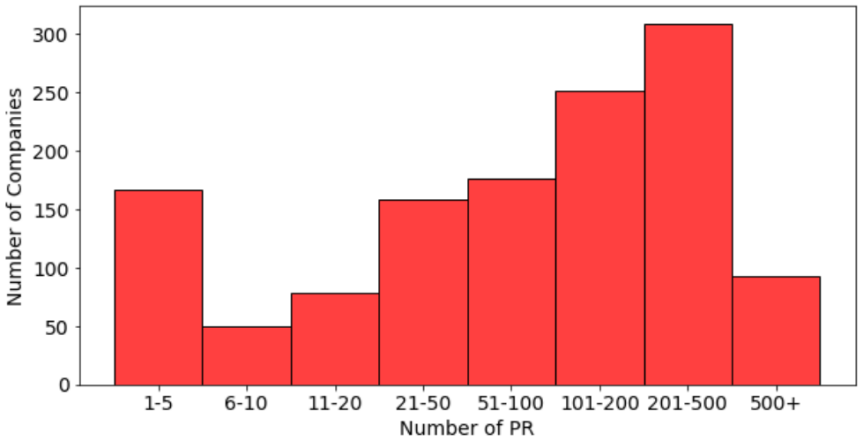

Figure 4.2: The number of companies over the number of available press releases (PR). Roughly 300 companies have issued between 201-500 press releases, and about 150 companies have issued less than six press releases.

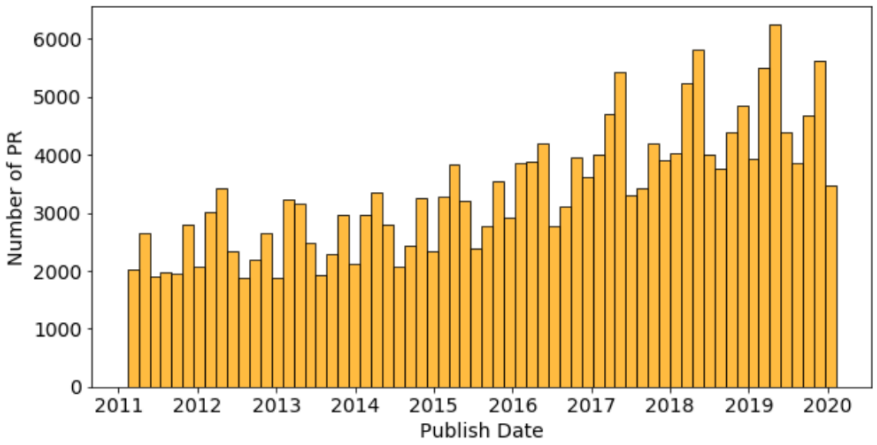

Figure 4.3: The distribution of the press releases (PR) over time.

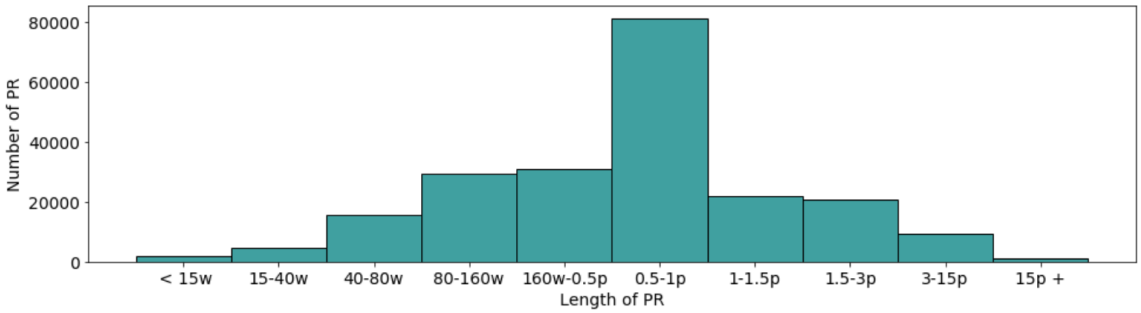

Figure 4.4: The number of press releases (PR) per size group, where the size group is based on the press releases' content length in words (w) and pages (p).

4.1.2 Market Data

The market data was retrieved through an API (Application Programming Interface) of Börsdata, a company that delivers financial information from various stock markets. The data was accessible through the API using standard HTTP methods such as GET and POST. The companies' historical stock prices and related stock market indexes were used. The data is described by the following features, where each feature considers the stocks' or indexes' price of the day it was traded:

- `o` - The opening price.
- `h` - The highest paid price.
- `l` - The lowest paid price.
- `c` - The closing price.
- `v` - The volume in shares traded.
- `d` - The date.

Metadata regarding each company, such as their main country of operations, sector, line of business, and stock exchange, were also retrieved. See Figure 4.5 and Figure 4.6.

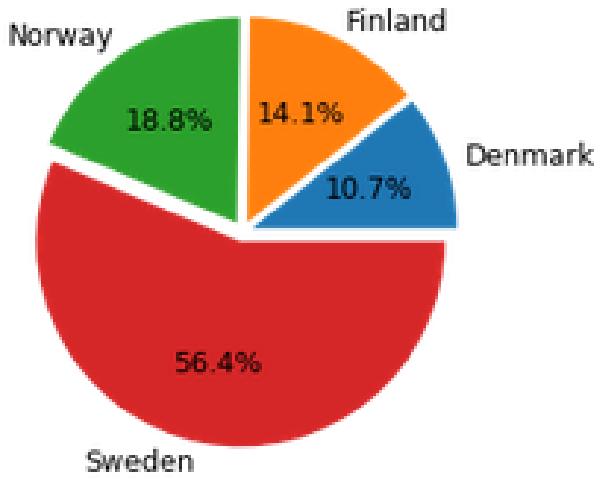

Figure 4.5: The distribution of companies per country.

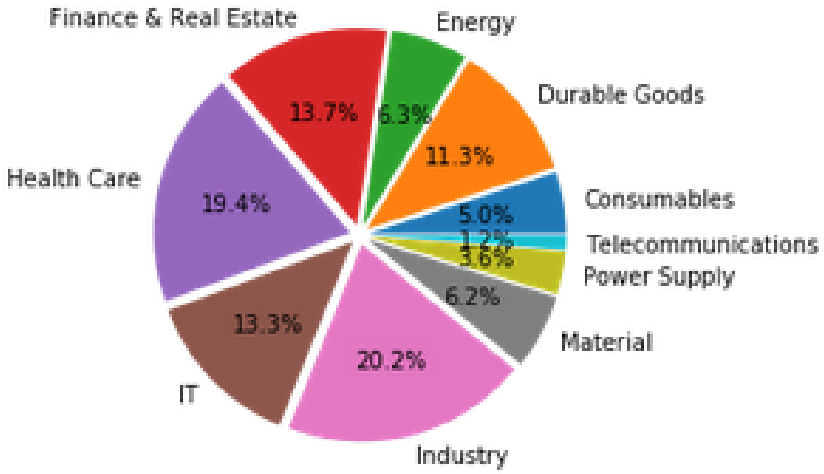

Figure 4.6: The distribution of companies per sector.

The distribution of the lines of businesses and the studied stock exchanges can be seen in Appendix B.

4.2 Press Release Labeler

The Press Release Labeler was built with the Python library *Snorkel*. The Snorkel framework is discussed in greater detail in section 3.2. The press releases were divided into ten unique categories and an unknown category, which can be studied in Table 4.1. The categories were chosen with the ambition to be easily interpreted by human readers.

Table 4.1: The press release categories that were labeled through weak supervision. Only one category can be assigned for each press release.

Category Id	Category Name	Category Acronym
0	share_buyback	SB
1	new_deal	ND
2	acquisition	AC
3	earnings_report	ER
4	new_key_person	NK
5	lost_key_person	LK
6	insider_buy	IB
7	insider_sell	IS
8	award	AW
9	divestment	DI
10	unknown	UN

4.2.1 Model Selection

The generative model was created by a set of 38 unique labeling functions (LF), where text from the press release's `title` and `content` was considered.
Most LFs were written in a similar fashion, where phrases that indicated a high probability of a particular class were caught, followed by a set of filters to remove common false positives. An LF is presented for the category `acquisition` below:

```
def acquisition_lf1_func(x, label_lookup):
    regex_str = r"acqui[^\.]*(company\b|all[^\.]*share)"
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_ACQUISITION = anti_acquisition_words(x)
        if NOT_ACQUISITION:
            return ABSTAIN

        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(
        x, 3, regex_str)

        if EXIST_IN_THREE_FIRST_SENTENCES == False:
            return ABSTAIN

        return label_lookup[ACQUISITION]
    else:
        return ABSTAIN
```

The LF `acquisition_lf1_func`, which can be seen above, uses RegEx to perform

pattern matching by looking for a sentence containing a prefix `'acqui'`, eventually followed by the word `'company'`, or `'all'` followed by `'share'` in the press release body.

Suppose a press release includes this string sequence. In that case, it will also require the following: (i) it does not contain common earning report words in the title since interim and annual reports often mention previous acquisitions; (ii) the press release include relevant keywords such as `'million'` or `'billion'` since the absence of the amount spent would highly indicate that the press release would not be of the category `acquisition`; (iii) the acquisition is mentioned in any of the three first sentences.

There are further a few additional conditions, and given that the press release passes all the requirements, it will be labeled by this LF as an `acquisition` press release. The remaining LFs can be studied in detail in Appendix A.

Finally, the LFs were structured as a label matrix and trained through Snorkel's data programming framework. The produced probabilistic target labels from the final generative model were turned into hard labels as no discriminative end-model was utilized.

4.2.2 Optimization and Evaluation

The LFs were written together with a small labeled development set of 326 press releases, where a set of press releases was manually labeled and used to gain insight for different press release categories. However, 204 labels were of the category `unknown` and were therefore not used. New categories were also successively added throughout the process during the project. Precision was prioritized to ensure that the categories represented the intended press release.

An observation from the development set was that companies tend to append previous summaries of accomplishments to the end of their press releases. The assumption was made that relevant information will often appear at the beginning of a press release.

The hyperparameters for creating the generative model were selected through trial and error, where a tie break policy of abstain was applied and where the model trained for 5000 epochs. A small labeled test set consisting of 293 press releases was also used to evaluate the performance of the generative model, where 175 of these press releases were of the category `unknown`.

4.3 Stock Price Predictors

Two different Stock Price Predictors were created to predict the price movement direction and large price movements of a stock given a press release. We define $p_{c,j,t}$ and $p_{o,j,t}$ to be the closing and opening price for company j at time t respectively. The target label for the price movement direction task is assigned as 1 given that the price had increased during the day. Conversely, if the price did not change or decreased, it is assigned the value of 0. We define the target label of the price

movement direction more formally as:

$$y_{j,t} = \begin{cases} 1 & \frac{p_{c,j,t}}{p_{o,j,t}} > 1 \\ 0 & \frac{p_{c,j,t}}{p_{o,j,t}} \leq 1 \end{cases}$$

A large price movement is labeled as 1 given that the price change between the opening and closing price is greater than 3%. Otherwise, it is considered a small price change, and it will be labeled as 0. Thus, we define the target label of a large price movement more formally as:

$$y_{j,t} = \begin{cases} 1 & |\frac{p_{c,j,t} - p_{o,j,t}}{p_{o,j,t}}| \times 100 > 3 \\ 0 & |\frac{p_{c,j,t} - p_{o,j,t}}{p_{o,j,t}}| \times 100 \leq 3 \end{cases}$$

where we in these formalized definitions assume that the press release was issued in the morning before opening hours.

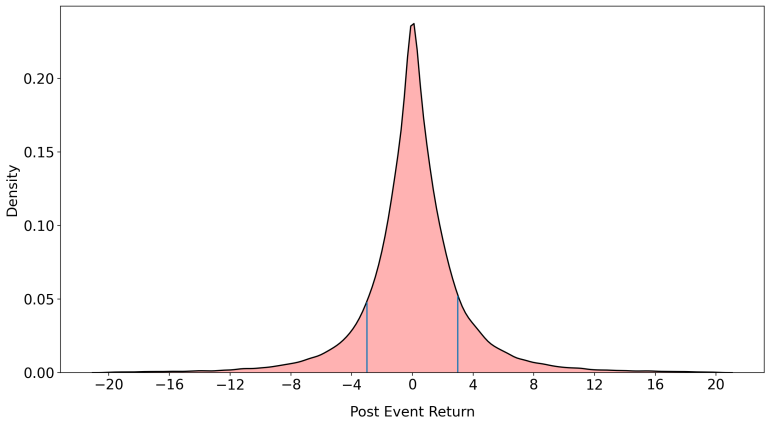

Figure 4.7: The distribution of returns given by stocks provided a press release.

No hinge or threshold was seen in the density plot in Figure 4.7 such that the choice to define extraordinary return was clear. Thus, a |3%| threshold was selected as the threshold between large and small returns, where returns above |3%| represent roughly 25% of all stock returns given a press release.

4.3.1 Model Selection

Four sets of XG-Boost Classifiers and SVM classifiers were selected and trained as Stock Price Predictors, where two sets of classifiers were used for stock price direction prediction, and two were used for large price movement prediction. One of the algorithms was trained with the given set of known press release categories and market data, and the other algorithm was trained only through market data.

The XG-Boost classifier was selected since it has shown to be a strong candidate in a wide range of classification competitions [10]. The SVM classifier was selected as it has shown to be a prosperous approach in previous stock market prediction research [25]. The selected hyperparameters of the XG-Boost and SVM models can be seen in Table 4.2 and Table 4.3.

Table 4.2: The XG-Boost models' hyperparameters for the Stock Price Direction Predictor (pd) and Large Price Movement Predictor (lm).

		$XGB_{pd,\Phi}$	$XGB_{lm,\Phi}$	$XGB_{pd,\varnothing}$	$XGB_{lm,\varnothing}$
Hyperparameter	objective	binary:logistic	binary:logistic	binary:logistic	binary:logistic
	reg_lambda	0.65	0.60	0.65	0.60
	gamma	0.15	0.10	0.20	0.10
	max_depth	7	7	5	6
	learning_rate	0.03	0.01	0.03	0.01
	scale_pos_weight	0.925	3.0	0.925	3.0
	subsample	0.9	0.9	0.9	0.9
	colsample_bytree	0.5	0.5	0.5	0.5

Table 4.3: The SVM models' hyperparameters for the Stock Price Direction Predictor (pd) and Large Price Movement Predictor (lm).

		$SVM_{pd,\Phi}$	$SVM_{lm,\Phi}$	$SVM_{pd,\varnothing}$	$SVM_{lm,\varnothing}$
Hyperparameter	kernel	linear	linear	linear	linear
	C	0.005	30	0.02	2
	dual	False	False	False	False
	penalty	$l2$	$l2$	$l2$	$l2$
	loss	squared_hinge	squared_hinge	squared_hinge	squared_hinge
	tol	0.0005	0.001	0.0001	0.001
	class_weight	{0: 1, 1: 1}	{0: 1, 1: 3}	{0: 1, 1: 1}	{0: 1, 1: 3}
	max_iter	100	100	100	100
	intercept_scaling	0.5	1.0	1.5	2.5

Some feature engineering was done in order to provide a set of well-defined features for the training process. Features that were used included a set of categorical features, such as the company's country, sector, line of business, and exchange, as well as the press releases category, labeled by the Press Release Labeler. Market data prior to the press release regarding the price and volume of the stock and a benchmark index were also included.

The benchmark index for each country was selected as OMX Stockholm PI for Sweden, Oslo Exchange All Share for Norway, OMX Helsinki PI for Finland, and OMX Copenhagen PI for Denmark.

The input features can be studied in table Table 4.4.

Table 4.4: The input features of the XG-Boost and SVM stock price predictor models. Stock Returns of N Days is the return that the stock has given during the N previous trading days before the press release. The Stock Press Release Reaction feature is $p_{o,j,t}/p_{c,j,t-1}$, assuming that the press release was issued in the morning before opening hours. The volatility features are measured over the previous 32 days, where the standard deviation measures volatility. R^2 measures how well the stock that issued the press release is following the benchmark index.

Feature Id(s)	Variable Type	Input feature
14-937	Categorical	Stock Id
1098-1108	Categorical	Press Release Class
1093-1097	Categorical	Stock Country Id
1082-1092	Categorical	Stock Sector Id
987-1081	Categorical	Line of Business Id
938-986	Categorical	Stock Market Id
1	Float	Stock Price Close Volatility
12	Float	Stock Price Low-High Volatility
0	Float	Stock Volume Volatility
4	Float	Stock Press Release Reaction
5	Float	Stock Returns of 8 Days
6	Float	Stock Returns of 16 Days
7	Float	Stock Returns of 32 Days
2	Float	Benchmark Index Close Volatility
13	Float	Benchmark Index Low-High Volatility
8	Float	Benchmark Index Press Release Reaction
9	Float	Benchmark Index Returns of 8 Days
10	Float	Benchmark Index Returns of 16 Days
11	Float	Benchmark Index Returns of 32 Days
3	Float	R^2

4.3.2 Data Preprocessing

The Stock Price Predictors used a range of different features, which were studied in Table 4.4. All the categorical features were one-hot encoded since some ML algorithms would have interpreted the features as continuous values if their initial numerical representation was kept. The country identifier is 1 for Sweden, 2 for Norway, 3 for Finland, and 4 for Denmark, but grouping some countries together based on their numerical value would not make any sense.
The volatility sequences were normalized through Min-Max Scaling, defined as:

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

where $z = (z_1, \ldots, z_n)$ is the normalized data of $x = (x_1, \ldots, x_n)$. Numerical data is commonly normalized if one does not care about the magnitude but rather how small or large the value is in proportion to the rest of the dataset.

Press releases issued during the market opening hours were not included in the final stock price predictor models due to a lack of intraday prices. Press releases belonging to companies that had missing data were also dropped from the dataset. Finally, stocks that were rarely trading were also discarded.

4.3.3 Optimization and Evaluation

All the models were optimized through hyperparameter tuning. The XG-Boost and SVM model used a five-fold Grid Search CV to find the optimal values of each models' hyperparameters.
The Stock Price Direction Predictor model was optimized with accuracy in mind. This choice felt reasonable, as the dataset contained a balanced target label distribution, and since both true positives (TP) and true negatives (TN) were of interest. One can go *long*, i.e., buy a stock with the intention of selling it at a higher price, or go *short*, i.e., sell a stock with the intention of buying it back at a lower price. The number of correct predictions in both price directions was thus of interest.
The Large Price Movement Predictor was optimized with F1-score for large price movements in mind since there is little interest in predicting the absence of a large price movement.
A test set was used to evaluate the performance of the end model after training and optimization. A scikit-learn Dummy classifier with a stratified strategy was used as a baseline to compare the predictability of the final models. The Wilcoxon signed-rank test was later used to compare the models' performance.

4.4 Tools

The tools used to investigate the studied field in this report will be described in more detail in the following subsections.

4.4.1 Python

All programs in this thesis have been implemented through *Python*[1], an interpreted high-level, general-purpose programming language. Python was selected due to its strong support in the data science and ML community, where library packages such as *NumPy* and *pandas* are easily accessible and significantly simplify any work where linear algebra or data tables are used. Additional packages, such as Snorkel and a wide range of other ML and NLP packages (discussed in the following sub-sections) was another important reason for choosing Python.

4.4.2 Scikit-learn and XG-Boost

Scikit-learn[2] is an open-source software library for machine learning. It was used in this project due to its simplicity and efficiency in designing ML models [26]. The

[1]https://www.python.org/
[2]https://scikit-learn.org/

SVM model and the Dummy Classifier baseline were designed through Scikit-learn, where algorithms could be instantiated, trained, and tuned in a few days without significant demands.
The XG-Boost library[3] is an open-source software library used to create the XG-Boost model. The library can be used together with Scikit-learn's optimization methods to tune the XG-Boost model's hyperparameters.

4.4.3 Snorkel and NLP Libraries

The Snorkel library[4] is a software library that can be used to implement the data programming framework, which can be studied in more detail in section 3.2. The Snorkel library integrates well with NLP libraries and ML libraries [18]. When creating the LFs, the Python native *re* package was used to make RegEx pattern matching on the title and content of the press releases. Further, some of the LFs were designed using the *spaCy* software library[5], which is an open-source software library for part-of-speech tagging.

[3]https://xgboost.readthedocs.io/en/latest/
[4]https://www.snorkel.org/
[5]https://spacy.io/

5

Results

The results from the Press Release Labeler and the impact that the different types of news events have on stock returns are presented in the following sections. Finally, the result of the Stock Price Predictors is presented.

5.1 Press Release Labeler

The fraction and number of labeled press releases from the Press Release Labeler can be seen in Table 5.1.

Table 5.1: The number of labeled press releases per category. 52293 out of 218757 press releases were actively labeled as a category, where the remaining 76% were `unknown` (UN).

Category Acronym	Fraction	n
SB	0.077	16747
ND	0.016	3518
AC	0.017	3783
ER	0.071	15597
NK	0.024	5166
LK	0.005	1202
IB	0.018	3957
IS	0.004	863
AW	0.002	543
DI	0.004	917
UN	0.761	166464

A table showing the label matrix can be seen in Appendix A. The performance metrics of the Press Release Labeler can be seen in Table 5.2. Please revisit subsection 2.2.5 for a detailed explanation of the metrics used in the table.

Table 5.2: A set of performance metrics for the Press Release Labeler.

Category Acronym	Precision	Recall	F1-score	n
SB	1.00	0.79	0.88	19
ND	0.80	0.16	0.27	25
AC	0.71	0.56	0.63	9
ER	0.92	0.63	0.75	38
NK	0.83	0.83	0.83	6
LK	1.00	0.50	0.67	2
IB	0.80	0.40	0.53	10
IS	0.00	0.00	0.00	2
AW	0.00	0.00	0.00	4
DI	1.00	1.00	1.00	3
UN	0.75	0.97	0.84	175

The lower recall in Table 5.2 highlights that far from all of the labeled press releases were caught by the LF's, where the large majority instead were classified as `unknown` (UN). An example would be the low recall of `new_deal` (ND), emphasizing that further LFs may be needed to cover this category. However, the relatively high precision indicates that the press releases that were labeled were done so quite accurately. The `inside_sell` (IS) and `award` (AW) categories received a precision and recall of 0, possibly because there only exists a support of two and four test labels, respectively, for the categories.

5.2 Press Release Impact Analysis

The impact of press releases on stock returns of specific categories will be presented in this section. Only press releases issued after or before opening hours are selected. The average return during the first 20 days from the set of press release categories can be seen in Figure 5.1.

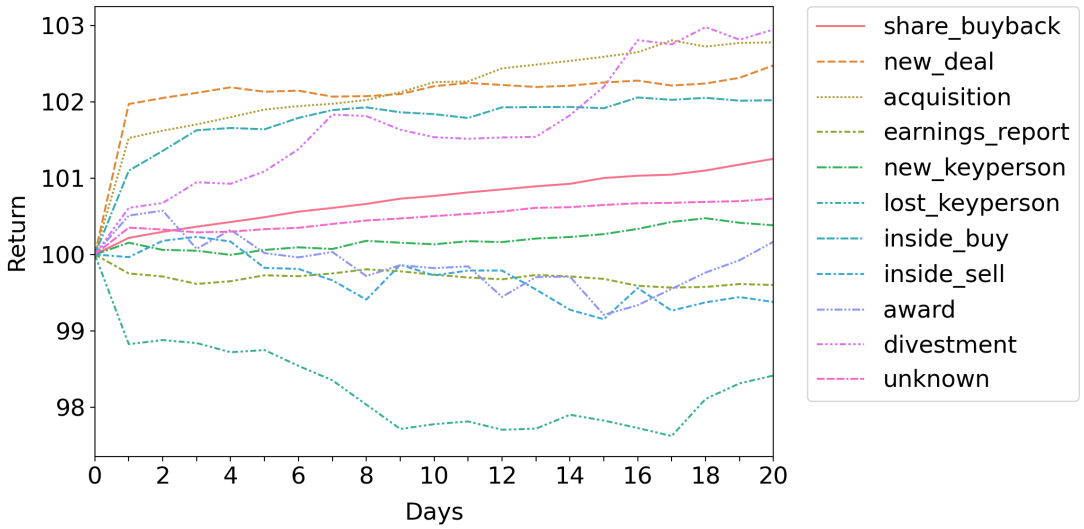

Figure 5.1: The average returns per day over 20 days before issued press releases of some certain category.

The stock price movement is significantly greater in magnitude for most categories during the first day of the event compared to the rest of the consecutive days. The average return of the press release categories during the first day of the event can be studied in more detail in Figure 5.2 and Figure 5.3.

A pre-event trade and post-event trade is investigated. Both the pre-event and post-event trades are sold at the closing price after one day of trading. However, the pre-event trade is initiated at the price prior to the press release, i.e., the recorded closing price of the trading day before the event. On the other hand, the post-event trade is initiated at the recorded opening price of the trading day after the announced press release.

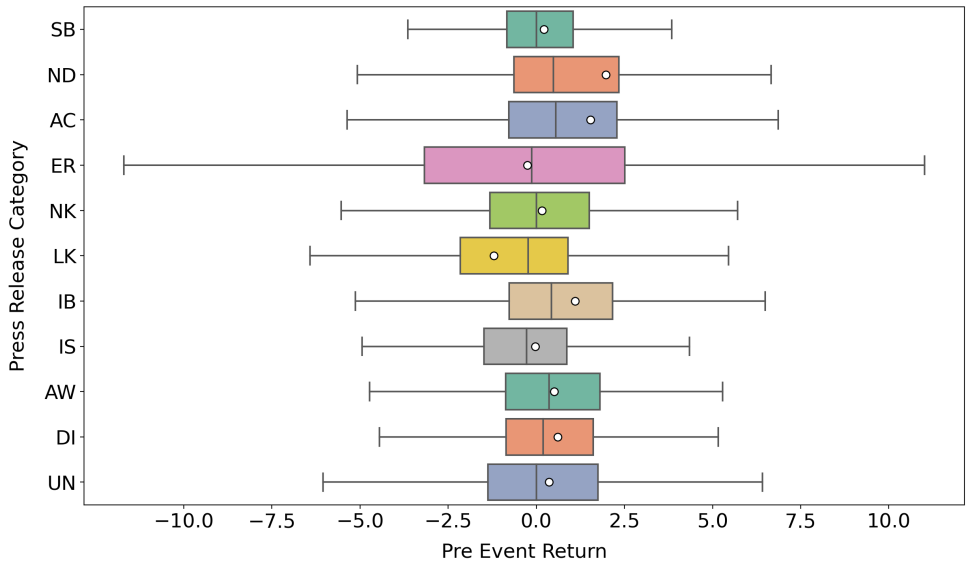

Figure 5.2: The pre-event trade for the set of different press release categories. The white dot represents the category's average stock return.

The pre-event trade in Figure 5.2 shows that the `new_deal` (ND), `acquisition` (AC), and the `inside_buy` (IB) press releases result in the most positive average returns. The `lost_key_person` (LK) results in the most negative returns on average. However, their median appears to stand out less to the remaining press release categories.

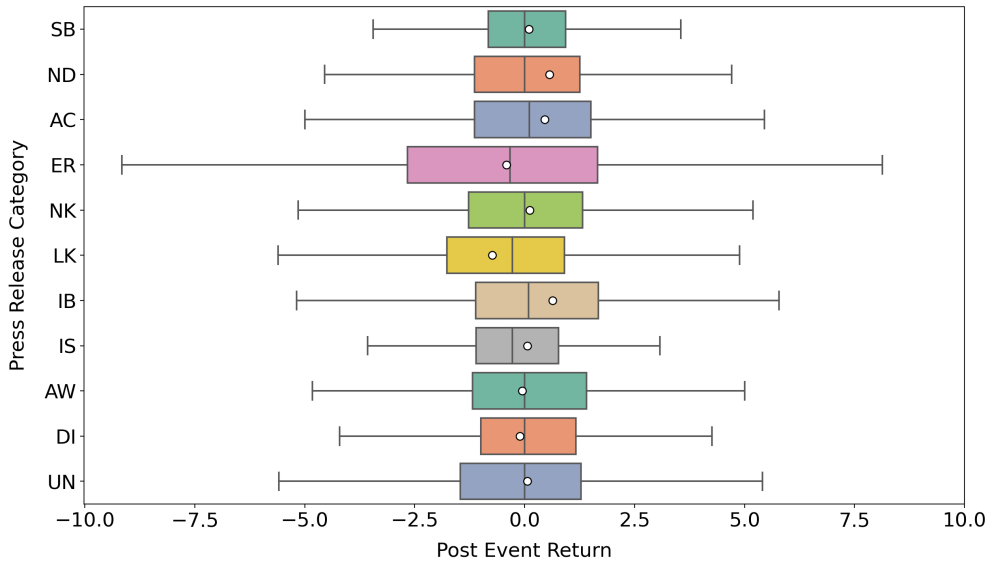

Figure 5.3: The post-event trade for the set of different press release categories. The white dot represents the category's average stock return.

The post-event trade in Figure 5.3 shows a less distinguishable result, but which nevertheless seem to follow a similar pattern with respect to the categories' pre-event returns.

The pre-event and post-event trade can be studied in greater detail in Table 5.3 and Table 5.4, where $\bar{x}$ is the average stock return, α is the average stock return minus the average index return, σ is the standard deviation of the stock returns, and R^2 measures how well the stock that issued the press release is following the benchmark index.

Again, one can see that the pre-event trade results in a larger positive average return for the `new_deal` (ND), `acquisition` (AC), and `inside_buy` (IB) press releases. The large α shows that the returns are significant with respect to their benchmark index. However, the volatility of the returns can be noted when observing the relatively high σ compared to, e.g., the less affected `share_buyback` (SB) category. A similar pattern can be seen for the `lost_key_person` (LK) press releases, which results in the largest negative returns on average.

Table 5.3: Financial metrics of press releases of different categories for a pre-event trade.

Category Acronym	$\bar{x}$	α	σ	R^2	n
SB	0.22	0.17	2.29	0.15	6422
ND	1.97	1.93	8.57	0.01	1611
AC	1.54	1.49	7.85	0.01	1605
ER	-0.25	-0.31	6.41	0.03	10242
NK	0.16	0.11	4.27	0.04	2471
LK	-1.21	-1.29	6.81	0.02	572
IB	1.10	1.12	5.24	0.05	1249
IS	-0.03	-0.08	4.46	0.06	203
AW	0.51	0.48	7.96	0.00	253
DI	0.61	0.58	4.83	0.06	411
UN	0.36	0.31	6.55	0.02	70904

In Table 5.4, one can see the result of the post-event trades. One can observe that the `share_buyback` (DI), `new_key_person` (NK), `inside_sell` (IS), `award` (AW), and `divestment` (DI) are all categories that on average does not result in any vast positive or negative returns. One can observe that companies that issue `earnings_report` (ER) press releases tend to drop further in price during the trading day. The average return in the post-event trade is -0.41% compared to the less extensive -0.25% in the pre-event trade. The remaining four categories previously discussed that resulted in the highest returns in the pre-event trade seem to also result in the highest returns in the post-event trade.

Table 5.4: Financial metrics of press releases of different categories for a post-event trade.

Category Acronym	$\bar{x}$	α	σ	R^2	n
SB	0.09	0.08	1.99	0.10	6422
ND	0.57	0.54	6.64	0.01	1611
AC	0.45	0.44	4.64	0.02	1605
ER	-0.41	-0.45	4.79	0.03	10242
NK	0.11	0.09	3.48	0.04	2471
LK	-0.74	-0.79	5.30	0.03	572
IB	0.63	0.66	4.25	0.06	1249
IS	0.06	-0.01	3.95	0.03	203
AW	-0.05	-0.01	5.04	0.00	253
DI	-0.11	-0.09	3.53	0.06	411
UN	0.07	0.04	4.95	0.02	70904

5.3 Stock Price Predictors

The results from the Stock Price Direction Predictor models and the Large Price Movement Predictor models are presented. The symbols Φ and $\varnothing$ are used as subscripts in the classifiers, where Φ indicates that press release categories are available, and where $\varnothing$ indicates that they are not. The five models for each classification task were the following: the two XG-Boost classifiers with (XGB_Φ) and without ($XGB_\varnothing$) access to the press release categories; the SVM classifiers with (SVM_Φ) and without ($SVM_\varnothing$) access to the press release categories; as well as the Dummy classifier with strategy stratified (D_s).

Table 5.5: The evaluation metrics for the Stock Price Predictor models for price direction movement. XGB_Φ reaches the accuracy for predicting stock price direction out of all the classifiers.

Model	Accuracy	P_{down}	R_{down}	$F1_{down}$	P_{up}	R_{up}	$F1_{up}$
SVM_Φ	0.533	0.536	0.370	0.438	0.532	0.691	0.601
$SVM_\varnothing$	0.528	0.528	0.386	0.446	0.529	0.666	0.590
XGB_Φ	0.544	0.538	0.518	0.528	0.550	0.569	0.560
$XGB_\varnothing$	0.542	0.536	0.515	0.525	0.548	0.569	0.558
D_s	0.497	0.488	0.466	0.477	0.506	0.528	0.516

The metrics for the Stock Price Direction Predictor models can be seen in Table 5.5. The XG-Boost models seem to perform the best out of all the models with respect to accuracy. The XGB_Φ model seems to slightly outperform the $XGB_\varnothing$ model.

Table 5.6: The evaluation metrics for the Stock Price Predictor models of large price movements. XGB_{Φ} reaches the highest F1-score for predicting large stock price movements out of all the classifiers.

		Accuracy	P_{small}	R_{small}	$F1_{small}$	P_{large}	R_{large}	$F1_{large}$
Model	SVM_{Φ}	0.618	0.836	0.577	0.683	0.406	0.719	0.519
	$SVM_{\varnothing}$	0.617	0.834	0.577	0.682	0.405	0.715	0.517
	XGB_{Φ}	0.643	0.841	0.615	0.710	0.427	0.712	0.534
	$XGB_{\varnothing}$	0.645	0.838	0.623	0.714	0.428	0.701	0.532
	D_s	0.603	0.715	0.737	0.726	0.294	0.271	0.282

The metrics for the Large Price Movement Predictor models can be seen in Table 5.6. Again, the XG-Boost models seem to perform the best in terms of F1-score ($F1_{large}$), with the XGB_{Φ} model marginally outperforming the $XGB_{\varnothing}$ model.
A significance test was later conducted between the XGB_{Φ} and the D_s model for both classification tasks to see if stock price movements are predictable. See Table 5.7 and Table 5.8. The null hypothesis H_0 was stated as:

H_0: Stock price movements in the short term are entirely random.

The performed Wilcoxon signed-rank test from the results that can be seen in table Table 5.7 and Table 5.8 resulted in a p-value of $1.55e{-}04$ and $1.91e{-}06$, respectively. The two p-values rejects the null hypothesis for both prediction tasks with a 99% confidence interval.

Table 5.7: The accuracy (denoted as A) of the $XGB_{pd,\Phi}$ and $D_{pd,s}$ model for stock price direction prediction.

Test	A_{XGB_Φ}	A_{D_s}	n
1	0.539	0.490	953
2	0.554	0.511	953
3	0.532	0.501	953
4	0.531	0.497	952
5	0.533	0.491	952
6	0.543	0.543	952
7	0.563	0.493	952
8	0.551	0.521	952
9	0.580	0.513	952
10	0.570	0.487	952
11	0.535	0.508	952
12	0.552	0.475	952
13	0.511	0.495	952
14	0.566	0.489	952
15	0.516	0.529	952
16	0.543	0.481	952
17	0.529	0.510	952
18	0.554	0.496	952
19	0.561	0.543	952
20	0.522	0.479	952

Table 5.8: The F1-score for the $XGB_{lm,\Phi}$ and $D_{lm,s}$ model for large stock price prediction.

Test	$F1_{XGB_\Phi}$	$F1_{D_s}$	n
1	0.564	0.327	953
2	0.480	0.259	953
3	0.463	0.208	953
4	0.540	0.237	952
5	0.602	0.306	952
6	0.487	0.285	952
7	0.511	0.232	952
8	0.573	0.262	952
9	0.523	0.277	952
10	0.500	0.256	952
11	0.563	0.266	952
12	0.562	0.288	952
13	0.495	0.262	952
14	0.535	0.314	952
15	0.511	0.254	952
16	0.535	0.194	952
17	0.540	0.243	952
18	0.568	0.285	952
19	0.510	0.256	952
20	0.542	0.273	952

6
Discussion

The results is further analyzed and discussed in the following sections.

6.1 Press Release Labeler

The Press Release Labeler managed to label 24% of all the press releases with a relatively low effort, highlighting the efficiency of the data programming framework. The precision for the categories with a sufficient number of test labels showed promising results, emphasizing the quality that weak supervision can provide. Time was not much of a limiting factor when it came to covering the press releases. Instead, the limiting factor was domain expertise of economics and the financial markets. The deliberate choice was thus made to keep the categories high level and few, such that one could investigate the fewer number of selected categories in greater detail.
An observation that can be made from Table 5.1 is the differences in frequency between the `new_key_person` (NK) and `lost_key_person` (LK) category. The number of labeled press releases for each of these categories is presented in Figure 6.1.

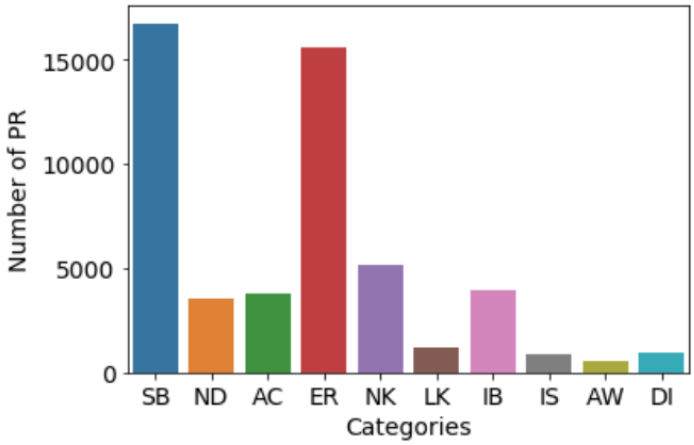

Figure 6.1: The number of labeled press releases (PR) for the ten defined press release categories.

One could expect that the ratio between these categories should be relatively similar. However, less encouraging news may be presented in more discrete ways, avoiding the common keywords upon which most LFs were based.

6.2 Press Release Impact Analysis

News events in the Nordic Stock Markets seem to mainly impact the stock returns in the short term, which can be seen in Figure 5.1. The `divestment` (DI) category seems to be an exception, as, over time, it seems to be the category that performs the strongest out of all categories during the 20-day period. A divestment might be a part of some new company strategy that the stock market might reward.

The `lost_key_person` (LK) category is the category that appears to lag behind the most. In reverse to the divestment theory, this category may reflect some fundamental problems in the company. A lack of confidence in the stock may be reflected in the stock price.

The returns of the pre-event trade presented in Figure 5.2 showed relatively large returns for many of the categories, where the `new_deal` (ND) category resulted in almost a two percent return on average per trade. The post-event trade presented in Figure 5.3 showed returns that were smaller in magnitude but still large enough to be considered. The `inside_buy` (IB) and `lost_key_person` (LK) categories resulted in a 0.63% and -0.74% return on average per trade. Thus it seems that the market may not adjust prices directly to a fair price given a news event of some particular category. These results suggest that one may utilize the press release category to predict what direction and to what extent a stock's price will move.

The Efficient Market Hypothesis (EMH) and its critics are carefully analyzed in 2003 by B. Malkiel [27], where multiple critics and their points of view are considered. It has been said that stocks can have *momentum*. Many internet companies during the *Dot-com bubble* were heavily traded, and the prices of these stocks gained rapidly over a short period of time. During this time, the state of many stock markets was described as irrational exuberance [28]. It was also found that companies at the end of the millennia that added the postfix *.com* to their corporate title would increase their likelihood to trade at a higher stock price than if they kept their old name [29].

Another inefficiency in the market has been the *January Effect*, early documented in 1983 by Donald B. Keim [30] and later popularized in the book *The Incredible January Effect* by Robert A. Haugen and Josef Lakonisho in 1987. It was found that stocks historically had returned abnormal daily returns between the end of December and the first days of January. The effect of the January Effect has diminished throughout recent years, possibly because it was revealed to the general public. The decrease of the effect might be natural, given how easily it can be exploited. Imagine a society where the January Effect is well known and where some people in this society might consider purchasing stocks a few days before "everyone else" in December. This advantage allows them to enter the market before stock prices rise due to the increasing demand and buying pressure from the rest of society. They might then consider selling their shares on the 1^{st} of January, again, before everyone else, to get out of the market before it starts falling again due to sales pressure from the remaining crowds. There exist a joke on Wall Street where it is said that the January Effect will more likely happen on the previous Thanksgiving than in January [27]. Malkiel's analysis concludes that the EMH tends to be the rule, even though exceptions occur at times, but where the market seems to correct any mispricings as time passes [27].

6.3 Stock Price Predictors

One can study the stock price predictors decision making in Figure 6.2 and Figure 6.3.

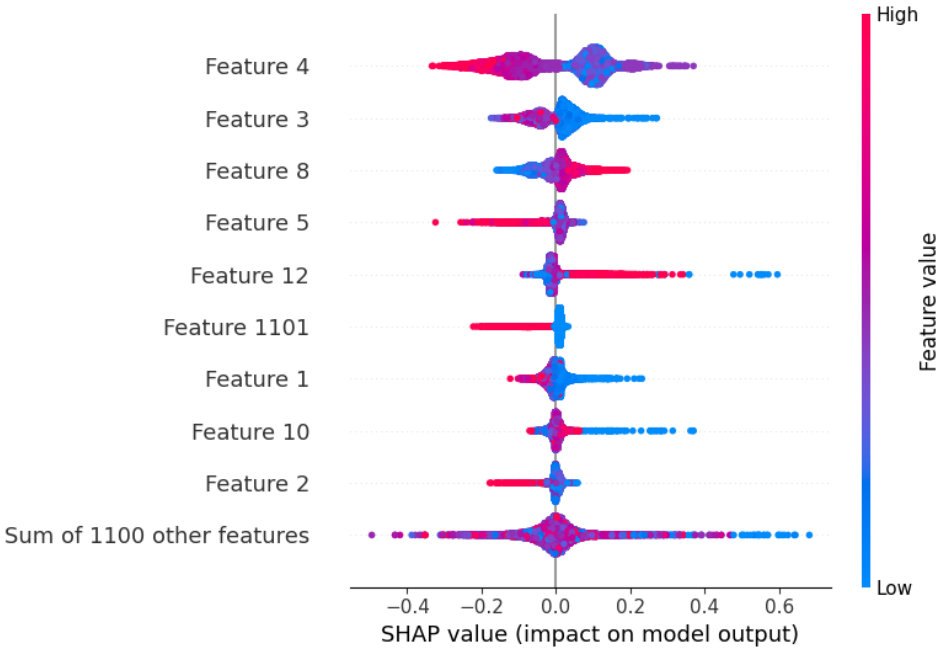

Figure 6.2: The SHAP plot of the $XGB_{\Phi,pd}$ model.

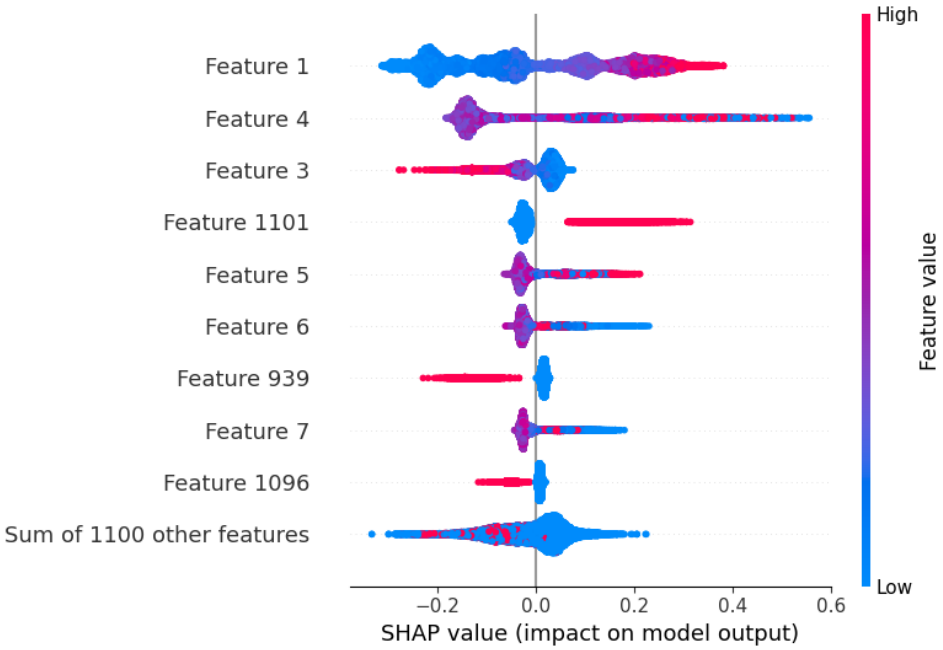

Figure 6.3: The SHAP plot of the $XGB_{\Phi,lp}$ model.

We can see that Feature 1101, corresponding to the `earnings_report` (ER) category, is being used as the sixth, respectively the fourth most valuable feature for the Stock Price Direction Predictor and the Large Price Movement Predictor in the SHAP (Shapley Additive Explanations) plot. The category is mainly used as an indication of the downward direction for the Price Direction Predictor. The absence of the category does not give any strong indications of how the price will move, however. The Large Price Movement Predictor uses the category as an indicator for large price movements, where the absence of the press release again does not tell much.
The stock price predictors performed relatively well compared to the baseline. However, the stocks studied were the stocks that still were available in the databases, and this may introduce some *survivorship bias* [31]. Thus, one cannot exclude the possibility that increased predictability may be credited due to this bias, which should be considered when interpreting the results.
Limitations to implement this system in the open market should be highlighted. The thesis assumes that the opening price is the price at which stocks may be purchased. This assumption may be theoretically possible, but rarely the case in the open stock market. The quantity of the opening price of the studied stocks remains unknown. Any trading fees have not been taken into consideration. The purpose of this thesis has been to study the predictability of stock movements rather than any profitability. Stock prediction assumes that history will repeat itself, which is not always the case, and should be taken into account as we in this thesis have only studied a limited nine-year period.

6.4 Future Work

A few topics were not investigated in this thesis but have been left to future work. The weak supervision approach covered roughly a quarter of the available press releases, and labeling was mainly done by pattern matching. This limitation leaves room for future work to design additional LFs to cover remaining unknown unlabeled press releases.
Machine learning techniques to increase the generalization of the generative Snorkel model by building an end classifier are also left for future work. A suggested approach would be to utilize a deep learning model with word embedding layers to increase the recall of the string pattern matching approach utilized in this thesis [18].
One may improve the performance of the stock price predictors by utilizing a deep learning model, given previous success in this field [32].
Finally, an actual implementation of a trading system built on the techniques applied in this thesis would also be interesting to study to see how these methods would work in practice.

7
Conclusion

Press releases' impacts on stock returns in the Nordic stock market have been studied in this thesis. The result has shown that weak supervision and the data programming framework is a practical approach for data analysis in real world-settings, allowing us to analyze stock returns from different types of press releases for previously unlabeled data.
Stock prediction remains a challenging task, yet the stock prediction was feasible solely on market data in the Nordic stock market, a statistically significant result compared to the baseline model. Finally, it appears that press release labeling through weak supervision can increase the predictability to some extent for the Stock Price Predictor models. However, further research will be needed to confirm this observation.

Bibliography

[1] B. G. Malkiel and E. F. Fama, "Efficient capital markets: A review of theory and empirical work*," *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1970.tb00518.x

[2] V. Niederhoffer, "The analysis of world events and stock prices," *The Journal of Business*, vol. 44, no. 2, pp. 193–219, 1971.

[3] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.

[4] A. Ratner, C. De Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," *Advances in neural information processing systems*, vol. 29, p. 3567, 2016.

[5] W. Jiang, "Applications of deep learning in stock market prediction: recent progress," *arXiv preprint arXiv:2003.01859*, 2020.

[6] S. Basu, "Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis," *The journal of Finance*, vol. 32, no. 3, pp. 663–682, 1977.

[7] W. F. De Bondt and R. Thaler, "Does the stock market overreact?" *The Journal of finance*, vol. 40, no. 3, pp. 793–805, 1985.

[8] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[9] E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Neural networks for signal processing VII. Proceedings of the 1997 IEEE signal processing society workshop*. IEEE, 1997, pp. 276–285.

[10] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[11] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[12] Alex Ratner, Stephen Bach, Paroma Varma, Chris Ré, "An Overview of Weak Supervision," https://www.snorkel.org/blog/weak-supervision, 2017, online; accessed May 26, 2021.

[13] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[14] X. J. Zhu, "Semi-supervised learning literature survey," 2005.

[15] B. Settles, "Active learning literature survey," 2009.

[16] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 1003–1011.

[17] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[18] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 11, no. 3. NIH Public Access, 2017, p. 269.

[19] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith, "Predicting risk from financial reports with regression," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2009, pp. 272–280.

[20] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The azfin text system," *ACM Transactions on Information Systems (TOIS)*, vol. 27, no. 2, pp. 1–19, 2009.

[21] R. Luss and A. d'Aspremont, "Predicting abnormal returns from news using text classification," *Quantitative Finance*, vol. 15, no. 6, pp. 999–1012, 2015.

[22] Z. Zhou, M. Gao, Q. Liu, and H. Xiao, "Forecasting stock price movements with multiple data sources: Evidence from stock market in china," *Physica A: Statistical Mechanics and its Applications*, vol. 542, p. 123389, 2020.

[23] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *arXiv preprint arXiv:1310.4546*, 2013.

[24] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[25] N. I. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 24–38, 2009.

[26] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[27] B. G. Malkiel, "The efficient market hypothesis and its critics," *Journal of Economic Perspectives*, vol. 17, no. 1, pp. 59–82, March 2003. [Online]. Available: https://www.aeaweb.org/articles?id=10.1257/089533003321164958

[28] R. C. Shiller, "Irrational exuberance," *Philosophy and Public Policy Quarterly*, vol. 20, no. 1, pp. 18–23, 2000.

[29] M. J. Cooper, O. Dimitrov, and P. R. Rau, "A rose. com by any other name," *The journal of Finance*, vol. 56, no. 6, pp. 2371–2388, 2001.

[30] D. B. Keim, "Size-related anomalies and stock return seasonality: Further empirical evidence," *Journal of Financial Economics*, vol. 12, no. 1, pp. 13–32,

1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0304405X83900259

[31] S. J. Brown, W. Goetzmann, R. G. Ibbotson, and S. A. Ross, "Survivorship bias in performance studies," *The Review of Financial Studies*, vol. 5, no. 4, pp. 553–580, 1992.

[32] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

A

Labeling Functions

The labeling function and the label matrix can be studied in greater detail in this chapter.

A.1 Helper Functions

```
re_sos = r"\A[^\.]*"
re_eos = r"[^\.]*[\.!?](\s|\\\\n)+[A-Z]"

all_years = r"(2011|2012|2013|2014|2015|2016|2017|2018|2019|2020)"
all_months = r"(January|February|March|April
        |May|June|July|August|September|October|November|December)"

corporate_titles = ["Chief Administrative Officer", "Chief  Analytics Officer",
 "Chief Brand Officer", "Chief Business Development Officer",
 "Chief Business Officer", "Chief Commercial Officer",
 "Chief Communications Officer", "Chief Compliance Officer",
 "Chief Creative Officer", "Chief Content Officer",
 "Chief Customer Officer", "Chief Data Officer",
 "Chief Design Officer", "Chief Digital Officer",
 "Chief Diversity Officer", "Chief Events Officer",
 "Chief Executive Officer", "Chief Experience Officer",
 "Chief Financial Officer", "Chief Gaming Officer",
 "Chief Genealogical Officer", "Chief Growth Officer",
 "Chief Human Resources Officer", "Chief Information Officer",
 "Chief Information Security Officer", "Chief Innovation Officer",
 "Chief Investment Officer", "Chief Knowledge Officer",
 "Chief Learning Officer", "Chief Legal Officer",
 "Chief Marketing Officer", "Chief Media Officer",
 "Chief Medical Officer", "Chief Operating Officer",
 "Chief People Officer", "Chief Privacy Officer",
 "Chief Process Officer", "Chief Product Officer",
 "Chief Reputation Officer", "Chief Research Officer",
 "Chief Restructuring Officer", "Chief Revenue Officer",
 "Chief Risk Officer", "Chief Scientific Officer",
 "Chief Security Officer", "Chief Services Officer",
 "Chief Solutions Officer", "Chief Strategy Officer",
```

```
 "Chief Supply Chain Officer", "Chief Sustainability Officer",
 "Chief Technological Product Designer", "Chief Technology Officer",
 "Chief Technical Officer", "Chief Visibility Officer",
 "Chief Visionary Officer", "Chief Web Officer",
 "President", "Chairman", "Board Member"]

corporate_titles_acronyms = [r"\bCAO\b", r"\bCBO\b", r"\bCBDO\b",
 r"\bCCO\b", r"\bCDO\b",
 r"\bCVO\b", r"\bCEO\b", r"\bCXO\b", r"\bCFO\b", r"\bCGO\b",
 r"\bCHRO\b", r"\bCIO\b", r"\bCISO\b", r"\bCINO\b", r"\bCKO\b",
 r"\bCLO\b", r"\bCMO\b", r"\bCOO\b", r"\bCPO\b", r"\bCRO\b",
 r"\bCSO\b", r"\bCSCO\b", r"\bCTPO\b", r"\bCTO\b", r"\bCVO\b",
 r"\bCWO\b"]

def regex_alternatives(keywords):
    return ("|").join(keywords)

corporate_titles_joined = regex_alternatives(corporate_titles)
corporate_titles_acronyms_joined = regex_alternatives(corporate_titles_acronyms)

nlp = spcy.load("en_core_web_sm")
pre_spacy_content = SpacyPreprocessor(text_field="content",
doc_field="doc", memoize=True)
pre_spacy_title = SpacyPreprocessor(text_field="title",
doc_field="doc", memoize=True)

def contains_report_words(x):
    if re.search(r"(annual|interim|financial)\sreport", x.title,
     flags=re.I):
        return True
    elif re.search(r"(\bq1\b|\bq2\b|\bq3\b|\bq4\b)\sreport",
     x.title, flags=re.I):
        return True
    elif re.search(r"(first quarter|second quarter|third quarter|fourth quarter)",
     x.title, flags=re.I):
        return True
    elif re.search(r"(January|February|March|April
      |May|June|July|August|September|October
      |November|December).*report", x.title, flags=re.I):
        return True
    elif re.search(r"report.*(January|February|March|April
      |May|June|July|August|September
      |October|November|December)", x.title, flags=re.I):
        return True
    elif re.search(r"report.*(2010|2011|2012|2013|2014
      |2015|2016|2017|2018|2019|2020)", x.title, flags=re.I):
```

```
        return True
    elif re.search(r"(2010|2011|2012|2013|2014
      |2015|2016|2017|2018|2019|2020).*report", x.title, flags=re.I):
        return True
    elif re.search(r"quarterly results", x.title, flags=re.I):
        return True
    elif re.search(r"Financial\sStatement", x.title, flags=re.I):
        return True
    elif re.search(re_sos + r"(net profit|ebit|organic)[\.]*
      (increas|decreas|rise|rose|fell|fall)" + re_eos, x.content, flags=re.I):
        return True
    else:
        return False


def anti_sharebuyback_words(x):
    if re.search(r"shares?\b", x.content, flags=re.I) == None:
        return True
    elif re.search(r"(nok\b|sek\b|eur\b
      |dkk\b|usd\b|cad\b|\bPL\b|\$|kr|€)",
       x.content, flags=re.I) == None and re.search(r"\d",
       x.content, flags=re.I) == None and re.search(r"program",
       x.content, flags=re.I) == None:
        return True
    elif re.search(r"annual\s.*meeting", x.title, flags=re.I):
        return True
    elif re.search(r"bonds?\b", x.title, flags=re.I):
        return True
    elif re.search(re_sos +r"bonds?" + re_eos, x.content, flags=re.I):
        return True
    elif "bond" in x.content.lower() and "maturity" in
      x.content.lower() and "issue" in x.content.lower():
        return True
    else:
        return False


def anti_acquisition_words(x):
    if re.search(r"(company|companies)", x.content, flags=re.I) == None:
        return True

    re_buyback_str = r"(repurchas\w+|buy[\s-]?back|own\sshare
       |purchas.*share|case\sno\.?)"
    if re.search(re_buyback_str, x.title, flags=re.I):
        return True
    EXIST_WORDS_IN_FIRST_SENTENCES =
```

```
      word_exist_in_first_N_sentences(x, 3, re_buyback_str)
    if EXIST_WORDS_IN_FIRST_SENTENCES:
        return True

    if re.search(r"(million|billion)", x.content, flags=re.I) == None:
        return True
    if re.search(r"receiv", x.title, flags=re.I):
        return True
    else:
        return False


def anti_new_business_words(x):
    if re.search(r"(\d|million|billion|thousand)", x.content) == None:
        return True
    elif re.search(r"\?", x.title):
        return True
    elif re.search(r"Award[^\.]*" + all_years, x.title):
        return True
    elif re.search(r"(stock|acqui|\bloan\b)", x.title, flags=re.I):
        return True
    elif re.search(r"annual\s.*meeting", x.title, flags=re.I):
        return True
    else:
        return False

def anti_inside_sell_words(x):
    if re.search(r"(\bacqui|\bsubscrib|\bsales|\bpurchas|\bbuys?\b|
      \bbought|result|sales|\bsubsri)", x.content, flags=re.I):
        return True
    elif re.search(r"\bshare", x.content, flags=re.I) == None:
        return True
    elif re.search(r"(repurch|buy[\s-]?back)", x.content, flags=re.I):
        return True
    elif re.search(r"\bown\bshare", x.content, flags=re.I):
        return True
    elif re.search(r"share[^\.]*saving[^\.]*plan", x.title, flags=re.I):
        return True
    elif re.search(r"share[^\.]*sasving[^\.]*plan", x.content, flags=re.I):
        return True
    elif re.search(re_sos + "subscrib" + re_eos, x.content, flags=re.I):
        return True
    elif re.search(re_sos + "has\ssubscrib" + re_eos, x.content, flags=re.I):
        return True
    else:
        return False
```

```
def anti_inside_buy_words(x):
    if re.search(r"(\bsell|\bresult|\bsales|\bdisposal)", x.content, flags=re.I):
        return True
    elif re.search(r"\bshare", x.content, flags=re.I) == None:
        return True
    elif re.search(r"(repurch|buy[\s-]?back)", x.content, flags=re.I):
        return True
    elif re.search(r"\bown\bshare", x.content, flags=re.I):
        return True
    else:
        return False

def anti_awards_title_words(x):
    if re.search(r"[A-Z]", x.title) == None:
        return True
    elif re.search(r"\?", x.title):
        return True
    elif re.search(r"(contract|deal|stock|apply)", x.title, flags=re.I):
        return True
    elif re.search(r"annual\s.*meeting", x.title, flags=re.I):
        return True
    else:
        return False


def anti_divestment_words(x):
    if re.search(r"(termination|share)", x.title, flags=re.I):
        return True
    else:
        return False

def anti_earningsreport_words(x):
    if re.search(r"(\%|million|thousand|billion)", x.content, flags=re.I) == None:
        return True
    elif re.search(r"(nok\b|sek\b|eur\b|dkk\b|usd\b
      |cad\b|\bPL\b|\$|kr|€)", x.content, flags=re.I) == None:
        return True
    elif re.search(r"(publish|publication)", x.title, flags=re.I):
        return True
    elif re.search(re_sos + r"will[\.]*present" + re_eos, x.content, flags=re.I):
        return True
    else:
        return False
```

```
def word_exist_in_first_N_sentences(x, N, regex_str, person_must_include = False):
    first_N_sentences =' '.join(re.split(r'(?<=[^0-9][.!?])[\s\\\\n]
      +[A-Z\(]', x.content)[:N])
    if re.search(regex_str, first_N_sentences, flags=re.I):
        if person_must_include:
            doc = nlp(first_N_sentences)
            person_included = any([ent.label_ == "PERSON" for ent in doc.ents])
            return person_included
        else:
            return True
    else:
        return False
```

A.2 Share Buyback Labeling Function

```
    def buyback_lf1_func(x, label_lookup):
    regex_str = r"(repurchas\w+|buy[\s-]?back)"
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_BUYBACK = anti_sharebuyback_words(x)
        if NOT_BUYBACK:
            return ABSTAIN

        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(x,
          3, regex_str)
        if EXIST_IN_THREE_FIRST_SENTENCES == False:
            return ABSTAIN

        return label_lookup[SHARE_BUYBACK]
    else:
        return ABSTAIN

def buyback_lf2_func(x, label_lookup):
    regex_str = r"(purchas\w+|bought)[^\.]*(its|own)[^\.]*share"
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_BUYBACK = anti_sharebuyback_words(x)
```

```
        if NOT_BUYBACK:
            return ABSTAIN

        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(x,
          3, regex_str)
        if EXIST_IN_THREE_FIRST_SENTENCES == False:
            return ABSTAIN

        return label_lookup[SHARE_BUYBACK]
    else:
        return ABSTAIN

def buyback_lf3_func(x, label_lookup):
    if re.search(r"(repurchas\w+|buy[\s-]?back)", x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_BUYBACK = anti_sharebuyback_words(x)
        if NOT_BUYBACK:
            return ABSTAIN

        return label_lookup[SHARE_BUYBACK]
    else:
        return ABSTAIN




def make_buyback_lf1(label_lookup):
    return LabelingFunction(
        name="buyback_lf1",
        f=buyback_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_buyback_lf2(label_lookup):
    return LabelingFunction(
        name="buyback_lf2",
        f=buyback_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_buyback_lf3(label_lookup):
    return LabelingFunction(
        name="buyback_lf3",
```

```
        f=buyback_lf3_func,
        resources=dict(label_lookup=label_lookup)
    )
```

A.3 Acquisition Labeling Function

```
    def acquisition_lf1_func(x, label_lookup):
    regex_str = r"acqui[^\.]*(company\b|all[^\.]*share)"
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_ACQUISITION = anti_acquisition_words(x)
        if NOT_ACQUISITION:
            return ABSTAIN

        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(x,
          3, regex_str)
        if EXIST_IN_THREE_FIRST_SENTENCES == False:
            return ABSTAIN

        return label_lookup[ACQUISITION]
    else:
        return ABSTAIN


def acquisition_lf2_func(x, label_lookup):
    regex_str = r"acqui"
    if re.search(regex_str, x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_ACQUISITION = anti_acquisition_words(x)
        if NOT_ACQUISITION:
            return ABSTAIN

        return label_lookup[ACQUISITION]
    else:
        return ABSTAIN



def acquisition_lf3_func(x, label_lookup):
    regex_str = r"purchas[^\.]*(company\b|all[^\.]*share|asset)"
```

```
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_ACQUISITION = anti_acquisition_words(x)
        if NOT_ACQUISITION:
            return ABSTAIN

        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(x,
          3, regex_str)
        if EXIST_IN_THREE_FIRST_SENTENCES == False:
            return ABSTAIN

        return label_lookup[ACQUISITION]
    else:
        return ABSTAIN

def acquisition_lf4_func(x, label_lookup):
    regex_str = r"(buys?\s|bought\s|purchase[sd]?\s)"
    if re.search(regex_str, x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_ACQUISITION = anti_acquisition_words(x)
        if NOT_ACQUISITION:
            return ABSTAIN
        return label_lookup[ACQUISITION]
    else:
        return ABSTAIN


def make_acquisition_lf1(label_lookup):
    return LabelingFunction(
        name="acquisition_lf1",
        f=acquisition_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_acquisition_lf2(label_lookup):
    return LabelingFunction(
        name="acquisition_lf2",
        f=acquisition_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )
```

```
def make_acquisition_lf3(label_lookup):
    return LabelingFunction(
        name="acquisition_lf3",
        f=acquisition_lf3_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_acquisition_lf4(label_lookup):
    return LabelingFunction(
        name="acquisition_lf4",
        f=acquisition_lf4_func,
        resources=dict(label_lookup=label_lookup)
    )
```

A.4 New Deal Labeling Function

```
def new_business_lf1_func(x, label_lookup):
    regex_str = r"been award[^\.]*(\bcontract|\bgrants?\b|\bdesigns?
      |\bpartnership|\bassignment|\bfinanc|\bproject
      |\blicence|\bloi|\bmillion|\bbillion)"
    if re.search(regex_str, x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_NEW_BUSINESS = anti_new_business_words(x)
        if NOT_NEW_BUSINESS:
            return ABSTAIN

        return label_lookup[NEW_BUSINESS]
    else:
        return ABSTAIN


def new_business_lf2_func(x, label_lookup):
    regex_str = r"been award[^\.]*(\bcontract|\bgrants?\b|\bdesigns?\b
      |\bpartnership|\bassignment|\bfinanc|\bproject
      |\blicence|\bloi|\bmillion|\bbillion)"
    if re.search(re_sos + regex_str + re_eos, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_NEW_BUSINESS = anti_new_business_words(x)
        if NOT_NEW_BUSINESS:
```

```
            return ABSTAIN

        return label_lookup[NEW_BUSINESS]
    else:
        return ABSTAIN


#preprocess
def new_business_lf3_func(x, label_lookup):
    regex_str = r"sign[^\.]*(\bcontract|\bgrants?\b|\bdesigns?\b
      |\bpartnership|\bassignment|\bfinanc|\bproject
      |\blicence|\bloi|\bmillion|\bbillion)"
    if re.search(regex_str, x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_NEW_BUSINESS = anti_new_business_words(x)
        if NOT_NEW_BUSINESS:
            return ABSTAIN

        return label_lookup[NEW_BUSINESS]
    else:
        return ABSTAIN

def new_business_lf4_func(x, label_lookup):
    regex_str = r"sign[^\.]*(\bcontract|\bgrants?\b
      |\bdesigns?\b|\bpartnership|\bassignment|\bfinanc
      |\bproject|\blicence|\bloi|\bmillion|\bbillion)"
    if re.search(re_sos + regex_str + re_eos, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_NEW_BUSINESS = anti_new_business_words(x)
        if NOT_NEW_BUSINESS:
            return ABSTAIN

        return label_lookup[NEW_BUSINESS]
    else:
        return ABSTAIN


def make_new_business_lf1(label_lookup):
    return LabelingFunction(
        name="new_business_lf1",
```

```
        f=new_business_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_new_business_lf2(label_lookup):
    return LabelingFunction(
        name="new_business_lf2",
        f=new_business_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_new_business_lf3(label_lookup):
    return LabelingFunction(
        name="new_business_lf3",
        f=new_business_lf3_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_new_business_lf4(label_lookup):
    return LabelingFunction(
        name="new_business_lf4",
        f=new_business_lf4_func,
        resources=dict(label_lookup=label_lookup),
    )
```

A.5 Earnings Report Labeling Function

```
def earnings_report_lf1_func(x, label_lookup):
    if re.search(r"(q1\b|q2\b|q3\b|q4\b|first\squarter
      |second\squarter|third\squarter|fourth\squarter
      |1st\squarter|2nd\squarter|3rd\squarter|4th\s|quarter)",
      x.title, flags=re.I):
        if re.search(r"(report)", x.title, flags=re.I) == None
        and re.search(all_years, x.title, flags=re.I) == None:
            return ABSTAIN
        NOT_FINANCIAL_REPORT = anti_earningsreport_words(x)
        if NOT_FINANCIAL_REPORT:
            return ABSTAIN

        return label_lookup[EARNINGS_REPORT]
    else:
        return ABSTAIN
```

```
def earnings_report_lf2_func(x, label_lookup):
    if re.search(r"(interim.*report|year[-\s]end.* report)", x.title, flags=re.I):
        NOT_FINANCIAL_REPORT = anti_earningsreport_words(x)
        if NOT_FINANCIAL_REPORT:
            return ABSTAIN

        return label_lookup[EARNINGS_REPORT]
    else:
        return ABSTAIN




def make_earnings_report_lf1(label_lookup):
    return LabelingFunction(
        name="earnings_report_lf1",
        f=earnings_report_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_earnings_report_lf2(label_lookup):
    return LabelingFunction(
        name="earnings_report_lf2",
        f=earnings_report_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )
```

A.6 New Key Person Labeling Function

```
def new_keyperson_lf1_func(x, label_lookup):
    if re.search(corporate_titles_acronyms_joined, x.title,
      flags=re.I) or re.search(corporate_titles_joined, x.title,
      flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        re_join = r"(\bappoint|\bnew\b|\btakes?\b|\bstep.*up\b
          |\bhire[sd]?\b|\bhiring\b|\bbecome|\bbecoming
          |\belect|\bstart|\bnames?|\bserves?\b|\bserving
          |\brecruit|\bjoin)"
        if re.search(re_join, x.title, flags=re.I):
            return label_lookup[NEW_KEYPERSON]
        if re.search(r"\bchanges?\b", x.title, flags=re.I):
```

```
            re_join_stronger = r"(\bappoint|\bhire[sd]?\b|\bhiring\b|\belects?\b
              |\belecting\b|\belected\b)"
            IS_JOINING = word_exist_in_first_N_sentences(x,
              2, re_join_stronger, True)
            if IS_JOINING:
                return label_lookup[NEW_KEYPERSON]
        return ABSTAIN
    return ABSTAIN



def new_keyperson_lf2_func(x, label_lookup):
    set_of_titles = None
    re_corp = re.search(corporate_titles_joined, x.content, flags=re.I)
    if re_corp:
        set_of_titles = corporate_titles_joined
    re_corp_acro = re.search(corporate_titles_acronyms_joined,
      x.content, flags=re.I)
    if re_corp_acro:
        set_of_titles = corporate_titles_acronyms_joined
    if re_corp or re_corp_acro:
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        re_join_words = r"(\bappoint|\bhire[sd]?\b|\bhiring\b|\belects?\b
          |\belecting\b|\belected\b)"
        re_join1 = r"(" + set_of_titles + r")[^\.]*"
          + re_join_words + r".*\."
        IS_JOINING1 = word_exist_in_first_N_sentences(x,
          2, re_join1, True)
        if IS_JOINING1:
            return label_lookup[NEW_KEYPERSON]
        re_join2 = re_join_words + r"[^\.]*(" + set_of_titles
          + r").*\."
        IS_JOINING2 = word_exist_in_first_N_sentences(x, 2, re_join2, True)
        if IS_JOINING2:
            return label_lookup[NEW_KEYPERSON]
        return ABSTAIN
    return ABSTAIN

#preprocess
def new_keyperson_lf3_func(x, label_lookup):
    if any([ent.label_ == "PERSON" for ent in x.doc.ents]):
        re_join = r"(\bappoint)"
        if re.search(re_join, x.title, flags=re.I):
```

```
            re_exp = r"(" + corporate_titles_acronyms_joined
              + r"|" + corporate_titles_joined + r")"
            TITLE_EXIST = word_exist_in_first_N_sentences(x
              , 3, re_exp, True)
            if TITLE_EXIST:
                IS_REPORT = contains_report_words(x)
                if IS_REPORT:
                    return ABSTAIN
                return label_lookup[NEW_KEYPERSON]

    return ABSTAIN



def make_new_keyperson_lf1(label_lookup):
    return LabelingFunction(
        name="new_keyperson_lf1",
        f=new_keyperson_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_new_keyperson_lf2(label_lookup):
    return LabelingFunction(
        name="new_keyperson_lf2",
        f=new_keyperson_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_new_keyperson_lf3(label_lookup):
    return LabelingFunction(
        name="new_keyperson_lf3",
        f=new_keyperson_lf3_func,
        resources=dict(label_lookup=label_lookup),
        pre=[pre_spacy_title]
    )
```

A.7 Lost Key Person Labeling Function

```
def lost_keyperson_lf1_func(x, label_lookup):
    if re.search(corporate_titles_acronyms_joined, x.title,
      flags=re.I) or re.search(corporate_titles_joined, x.title,
      flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
```

```
            return ABSTAIN

        re_leave = r"(\bleaves?\b|\bleaving\b|\bquits?\b|\bexits?\b
          |\bresign|\bterminat|\bstep.*down\b|\bdepartur
          |\bfire[sd]?\b|\bfiring\b|\bretire[sd]?\b
          |\bretiring\b)"
        if re.search(re_leave, x.title, flags=re.I):
            return label_lookup[LOST_KEYPERSON]
        if re.search(r"\bchanges?\b", x.title, flags=re.I):
            re_leave_stronger = r"(\bleaves?\b|\bleaving\b|\bquits?\b|\bresign
              |\bfire[sd]?\b|\bfiring\b|\bretire[sd]?\b
              |\bretiring\b)"
            IS_LEAVING = word_exist_in_first_N_sentences(x,
              2, re_leave_stronger, True)
            if IS_LEAVING:
                return label_lookup[LOST_KEYPERSON]
        return ABSTAIN
    return ABSTAIN


def lost_keyperson_lf2_func(x, label_lookup):
    set_of_titles = None
    re_corp = re.search(corporate_titles_joined, x.content, flags=re.I)
    if re_corp:
        set_of_titles = corporate_titles_joined
    re_corp_acro = re.search(corporate_titles_acronyms_joined,
      x.content, flags=re.I)
    if re_corp_acro:
        set_of_titles = corporate_titles_acronyms_joined
    if re_corp or re_corp_acro:
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        re_leave_words = r"(\bleaves?\b|\bleaving\b|\bquits?\b|\bresign
          |\bfire[sd]?\b|\bfiring\b|\bretire[sd]?\b
          |\bretiring\b)"
        re_leave1 = r"(" + set_of_titles + r")[^\.]*"
          + re_leave_words + r".*\."
        IS_LEAVING1 = word_exist_in_first_N_sentences(x,
          2, re_leave1 , True)
        if IS_LEAVING1:
            return label_lookup[LOST_KEYPERSON]
        re_leave2 = re_leave_words + r"[^\.]*("
          + set_of_titles + r").*\."
        IS_LEAVING2 = word_exist_in_first_N_sentences(x,
```

```
          2, re_leave2 , True)
        if IS_LEAVING2:
            return label_lookup[LOST_KEYPERSON]
        return ABSTAIN
    return ABSTAIN


#preprocess
def lost_keyperson_lf3_func(x, label_lookup):
    if any([ent.label_ == "PERSON" for ent in x.doc.ents]):
        re_leave = r"(\bleaves?|leaving|quit|resign|retire|retiring)"
        if re.search(re_leave, x.title, flags=re.I):
            re_exp = r"(" + corporate_titles_acronyms_joined
              + r"|" + corporate_titles_joined + r")"
            TITLE_EXIST = word_exist_in_first_N_sentences(x,
              3, re_exp, True)
            if TITLE_EXIST:
                IS_REPORT = contains_report_words(x)
                if IS_REPORT:
                    return ABSTAIN
                return label_lookup[LOST_KEYPERSON]

    return ABSTAIN



def make_lost_keyperson_lf1(label_lookup):
    return LabelingFunction(
        name="lost_keyperson_lf1",
        f=lost_keyperson_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_lost_keyperson_lf2(label_lookup):
    return LabelingFunction(
        name="lost_keyperson_lf2",
        f=lost_keyperson_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_lost_keyperson_lf3(label_lookup):
    return LabelingFunction(
        name="lost_keyperson_lf3",
        f=lost_keyperson_lf3_func,
        resources=dict(label_lookup=label_lookup),
        pre=[pre_spacy_title]
```

```
    )
```

A.8 Inside Buy Labeling Function

```
def inside_buy_lf1_func(x, label_lookup):
    re_clue = r"(mandory\snotif|transaction|trading|trade|insider)"
    if re.search(re_clue, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        TITLE_EXIST = word_exist_in_first_N_sentences(x,
          1, re_exp)
        if TITLE_EXIST:
            re_buy = r"(\bpurchas|\bbuy|\bacqui|\bbought)[^\.]*
              \d+[^\.]*share"
            if re.search(re_buy, x.content, flags=re.I):
                IS_REPORT = contains_report_words(x)
                if IS_REPORT:
                    return ABSTAIN
                NOT_INSIDE_BUY = anti_inside_buy_words(x)
                if NOT_INSIDE_BUY:
                    return ABSTAIN
                return label_lookup[INSIDE_BUY]

    return ABSTAIN




def inside_buy_lf2_func(x, label_lookup):
    re_buy = r"(\bpurchas|\bbuy\acqui)[^\.]*\d+[^\.]*share"
    if re.search(re_buy, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        TITLE_EXIST = word_exist_in_first_N_sentences(x,
          1, re_exp)
        if TITLE_EXIST:
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_BUY = anti_inside_buy_words(x)
            if NOT_INSIDE_BUY:
                return ABSTAIN
            return label_lookup[INSIDE_BUY]
```

```
    return ABSTAIN

def inside_buy_lf3_func(x, label_lookup):
    re_buy = r"(\bpurchas|\bbuy\acqui)[^\.]*\d+[^\.]*share"
    if re.search(re_buy, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        if re.search(re_exp, x.title, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_BUY = anti_inside_buy_words(x)
            if NOT_INSIDE_BUY:
                return ABSTAIN
            return label_lookup[INSIDE_BUY]

    return ABSTAIN

def inside_buy_lf4_func(x, label_lookup):
    re_clue = r"Nature\sof\sthe\stransaction"
    if re.search(re_clue, x.content, flags=re.I):
        re_clue2 = r"acqui"
        if re.search(re_clue2, x.content, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_BUY = anti_inside_buy_words(x)
            if NOT_INSIDE_BUY:
                return ABSTAIN
            return label_lookup[INSIDE_BUY]


    return ABSTAIN


def inside_buy_lf5_func(x, label_lookup):
    re_clue = r"Buy\sor\sSell.*Buy"
    if re.search(re_clue, x.content):
        return label_lookup[INSIDE_BUY]


    return ABSTAIN


def inside_buy_lf6_func(x, label_lookup):
    re_clue = r"close[^\.]*associat[^\.]*person"
```

```
    if re.search(re_clue, x.content, flags=re.I):
        re_buy = r"(\bpurchas|\bbuy\acqui)[^\.]*\d+[^\.]*share"
        if re.search(re_buy, x.content, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_BUY = anti_inside_buy_words(x)
            if NOT_INSIDE_BUY:
                return ABSTAIN
            return label_lookup[INSIDE_BUY]

    return ABSTAIN



def make_inside_buy_lf1(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf1",
        f=inside_buy_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_inside_buy_lf2(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf2",
        f=inside_buy_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_inside_buy_lf3(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf3",
        f=inside_buy_lf3_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_buy_lf4(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf4",
        f=inside_buy_lf4_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_buy_lf5(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf5",
```

```
        f=inside_buy_lf5_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_buy_lf6(label_lookup):
    return LabelingFunction(
        name="inside_buy_lf6",
        f=inside_buy_lf6_func,
        resources=dict(label_lookup=label_lookup),
    )
```

A.9 Inside Sell Labeling Function

```
def inside_sell_lf1_func(x, label_lookup):
    re_clue = r"(mandory\snotif|transaction|trading|trade|insider)"
    if re.search(re_clue, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        TITLE_EXIST = word_exist_in_first_N_sentences(x,
          1, re_exp)
        if TITLE_EXIST:
            re_sell = r"(\bdispos|\bsells?\b|\bsold\b)[^\.]*\d+[^\.]*share"
            if re.search(re_sell, x.content, flags=re.I):
                IS_REPORT = contains_report_words(x)
                if IS_REPORT:
                    return ABSTAIN
                NOT_INSIDE_SELL = anti_inside_sell_words(x)
                if NOT_INSIDE_SELL:
                    return ABSTAIN
                return label_lookup[INSIDE_SELL]

    return ABSTAIN



def inside_sell_lf2_func(x, label_lookup):
    re_sell = r"(\bdispos|\bsells?\b|\bsold\b)[^\.]*\d+[^\.]*share"
    if re.search(re_sell, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        TITLE_EXIST = word_exist_in_first_N_sentences(x,
          1, re_exp)
        if TITLE_EXIST:
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
```

```
                return ABSTAIN
            NOT_INSIDE_SELL = anti_inside_sell_words(x)
            if NOT_INSIDE_SELL:
                return ABSTAIN
            return label_lookup[INSIDE_SELL]

    return ABSTAIN

def inside_sell_lf3_func(x, label_lookup):
    re_buy = r"(\bdispos|\bsells?\b|\bsold\b)[^\.]*\d+[^\.]*share"
    if re.search(re_buy, x.title, flags=re.I):
        re_exp = r"(" + corporate_titles_acronyms_joined
          + r"|" + corporate_titles_joined + r")"
        if re.search(re_exp, x.title, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_SELL = anti_inside_sell_words(x)
            if NOT_INSIDE_SELL:
                return ABSTAIN
            return label_lookup[INSIDE_SELL]

    return ABSTAIN

def inside_sell_lf4_func(x, label_lookup):
    re_clue = r"Nature\sof\sthe\stransaction"
    if re.search(re_clue, x.content, flags=re.I):
        re_clue2 = r"(\bdisposes?\b|\bdisposal\b)"
        if re.search(re_clue2, x.content, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_SELL = anti_inside_sell_words(x)
            if NOT_INSIDE_SELL:
                return ABSTAIN
            return label_lookup[INSIDE_SELL]


    return ABSTAIN


def inside_sell_lf5_func(x, label_lookup):
    re_clue = r"Buy\sor\sSell.*Sell"
    if re.search(re_clue, x.content):
        return label_lookup[INSIDE_SELL]
```

```
    return ABSTAIN


def inside_sell_lf6_func(x, label_lookup):
    re_clue = r"close[^\.]*associat[^\.]*person"
    if re.search(re_clue, x.content, flags=re.I):
        re_sell = r"(\bdispos|\bsells?\b|\bsold\b)[^\.]*\d+[^\.]*share"
        if re.search(re_sell, x.content, flags=re.I):
            IS_REPORT = contains_report_words(x)
            if IS_REPORT:
                return ABSTAIN
            NOT_INSIDE_SELL = anti_inside_sell_words(x)
            if NOT_INSIDE_SELL:
                return ABSTAIN
            return label_lookup[INSIDE_SELL]

    return ABSTAIN



def make_inside_sell_lf1(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf1",
        f=inside_sell_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_inside_sell_lf2(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf2",
        f=inside_sell_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_inside_sell_lf3(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf3",
        f=inside_sell_lf3_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_sell_lf4(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf4",
        f=inside_sell_lf4_func,
```

```
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_sell_lf5(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf5",
        f=inside_sell_lf5_func,
        resources=dict(label_lookup=label_lookup),
    )

def make_inside_sell_lf6(label_lookup):
    return LabelingFunction(
        name="inside_sell_lf6",
        f=inside_sell_lf6_func,
        resources=dict(label_lookup=label_lookup),
    )
```

A.10 Award Labeling Function

```
def award_lf1_func(x, label_lookup):
    regex_str = r"(Of\\sThe\\sYear|Awards?\b
      |Trophy|Trophies|Medals?|Gala|Prize|Top\s\d)"
    if re.search(regex_str, x.title):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_AWARD = anti_awards_title_words(x)
        if NOT_AWARD:
            return ABSTAIN

        return label_lookup[AWARD]

    return ABSTAIN




def award_lf2_func(x, label_lookup):
    regex_str = re_sos + r"recognised[^\.]*" + re_eos
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN
```

```
        NOT_AWARD = anti_awards_title_words(x)
        if NOT_AWARD:
            return ABSTAIN

        return label_lookup[AWARD]
    else:
        return ABSTAIN

def award_lf3_func(x, label_lookup):
    regex_str = re_sos + r"\btop\s\d[^\.]*" + re_eos
    if re.search(regex_str, x.content, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN

        NOT_AWARD = anti_awards_title_words(x)
        if NOT_AWARD:
            return ABSTAIN

        return label_lookup[AWARD]
    else:
        return ABSTAIN



def make_award_lf1(label_lookup):
    return LabelingFunction(
        name="award_lf1",
        f=award_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_award_lf2(label_lookup):
    return LabelingFunction(
        name="award_lf2",
        f=award_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )

def make_award_lf3(label_lookup):
    return LabelingFunction(
        name="award_lf3",
        f=award_lf3_func,
        resources=dict(label_lookup=label_lookup),
    )
```

A.11 Divestment Labeling Function

```
def divestment_lf1_func(x, label_lookup):
    if re.search(r"\bdivest", x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        NOT_DIVESTMENT = anti_divestment_words(x)
        if NOT_DIVESTMENT:
            return ABSTAIN
        if IS_REPORT:
            return ABSTAIN
        return label_lookup[DIVESTMENT]
    return ABSTAIN



def divestment_lf2_func(x, label_lookup):
    if re.search(r"\bdispose", x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN
        NOT_DIVESTMENT = anti_divestment_words(x)
        if NOT_DIVESTMENT:
            return ABSTAIN
        return label_lookup[DIVESTMENT]

    return ABSTAIN


def divestment_lf3_func(x, label_lookup):
    if re.search(r"complete[^\.]*(exit|divest)", x.title, flags=re.I):
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN
        NOT_DIVESTMENT = anti_divestment_words(x)
        if NOT_DIVESTMENT:
            return ABSTAIN
        return label_lookup[DIVESTMENT]

    return ABSTAIN

def divestment_lf4_func(x, label_lookup):
    re_exp = r"complete[^\.]*(exit|divest)"
    if re.search(re_exp, x.content, flags=re.I):
```

```
        IS_REPORT = contains_report_words(x)
        if IS_REPORT:
            return ABSTAIN
        NOT_DIVESTMENT = anti_divestment_words(x)
        if NOT_DIVESTMENT:
            return ABSTAIN
        EXIST_IN_THREE_FIRST_SENTENCES = word_exist_in_first_N_sentences(x,
          3, re_exp)
        if EXIST_IN_THREE_FIRST_SENTENCES:
            return label_lookup[DIVESTMENT]
    return ABSTAIN


def make_divestment_lf1(label_lookup):
    return LabelingFunction(
        name="divestment_lf1",
        f=divestment_lf1_func,
        resources=dict(label_lookup=label_lookup)
    )


def make_divestment_lf2(label_lookup):
    return LabelingFunction(
        name="divestment_lf2",
        f=divestment_lf2_func,
        resources=dict(label_lookup=label_lookup)
    )



def make_divestment_lf3(label_lookup):
    return LabelingFunction(
        name="divestment_lf3",
        f=divestment_lf3_func,
        resources=dict(label_lookup=label_lookup)
    )



def make_divestment_lf4(label_lookup):
    return LabelingFunction(
        name="divestment_lf4",
        f=divestment_lf4_func,
        resources=dict(label_lookup=label_lookup)
    )
```

A.12 The Label Matrix

Table A.1: The label matrix which shows the coverage, overlap and conflict of each LF.

Name	Coverage	Overlaps	Conflicts
inside_sell_lf1	0.001234	0.000114	0.000069
inside_sell_lf2	0.000023	0.000023	0.000000
inside_sell_lf3	0.000046	0.000023	0.000000
inside_sell_lf4	0.002880	0.000046	0.000000
inside_sell_lf5	0.000091	0.000000	0.000000
inside_sell_lf6	0.000023	0.000023	0.000000
inside_buy_lf1	0.011565	0.000526	0.000366
inside_buy_lf2	0.000114	0.000069	0.000000
inside_buy_lf3	0.000046	0.000046	0.000000
inside_buy_lf4	0.007611	0.000457	0.000023
inside_buy_lf5	0.000183	0.000000	0.000000
inside_buy_lf6	0.000549	0.000343	0.000000
new_business_lf1	0.000229	0.000137	0.000000
new_business_lf2	0.004206	0.000526	0.000046
new_business_lf3	0.005646	0.002331	0.000091
new_business_lf4	0.009874	0.002880	0.000366
new_business_lf5	0.011428	0.005417	0.000000
new_business_lf6	0.007497	0.005257	0.000023
award_lf1	0.002331	0.000160	0.000069
award_lf2	0.000137	0.000091	0.000023
award_lf3	0.000160	0.000023	0.000000
buyback_lf1	0.066398	0.062056	0.000091
buyback_lf2	0.033919	0.028045	0.000343
buyback_lf3	0.060661	0.059793	0.000023
acq_lf1	0.007520	0.005417	0.000137
acq_lf2	0.014057	0.005326	0.000137
acq_lf3	0.000617	0.000503	0.000069
acq_lf4	0.000960	0.000411	0.000046
earnings_report_lf1	0.039839	0.008000	0.000114
earnings_report_lf2	0.039862	0.007886	0.000000
new_keyperson_lf1	0.016091	0.010034	0.001531
new_keyperson_lf2	0.013943	0.007726	0.001577
new_keyperson_lf3	0.007497	0.006400	0.000503
lost_keyperson_lf1	0.004091	0.002697	0.000983
lost_keyperson_lf2	0.004708	0.003451	0.001920
lost_keyperson_lf3	0.002171	0.001646	0.000457

divestment_lf1	0.004183	0.000891	0.000320
divestment_lf2	0.000114	0.000023	0.000023
divestment_lf3	0.000503	0.000503	0.000000
divestment_lf4	0.000686	0.000480	0.000000

B

Stock Market Prediction

The data used for the stock price prediction task is presented in more detail in the following sections.

B.1 Stock Exchanges

Table B.1: Number of stocks per stock exchange.

Market	n
Nordic Growth Market	16
OMX Copenhagen	101
OMX Helsinki	135
OMX Stockholm	470
Oslo Stock Exchange	180
Spotlight Stock Market	54

B.2 Countries

Table B.2: Number of stocks per country.

Country	n
Denmark	102
Finland	135
Norway	180
Sweden	539

B.3 Sectors

Table B.3: Number of stocks per sector.

Sector	n
Consumables	48
Durable Goods	108
Energy	60
Finance & Real Estate	131

Health Care	185
IT	127
Industry	193
Material	59
Power Supply	34
Telecommunications	11

B.4 Lines of Businesses

Table B.4: Number of stocks per line of business.

Line of Business	n
Accessories	1
Agriculture	2
Banks	24
Betting & Casino	15
Biometrics	11
Capital Management	6
Car & Motor	10
Clothes & Shoes	9
Computers & Hardware	4
Consumer Service	2
Credit & Financing	11
Electronic Equipment	6
Electronic components	8
Electronics & Manufacturing	9
Farmacy	1
Fish farming	9
Food	11
Forestry companies	12
Furniture & Furnishings	6
Gaming & Games	21
Grocery Stores	4
Health products	8
Home Electronics	5
Hotel & Camping	1
Hygiene Products	4
Insurance	8
Investment company	27
Leisure & Sports	6
Mining - Gemstones	1
Mining - Gold & Silver	8
Mining - Prospectus & Operations	9
Niche banks	9
Pharmaceuticals	41
Property - REIT	1

Real Estate	45
Restaurant & Café	1
Retail	7
Shipping & Shipping	26
Tobacco	2
Train & Truck Transport	6
Travel & Entertainment	2
Air Transport	6
Bioenergy	3
Biotech	64
Brewery	7
Broadband & Telephony	6
Building Interiors	4
Building Materials	11
Business & IT Systems	40
Business Consultants	17
Chemicals	16
Communication	16
Construction & Infrastructure	12
Education	2
Electricity supply	5
Energy & Recycling	11
Health Care & Assistive Technology	8
Hospitals & Nursing Homes	6
IT Consultants	21
Industrial components	29
Industrial machinery	17
Installation & Plumbing	16
Internet Services	4
Marketing	10
Measurement & Analysis	10
Media & Publishing	12
Medical Equipment	66
Military & Defense	2
Mining - Industrial metals	6
Oil & Gas - Drilling	8
Oil & Gas - Exploitation	22
Oil & Gas - Sales	2
Oil & Gas - Service	17
Oil & Gas - Transport	11
Packaging	7
Payment & E-Commerce	3
Renewable energy	9
Residential construction	3
Security	4
Security & Surveillance	10

Solar power	13
Space & Satellite Technology	3
Staffing	3
Support Services & Service	6
Telecom services	5
Wind power	4