Bachelor of Science in Computer Science
February 2021

Predicting the Movement Direction of OMXS30 Stock Index Using XGBoost and Sentiment Analysis

Elena Podasca

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science. The thesis is equivalent to 10 weeks of full-time studies.

The authors declare that they are the sole authors of this thesis and that they have not used any sources other than those listed in the bibliography and identified as references. They further declare that they have not submitted this thesis at any other institution to obtain a degree.

Contact Information:
Author(s):
Elena Podasca
E-mail: elpo19@student.bth.se

University advisor:
Suejb Memeti
Department of Computer Science

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

Background. Stock market prediction is an active yet challenging research area. A lot of effort has been put in by both academia and practitioners to produce accurate stock market predictions models, in the attempt to maximize investment objectives. Tree-based ensemble machine learning methods such as XGBoost have proven successful in practice. At the same time, there is a growing trend to incorporate multiple data sources in prediction models, such as historical prices and text, in order to achieve superior forecasting performance. However, most applications and research have so far focused on the American or Asian stock markets, while the Swedish stock market has not been studied extensively from the perspective of hybrid models using both price and text derived features.
Objectives. The purpose of this thesis is to investigate whether augmenting a numerical dataset based on historical prices with sentiment features extracted from financial news improves classification performance when predicting the daily price trend of the Swedish stock market index, OMXS30.
Methods. A dataset of 3,517 samples between 2006 - 2020 was collected from two sources, historical prices and financial news. XGBoost was used as classifier and four different metrics were employed for model performance comparison given three complementary datasets: the dataset which contains only the sentiment feature, the dataset with only price-derived features and finally, the dataset augmented with sentiment feature extracted from financial news.
Results. Results show that XGBoost has a good performance in classifying the daily trend of OMXS30 given historical price features, achieving an accuracy of 73% on the test set. A small improvement across all metrics is recorded on the test set when augmenting the numerical dataset with sentiment features extracted from financial news.
Conclusions. XGBoost is a powerful ensemble method for stock market prediction, reflected in a satisfactory classification performance of the daily movement direction of OMXS30. However, augmenting the numerical input set with sentiment features extracted from text did not have a powerful impact on classification performance in this case, as the improvements across all employed metrics were small.

Keywords: Machine learning, XGBoost, Sentiment analysis, Stock market prediction, OMXS30.

LIST OF ABBREVIATIONS

AB AdaBoost
ANN Artificial Neural Networks
AUC Area Under the Curve
CART Classification and Regression Tree
DNN Deep Neural Networks
DT Decision Tree
ET Extra Trees, standing for extremely randomized trees
GDP Gross domestic product, it expresses the market value of all the final goods and services produced by an economy in a specific time period
GRU Gated Recurrent Unit
KNN k-Nearest Neighbors
LR Logistic Regression
LSTM Long Short-Term Memory
MCC Matthews Correlation Coefficient
MKL Multi-kernel Learning
NB Naïve Bayes
OHLCV open, high, low, close price, and trading volume for a security
RB RobustBoost
RF Random Forest
ROC Receiver Operating Characteristic
SLR Stepwise Logistic Regression
SVM Support Vector Machine
VC Voting Classifier
XGB XGBoost

ACKNOWLEDGEMENTS

I would like to thank my supervisor Suejb Memeti for his valuable feedback and guidance throughout this thesis.

CONTENTS

1 INTRODUCTION

Accurate prediction of financial asset prices and market trends is one of the major concerns for investors in their endeavours to place profitable trades. However, asset price and market trend forecasting is a challenging task. The movement of financial time series such as stock prices are influenced by many exogenous factors such as news, investor sentiment, economic environment etc., which make them noisy and difficult to predict.

Considerable effort has been in put by academia in the past decades to develop forecasting models for the stock market [1], [2], [3], [4]. Traditional statistical models such as autoregressive moving average, conditional heteroscedasticity and their extended versions have been used for financial time series forecasting with good results, while logistic regression has been employed for predicting the directional movement of as asset prices [5].

While the theory behind these models is well established and understood, statistical models have limitations, as they fail to capture the complexity and non-linearity in the data [6]. Driven by data availability and increased computational power, machine learning models have shown better performance compared to statistical ones in financial time series forecasting in a range of problems [7]. Support vector machine, ensemble methods such as random forest have been popular choices in the literature [8], [9], while in practice, ensemble methods such as XGBoost have proven very successful in various Kaggle competitions [10].

Many research works have focused on stock price prediction. However, from a financial trading perspective, the actual price of a security is less important since the profit is generated by correctly anticipating the direction of price change. Research has shown that trading strategies based on classification models generate higher risk-adjusted returns than regression models [11]. Therefore, framing the problem as classification and predict the direction of movement instead of the nominal price of a security is sufficient.

In practice, traders and investors rely on a variety of information sources for decision making such as corporate disclosures, stock price and macroeconomic data, news and even social media. Evidence suggests that there is a relationship between sentiments extracted from financial text and stock market movement [12]. As such, there is a substantial body of literature analyzing the role of market participants' sentiment extracted from news and social media in financial market forecast [13]. Given the significant advancements being made in the field of text mining and natural language processing in recent years, there has also been a growing interest in models that combine textual analysis with machine learning techniques [12], [14].

The vast majority of these studies, however, have focused mainly on the major stock markets such as the US or China. The Swedish stock market is a domain that has not been studied extensively from the perspective of the applicability of ensemble machine learning models for market trend prediction based on disparate data sources.

The purpose of this thesis is to investigate the classification performance of XGBoost in predicting the daily up or down movement of the Swedish stock market index, OMXS30[1], when using two different types of features, based on historical price data and sentiments extracted from financial news, respectively. This is framed as a two-step binary classification problem. The first step is to collect and extract sentiments from a set of financial news. The second step is to augment the existing price information dataset with the sentiment feature and apply XGBoost as classifier to predict the daily price trend of OMXS30.

The results of this work will contribute to the existing body of knowledge in two ways. Firstly, they will convey whether XGBoost can serve as an effective classifier for trend prediction of the OMXS30. Secondly, it will shed more light as to whether including sentiments extracted from text can boost classification performance. These results may provide guidance to finance practitioners interested in trading index based financial instruments on the Swedish stock market.

[1] https://indexes.nasdaqomx.com/Index/Overview/OMXS30

1.1 Aim and objectives

The aim of this thesis is to investigate whether adding a sentiment feature derived from text to a selected feature set can increase the predictive performance of a tree-based ensemble model - XGBoost, when classifying the daily up or down price movement of the OMXS30 stock index.

In order to reach this goal, several objectives are derived as follows:

1) Collect financial textual data and extract sentiments accordingly.
2) Collect historical price data, perform pre-processing, feature extraction and selection.
3) Identify the best XGBoost specification (model selection).
4) Select appropriate metrics for evaluating model performance.
5) Train and test the model on three complementary datasets: the numerical and augmented datasets as well as the baseline input consisting of the sentiment feature only.
6) Compare classification performance when employing the different datasets.

1.2 Research questions

To accomplish the aim and objectives of this study, two research questions have been defined:

1) How well does XGBoost perform in predicting the daily price movement (up/down) of OMXS30?
2) What is the predictive power and classification performance impact of sentiments extracted from financial news?

1.2.1 Expected outcome

Regarding the first research question, the findings presented in related works in Chapter 2 indicate a classification performance for XGBoost that varies widely between roughly 61% - 83%. Although these results might not be directly comparable to this analysis' due to different datasets and features used, it is reasonable to expect the classification accuracy of XGBoost falling in this interval.

For the second research question, the expectation is in line with literature findings. That is, sentiment features extracted from text have predictive power and that adding them to the input set of price-derived features will significantly improve classification performance.

1.3 Background

This section introduces the main concepts used in the thesis. As the research topic is investigating how sentiments extracted from financial text can enhance the predictability of stock market index price using machine learning models, subsequent subsections in this chapter will introduce the relevant economic and computer science notions.

The following concepts will be introduced: stock market indices, the theoretical finance framework for stock market prediction as well as sentiment analysis. Furthermore, an introduction to machine learning and description of XGBoost will be made.

1.3.1 Stock market indices

Stock market indices are treated as proxies for stock markets as a whole [15]. They usually consist of the most actively traded stocks on respective stock markets. As stock market proxies, they are important in the pricing of other stocks, as proposed in theory by various asset pricing models [16]. As such, a variety of derivative instruments have stock indices as underlying asset, which makes stock indices popular tradeable securities in practice.

OMXS30 is the Swedish stock market index and it comprises of the 30 most actively traded Swedish companies.

1.3.2 Theoretical framework for stock market prediction

Several theories exist in finance to explain stock market behavior and predictability. One such theory is the random-walk hypothesis [17], which claims that changes in stock market prices are random and thus cannot be predicted.

Another prominent theory in financial economics with regards to the predictability of stock markets is the efficient-market hypothesis [18]. In its least stringent form, it posits that all past information is reflected in current stock prices and as a result, analyses of such information cannot provide investors with an advantage in the market.

Despite the stands of the efficient market hypothesis, considerable amount of research suggests that markets have at least some degree of predictability, when addressing the problem from a behavioral finance standpoint. Traditionally, there are two main views on stock market predictability, depending on what information is used for prediction, namely technical analysis and fundamental analysis [19].

Technical analysis assumes that future prices can be predicted based on patterns found in historical prices. For finding such patterns, a multitude of technical indicators are computed from open, high, low and close prices as well as volume (OHLCV) information. These figures are available for financial assets at each time interval, i.e. daily, hourly, every 15 minutes etc. An overview on technical analysis can be found in [20].

Fundamental analysis is based on company specific as well as macroeconomic information in order to evaluate the firms' prospects for profitability and thus future share price. At market level, some relevant indicators include GDP growth and interest rates.

1.3.3 Sentiment analysis

Sentiment analysis is a body of research concerned with mining views and opinions expressed in text [21]. The goal is to identify the emotional polarities in text, in order to classify whether it carries a positive, negative or neutral sentiment. Sentiment analysis can be performed at the document level, sentence level and aspect level. Research suggests that sentiments extracted from text play an important role in stock predictions [12].

Scholars have studied the impact of news and investor sentiment on financial assets' performance. In the field of behavioral finance for instance, they have commonly employed parametric models for investigating the relationships between independent variables on one hand, and asset prices, returns or movement direction on the other hand, based on economic theory [22], [23]. In the field of computational intelligence on the other hand, studies usually augment datasets with textual data in order to enhance the predictive power of forecasting models.

In the literature, the textual data to perform financial sentiment analysis on comes from three main sources: news media, corporate disclosure, and user generated content (UGC) such as blog and social media posts. News is regarded as a credible and reliable source of information in regards to fundamentals [23], and thus fit for analyzing a broader range of securities [23], while UGC reflects the mood of retail investors [22] and may be more suitable for small market capitalization securities [23].

A plethora of methodologies exist for sentiment analysis and classification: manual extraction, rules- and knowledge-based, dictionary-based, methods based on regular machine learning techniques and, more recently, methods based on deep neural networks[2]. Given the sheer number of methods and techniques available, an exhaustive introduction is beyond the scope of this thesis however, an overview of text mining and sentiment analysis techniques for finance is presented in [19] and [24].

[2] For an overview on artificial neural networks and deep learning, see [25].

1.3.4 Machine learning

Machine learning is an area of artificial intelligence which studies computer algorithms that learn from data [25, pp.2]. Machine learning algorithms can be used for a broad range of tasks such as classification, regression, clustering, anomaly detection etc.

Learning is achieved by first training the models on available data and then use the resulting models to perform the required task on new unseen data. Depending on the type and how much supervision machine learning systems get during training, there are four major categories of machine learning, briefly presented below.

Supervised learning

In supervised learning, the training data that is fed into the algorithm contains the expected outputs, also known as labels. A typical supervised learning task is classification, for example predicting whether an email is spam or not. Another common type of supervised learning algorithms is regression, where values are predicted instead of a limited number of classes, for example, predicting stock prices.

In this thesis, since the data is labeled, we are dealing with a supervised learning problem. The labels are stock index price movements ("up/down"). Therefore, classification algorithms are appropriate for prediction instead of regression.

Unsupervised learning

In unsupervised learning, the training data is unlabeled. Some algorithms included in this category are clustering, association rule learning and anomaly detection.

Semi-supervised learning

Semi-supervised learning is useful when obtaining labels for the entire training set is either expensive or ineffective. Instead, the algorithms can be used on partially labeled datasets. This is usually achieved by combining supervised and unsupervised algorithms [25, pp.13]. For instance, restricted Boltzmann machines are trained sequentially using unsupervised techniques, after which the system is fine-tuned in a supervised manner.

Reinforcement learning

Reinforcement learning is a different computational approach in which an agent learns by observing and interacting with the environment. The agent selects and performs an action based on a policy in order to maximize a reward. Financial trading can be set up as a reinforcement learning problem where the trading robot places trades in order to maximize expected returns.

In the following three subsections a background is given on the supervised learning algorithms and systems used in the research method for this thesis.

1.3.4.1 Decision trees

Decision trees are a non-parametric supervised learning method used for both regression and classification. Similar to the tree data structure, they consist of nodes and leaves.

When using numeric attributes, usually, the node tests the attribute value with a constant. The leaf nodes give a classification that applies to all instances that reach that leaf. An example of a decision tree classifier is presented in Fig. 1.1.

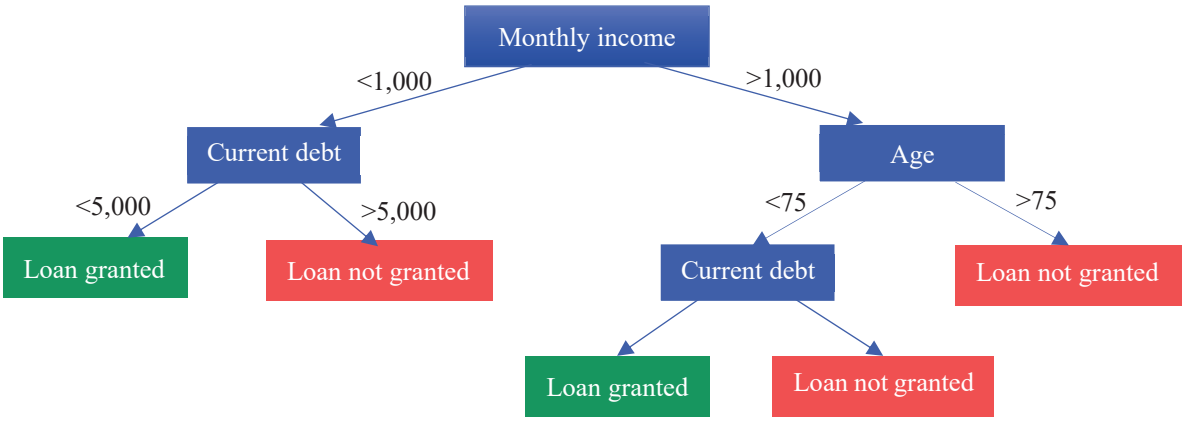

Fig. 1.1. Example of a decision tree classifier

The goal is to create a model that predicts the value of a target variable by learning simple decision rules from the data features. One of the most popular methods for building trees is CART (classification and regression tree) introduced by Breiman [26], which produces binary trees.

The primary challenge in the decision tree implementation is to identify which features should be chosen at each node. The criterion by which this is achieved in the CART algorithm is the Gini impurity [25, pp.177]:

$$G_i = 1 - \sum_{k=1}^{n} {p_{i,k}}^2$$

, where $p_{i,k}$ is the ratio of class k instances among the training instances in the i^{th} node. Purity of a node means that all training instances that it applies to belong to the same class. The Gini impurity can be interpreted as a cost function used to evaluate splits in the dataset.

Tree growing is achieved through recursive partitioning which works by splitting the training set in two non-overlapping subsets, based on a feature k from the feature set and a threshold t_k, such that the following cost function is minimized:

$$J(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}$$

, where $m_{left/right}$ is the number of instances in the left/right subset and $G_{left/right}$ measures the impurity of the left/right subset. The recursive splitting yields a tree like structure, and this procedure continues until a stopping criterion is met.

CART have several attractive properties as machine learning algorithms [27]. One of them is that they are non-parametric models and therefore do not require the data to belong to a specific type of distribution. Furthermore, CART are not particularly impacted by outliers in the input data. They can also use the same variables multiple times in different parts of the tree, thus revealing complex patterns and interdependencies between the variables. Furthermore, the tree like structure has higher explainability and interpretability compared to statistical methods. A drawback with tree-based models is that they are sensitive to the input data. Slight changes in the training dataset can result in very different trees.

1.3.4.2 Decision tree ensembles and gradient boosting

In machine learning, it is possible to improve predictive performance by aggregating the predictions of a group of weak predictors, which individually perform only slightly better than random chance. This is referred to as ensemble learning.

A powerful ensemble method is boosting, where many predictors (in this case decision trees) are trained sequentially and each subsequent predictor aims to correct the previous one [25, pp.199].
A popular boosting method is gradient boosting, proposed by Friedman [28]. The main idea is that each new predictor attempts to improve the residual error produced by its predecessor. This is practically a numerical optimization problem where the goal is to minimize the loss of the model by adding weak learners using a procedure similar to gradient descent.

With gradient descent, the objective is to update a set of parameters in order to minimize a loss function. In gradient boosting however, instead of parameters, decision trees are used as weak learners. After calculating the loss at each iteration, a new tree is added to the model aiming to reduce the loss, while all existing trees are left unchanged. The model is thus defined as stage-wise additive because the existing trees are not modified. Since trees are basically functions that map inputs to outputs, this approach is also called gradient descent with functions.

Gradient boosting is built upon a generic framework and can handle a large variety of loss functions, the only requirement is that they are differentiable. The type of functions depends on the type of problem. For regression, mean squared loss could be used, while for classification the logistic loss is appropriate:

$$L(\varphi) = \sum_{i=1}^{n} y_i \ln\left(1 + e^{-\hat{y}_t}\right) + (1 - y_i) \ln\left(1 + e^{\hat{y}_i}\right) \tag{1.1}$$

Since gradient boosting is a greedy algorithm, it can lead to overfitting of the training dataset, that is, the model fits the training data to a high degree but performs poorly on unseen data [25, pp.27]. Several regularization methods can be applied to basic gradient boosting to mitigate this problem, including tree constraints and shrinkage [29]. These are some of the hyperparameters of XGBoost [30].

Tree constraints are relevant because individual tree learners need to remain weak. Several parameters can be adjusted in order to constrain trees to prevent them from becoming too complex, such as the number of trees or tree depth. The number of trees should be increased incrementally until no further improvement is observed. In regard to tree depth, short, less complex trees are preferred to deeper, more complex ones.

Additionally, a learning rate (or shrinkage) is applied to further reduce overfitting. This mechanism was first proposed in [28]. This reduces the influence of each individual tree and leaves room for subsequent trees to improve the model. The consequence is that learning is slowed down, and a larger number of trees is needed for the model. Hence, there is an inverse relationship between learning rate and number of trees and a careful tuning of these hyperparameters should be done in practice.

1.3.4.3 XGBoost

XGBoost stands for “extreme gradient boosting” and is an efficient and scalable open-source implementation of the gradient boosting algorithm, suitable for classification and regression problems [10]. The starting point is a regularized learning objective [30]:

$$obj(\varphi) = L(\varphi) + R(\varphi)$$

, where L is a differentiable training loss function and R is the regularization term. For binary classification, the loss function can commonly take the form of the logistic loss as in equation (1.1). The output of the model is produced by an ensemble of trees. Since trees are functions mapping inputs to outputs, it means that objective is a function of functions. One implication of this fact is that traditional techniques for parameter optimization cannot easily be implemented. Hence, the stage-wise additive model is implemented. The mathematical foundation behind XGBoost is presented in [10] and [30] for further reference.

An interesting enhancement behind XGBoost is stochastic gradient boosting, an algorithm proposed by Friedman [31]. The concept behind is that trees are greedily grown from subsamples of the

training set. At each iteration, instead of the full sample a random subsample is drawn without replacement and is used to fit the weak learner. The subsampling can be done in several ways, either by subsampling rows or columns before creating each tree, or by subsampling columns before considering each split. Column subsampling is deemed to be beneficial in preventing overfitting [10].

Other enhancements include optimized handling of sparse data, support for parallel learning as well as out-of-core computations, offering good performance in large scale tasks. XGBoost is designed to be computationally efficient and almost always faster than other gradient boosting implementations. As a result, is it a popular solution in practice for supervised learning tasks, including many competitions on the Kaggle competitive data science platform [10].

1.4 Outline

The rest of this thesis is organized as follows. Chapter 2 presents relevant works in the field of predicting the trend of stocks and stock market indices using machine learning techniques. Chapter 3 outlines the proposed method and procedures employed in this thesis, including data collection, preprocessing, feature model selection as well as the datasets used in the analysis. The performance metrics for model evaluation are introduced. Further, Chapter 4 presents the results of the experiments performed using the datasets. A discussion of these results is presented in Chapter 5, before concluding the thesis and offering suggestions for future work in Chapter 6.

2 RELATED WORK

The following two subsections will present related works in the field of stock market trend prediction using classification algorithms with features extracted from both historical price data and text.

2.1 Stock trend prediction based on numerical features

A vast amount of research works has studied stock index movement prediction with machine learning models. Numerous different methods are employed in the literature, both when it comes to feature selection and algorithms used.

A recurrent theme is that feature selection is of utmost importance. Shen et al. [32] include global stock market indices as well as foreign exchange rates and commodities prices in the feature set, with the goal to predict the daily trend of three major US stock market indices. Applying SVM as classifier with the top four most relevant features as input, the authors report a classification accuracy of over 70%, and positive profitability in their trading simulation.

When it comes to the choice of machine learning algorithms for stock price trend prediction, performance results obtained in the literature often differ depending on the dataset used [33], [34], [35], choice of inputs [32], forecast horizon [36], or whether the economic evaluation takes transaction costs into consideration [9].

Patel et al. [37] compare the classification performance of four different algorithms - ANN, RF, SVM and NB to predict the trend of two stocks and two stock indices on the Bombay Stock Exchange. Ten technical indicators are used as input, both as continuous values and trend deterministic values (discrete values or either 1 or -1). For a dataset comprising of daily observations between years 2003 - 2012, their findings are that when using continuous values, RF has the highest classification accuracy of 83.59%, while NB has lowest accuracy of 73.31%. Interestingly, when trend deterministic input is used, classification accuracy is boosted. NB achieves highest average classification accuracy of 90.19%, followed closely by RF at 89.98% accuracy.

Huang [38] employs the same features and algorithms as in [37] for predicting the trend of the Taiwan stock exchange index. The dataset spans between years 2000 - 2018. The author finds that ANN has highest classification accuracy at 70.2% when using continuous input values, while NB has lowest accuracy of 58.46%. When trend deterministic input is used, all four algorithms have similar performance, with accuracies between 74 - 77%, SVM being the best performing algorithm.

The suitability of machine learning classifiers for a stock recommender system is explored in [34]. The authors compare the performance of single classifiers such as NB, DT, SVM and KNN with that of ensemble models (AB, RB and Bagging). Using a dataset of 293 stocks from the Bombay stock exchange and 10 technical indicators as input, their results indicate that RB and Bagging ensemble models generate most profits, while minimizing the amount of losing trades.

A comparison of various ensemble models is provided in [35], using a dataset of eight randomly chosen stocks from three different stock exchanges. 40 technical indicators are used as features, after which principal components analysis is performed. Experiments were conducted using RF, AB, XGB, ET and VC. On the selected dataset, XGB obtains an average accuracy of 82.66%, third most performant algorithm after VC and ET. ET yielded best performance on the test dataset, with an average performance of 83.75%.

With the continuing surge in computing power and development of sophisticated deep neural network (deep learning) models for a wide range of learning tasks, an increasing number of papers investigate the performance of such models for financial time series forecasting applications. Some studies compare the performance of deep learning models vis-à-vis traditional machine learning models. Yuan et al. [33] compare the performance of six traditional machine learning classification algorithms - CART, NB, RF, LR, SVM, XGB and six deep learning algorithms in the context of day stock trading using a comprehensive dataset of 424 constituent stocks from the S&P 500 index and 185 constituents from the Chinese CSI 300 stock index. Their experiments with and without transaction costs yield varying results. Traditional machine learning algorithms perform better in most of the directional evaluation indicators, with XGB being the best performing with 66% accuracy. However, these

algorithms are sensitive to transaction costs. Deep learning models, on the other hand, have better performance when transaction costs are considered.

2.2 Stock trend prediction using textual data

Advancements in textual analysis and natural language processing techniques have fueled a large body of research focusing on hybrid models for stock market prediction using both numerical and text data. The text data is sourced from a variety of sources such as news, corporate disclosures and social media. In recent years, the availability of Twitter data has led to a growing body of research regarding sentiment analysis and how it relates to companies' stock performance [22]. However, extracting market sentiment from tweets is challenging, since this data tends to be noisy, unstructured, and grammatically incorrect [39].

Fewer papers analyze the predictability of stock market indices based on both numerical and textual data, compared to those where only price-based features are taken into consideration. Furthermore, most of the research is conducted on specific companies and as such, the results tend to vary from case to case. Geva and Zahavi [40] develop a stock recommender system that uses sentiments extracted from company related news together with historical market data. They use a dataset comprising high frequency price data for 72 companies from the S&P500 stock index during 11.5 months between 2006 and 2007. In their trading simulations they use three algorithms, SLR, ANN and DT. They conclude that augmenting numerical inputs with textual data yields superior economic performance in trading simulations, and that using more advanced textual representations further enhances predictive accuracy.

Rahman et al. [41] use text mining to extract sentiment from financial news, which is subsequently used in a machine learning based prototype for investment decision on the Malaysian stock exchange. The textual data consists of nearly 15,000 news articles regarding five listed Malaysian companies. They use SVM for classifying the stocks' trend based on the textual input and achieve an average accuracy of 56%.

For predicting the following day's trend of the Indian stock market index, Bhat and Kamath [42] employ an ANN model with technical indicators and sentiment extracted from web articles and user generated content related to the Nifty index. They compare model performance with and without sentiment analysis and they find that the model with sentiment data generate an increased accuracy from 54% to 61-71%, depending on the availability of the textual input.

Teoh et al. [43] compare the performance of a GRU model with both news-based and numerical data with nine benchmark machine learning models for the next 10 days' trend direction for several US technology stocks as well as the NASDAQ index. The index dataset comprises 1,007 samples ranging from 2012 to 2016, with 25 news headlines being selected each day. The benchmark models are LSTM, RNN, DT, RF and SVM. In their experiments, SVM models perform the best, with an average classification accuracy of 87%, while the deep learning models have a poor performance of 51.2% on average. However, their findings indicate that adding news-based sentiment data to the input set increases the accuracy of the GRU model significantly from 50.1% to 78.57% on average.

Bouktif et al. [44] compare the performance of five supervised learning algorithms, SVM, LR, RF, XGB and ANN in predicting the trend of the Amazon stock price. They perform experiments on datasets consisting of combinations of OHLCV data with sentiment features extracted from tweets. Their results indicate that only using OHLCV data yields poor performance not significantly better than random chance. When augmenting the dataset with sentiment features, ensemble methods yield highest performance boost of around 10%, reaching an accuracy in the interval 61.2 - 62.7%.

Li et al. [12] explored the accuracy impact of including news sentiments when predicting stock index price movement on the Hong Kong stock exchange. They compare three models, MKL, LSTM and SVM using four different industry indices and four approaches to news sentiment analysis. The results indicate that the algorithms yield varying performance for each of the four indices. Further, the authors find that using a finance domain specific dictionary to model the news sentiment performs better compared to general purpose sentiment analysis tools.

Similar findings are presented in [45], where the authors use only text-derived features to investigate their capability in predicting the intraday price trend of 13 stocks on the Moroccan Stock

Exchange. They use a dataset of nearly 6,000 articles written in French and apply two separate methods for extracting features from text. First, they create a dictionary of ca 400 words that are deemed to impact the trend of a stock's price. Afterwards, they use the bag-of-words technique to extract most relevant features. Finally, they apply five supervised machine learning algorithms, SVM, LR, NB, KNN, DT to analyze which one can best predict stock price trends using features extracted from article headlines and corpus, respectively. Their results indicate that using bag-of-words for feature selection yields worst results due to high dimensionality. KNN and SVM obtain the highest accuracy of 57.77%, while DT yields 54% accuracy. Using the custom financial dictionary increased the accuracy for all algorithms with 2.83% on average. The increase in accuracy for DT was 3.29%.

3 METHOD

In order to answer the research questions in this thesis, experiments are conducted with three complementary datasets. Classification performance is measured employing four metrics described in section 3.8. The set of procedures for the experiments is presented in Fig. 3.1, while the features are introduced in Section 3.4.

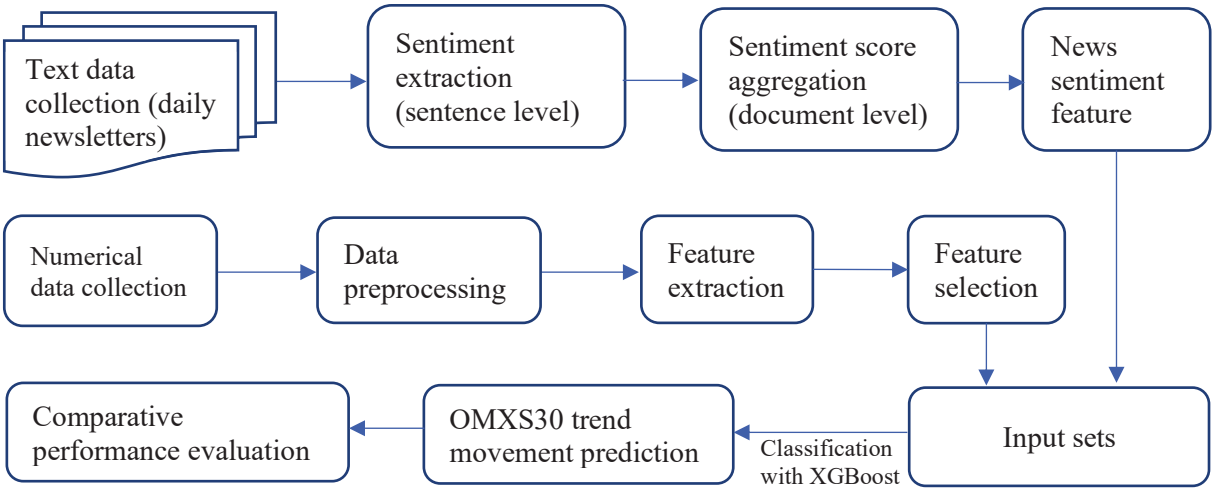

Fig. 3.1. Proposed workflow for the thesis

3.1 Environment description

All work was conducted on a laptop with the following specifications:

- Intel i5-4200 Dual Core CPU
- 8 GB RAM
- NVIDIA GeForce 750M GPU
- Windows 8.1 64-bit OS

In order to facilitate the data collection, preprocessing as well as the conduction of experiments using machine learning models, several software packages and libraries were used, as described in Table 3.1.

Table 3.1. Software packages used in the experiments for this thesis

Library/Software	Description
Python v3.7	Open-source programming language [46].
Jupyter Notebook v1.0.0	Web-based notebook environment for interactive computing [47].
Numpy v1.18.5	Fundamental package for array computing with Python [48].
Scipy v1.4.1	Scientific library for Python [49].
Scrapy v2.4.1	Open-source library for web scraping [50].
Scikit-learn v0.23.1	A set of modules for machine learning and data mining in Python [51].
Pandas v1.0.5	Data analysis library for Python [52].
Pandas TA v0.1.97b0	Technical analysis library for Python [53].
Matplotlib v3.2.2	Plotting library for Python [54].
Seaborn v0.10.1	Statistical data visualization library for Python based on Matplotlib [55].

3.2 Data collection

In this analysis, two types of data are used, textual data and numerical time series from technical indicators and price data. The following two subsections describe how each type of data was collected.

3.2.1 Text data

As outlined in subsections 1.3.3, various types of text can be used for extracting sentiments in a financial context, including news and social media. However, it was mentioned in Chapter 2 that social media text is difficult to analyze as it is unstructured and grammatically incorrect. Therefore, for the scope of this thesis, financial news is chosen for sentiment analysis, as it is deemed as a credible and reliable source [24].

A total of 3,348 daily newsletters were scraped from www.placera.nu, which is an online financial news platform managed by Avanza Bank[3]. Avanza is one of the largest financial services providers in Sweden, servicing over 1.2 million customers. The newsletters were scraped in a *.csv* file using Scrapy, a web scraper for Python. The period for the collected data is 18/09/2006 - 18/09/2020 and is determined by the earliest date for online availability.

The newsletters share a format that describes the previous day's developments in global financial markets, as well as the direction of movement of the OMXS30 index at opening call. They are usually released before market opening and consist of title, a short introduction and the article body. The data of interest that was used in this analysis was the introduction paragraphs, which contain sentiment information about the main global stock markets as well as the Swedish stock market. This text data was chosen as it presents the price movement direction of relevant stock market indices in a compact format, and thus has potentially high informational value for investors. An example of such an introductory paragraph is provided in Table 3.2.

Table 3.2 Example of title and introduction from daily newsletter, translated from Swedish

Date	**Title**	**Introduction**
2020-09-18	*Svag öppning väntas*	*Börserna på Wall Street stängde lägre på torsdagen medan Asien handlades upp högre under morgonen. Ledande terminer indikerar en öppning i negativt territorium för Stockholmsbörsen.*
	Translation: *Weak opening awaits*	Translation: *The Wall Street stock exchanges closed on lower levels yesterday while the Asian ones were up this morning. Futures indices indicate an opening in negative territory for the Stockholm stock exchange.*

3.2.2 Numerical data

The numerical data consists of daily OHLCV data for several financial time series, among which OMXS30. The close price is adjusted for stock splits and dividends. Fig. 3.2 displays the daily development in closing prices for OMXS30. Price data was downloaded in *.csv* format for approximately the same period as for the text data, 14/09/2006 - 19/09/2020, adding two samples in the beginning for calculating lagged returns in the feature extraction stage. There are 3,517 samples in total.

[3]https://avanza.se/start

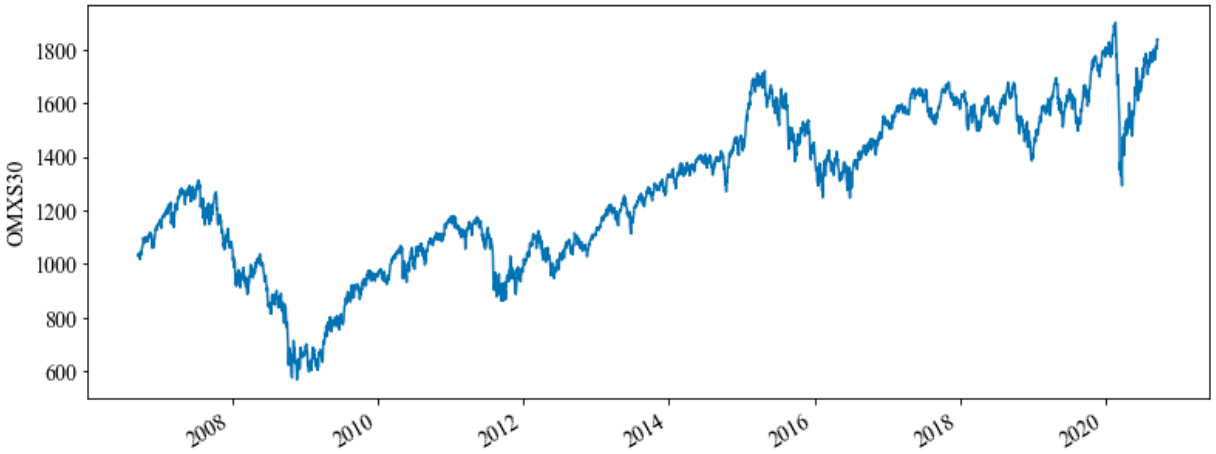

Fig. 3.2. OMXS30 daily closing prices for the entire dataset

The same price data is obtained for two of the most relevant Asian and American stock indices, Hang Seng and S&P500, respectively. This is inspired by both the information content of the newsletters as well as previous literature analyzing the relationships between stock trend movements and inter-market factors [32]. Fig 3.3 depicts the interplay of global equity markets with respect to trading hours.

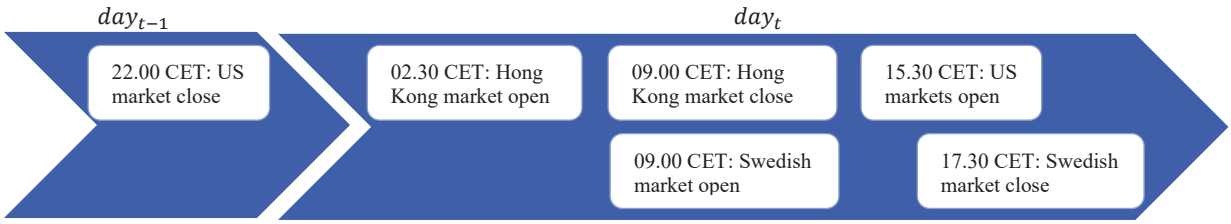

Fig. 3.3. Trading hours for US, Asian and Swedish stock markets

Furthermore, daily OHLCV data was obtained for two currency pairs, USDSEK and EURSEK, gold and WTI oil futures prices as economic proxies as well as the VIX index as proxy for market volatility, with reference to existing literature [32].

All data is publicly available and was retrieved from investing.com. Where there were any missing values, these were fetched from Yahoo Finance where available.

For answering the second research question and comparing classification performance when augmenting the dataset with sentiment feature extracted from text, the out-of-sample evaluation method is used. The dataset is split into a training and test set, where 2,817 samples are used for training and 700 for testing. This corresponds to an 80/20 split.

3.3 Data preprocessing

Given that the obtained dataset is not very large, all the data is consolidated in a single Excel file for preprocessing. The first step is to align the samples by date. There are 175 rows with missing newsletters. These were replaced with numerical values based on the sign of the average return for the US and Asian markets. The details of how daily stock index returns are calculated are provided in Section 3.4 on feature extraction. Additionally, since there are various holidays on the Asian and American markets which do not coincide with the Swedish ones, missing price values for the closed markets are imputed as the prices from the last active trading day.

3.4 Feature extraction

The next step in the method is to derive the relevant features used in the datasets for the experiment. For this thesis, three types of features will be extracted from text and numerical price data: technical indicators for OMXS30, sentiment features extracted from text as well as returns. The process for each is described in subsections 3.4.1 - 3.4.3 below.

3.4.1 Technical indicators

Based on the collected OHLCV data, 10 technical indicators were computed for OMXS30, based on the method employed by Patel et al. [37]. We refer to this paper for an overview of all the indicators. The technical indicators are then discretized, based on the authors' findings that employing variables in trend deterministic form improve classification performance. The discretized indicators take the values 1 or -1. A detailed description of the selected indicators and the discretization procedure is provided in Section 3.5.

The technical indicators are then computed using the Pandas TA library. An important aspect is to compute them up to previous day, otherwise it would be cheating to predict today's price movement based on technical indicators that use today's closing price. Thus, all technical indicators are lagged by 1.

3.4.2 Sentiment analysis

Unstructured text from financial news cannot be used directly in machine learning models. It must be processed and transformed in a machine-readable format. As presented in subsection 1.3.3, several approaches exist for sentiment classification including manual scoring, lexicon- and rules-based, regular machine learning and deep learning-based methods.

The advantage of machine learning models is that they can be trained on labeled data and automatically classify sentiments in the test data. However, if the data contains different types of sentiments and entities as it often is the case in financial texts, the models may not yield high classification accuracy. While significant advancements have been made in the field of natural language processing, few of these sophisticated deep learning models are available for the Swedish language, and none that addresses the financial domain in particular. Therefore, the same challenges as for machine learning algorithms remain.

While open-source tools for sentiment analysis in Swedish exist, such as Vader[4], these are trained on general, non-financial user generated content. As such, sentiment scores might be inaccurate, as in finance, some words such as "red" have negative meaning while in the common language it is neutral. Research suggests that financial domain dictionaries such as Loughran-McDonald[5] are better suited to extract sentiments from financial text compared to general purpose sentiment analysis tools [12]. However, an equivalent for the Swedish language is not available and direct translation to English using automatic tools might not be entirely accurate. Therefore, given that the size of the text dataset is not very large and the document structure and content are homogenous, it is appropriate to perform manual sentiment extraction at sentence level for the selected texts. Manual annotation is deemed to contribute to higher data quality.

The following procedure is employment for sentiment extraction from the introductory paragraphs. Each sentence in the paragraph receives a score from {-2, -1, 0, 1, 2}, where -2 expresses most negative and 2 most positive opinions. Afterwards, the sentiment is aggregated at document level by averaging the sentiments from each sentence. Hence, each document has a score that is a rational number between [-2, 2].

[4] https://github.com/cjhutto/vaderSentiment
[5] https://sraf.nd.edu/textual-analysis/resources/

3.4.3 Additional features

Predicting the direction of movement (up/down) for OMXS30 means that the output is a binary variable and a classification algorithm is appropriate. The output is defined as:

$$y = \begin{cases} 1, & if \ \frac{C_t}{C_{t-1}} > 1 \\ 0, & otherwise \end{cases} \tag{3.1}$$

, where C_t is today's closing price for OMXS30. Therefore, 1 corresponds to positive price movement, while 0 denotes a negative price movement. The distribution of up/down movements per year is presented in Table 3.3.

Table 3.3. Distribution of OMXS30 up/down movements per year

Year	Increase	%	Decrease	%	Total
2006	40	54.79%	33	45.21%	73
2007	134	53.60%	116	46.40%	250
2008	113	44.84%	139	55.16%	252
2009	131	52.19%	120	47.81%	251
2010	134	52.96%	119	47.04%	253
2011	129	50.99%	124	49.01%	253
2012	136	54.40%	114	45.60%	250
2013	137	54.80%	113	45.20%	250
2014	131	52.61%	118	47.39%	249
2015	135	53.78%	116	46.22%	251
2016	128	50.59%	125	49.41%	253
2017	129	51.39%	122	48.61%	251
2018	127	50.80%	123	49.20%	250
2019	141	56.40%	109	43.60%	250
2020	97	53.59%	84	46.41%	181

The daily returns for intermarket features are computed in a similar fashion. As daily stock returns exhibit autocorrelation, previous days' returns may have predictive power. Hence, for all stock indices used, we also compute lagged returns for the previous three days as follows:

$$R\_CC_t = \frac{C_t}{C_{t-1}} - 1 \tag{3.2}$$

, where C_t is price at day t. Returns can be calculated based on both close and opening prices. To leverage the latest information available for at market open, for the US markets we compute the following intraday return:

$$R\_CO_t = \frac{C_t}{O_t} - 1 \tag{3.3}$$

, where O_t is the opening price for day t. At 9 a.m. CET (Swedish stock market open), the latest available daily data for the US market is previous day's closing price, hence, the latest information available is the intraday return for the previous day.

Finally, to align with sentiment data, the daily stock index returns are discretized into four bins:

$$R_d = \begin{cases} 2, & if \ r > \delta \\ 1, & if \ r \epsilon (0, \delta] \\ -1, & if \ r \epsilon [-\delta, 0] \\ -2, & if \ r < -\delta \end{cases} \tag{3.4}$$

, where r is the return calculated according to equation (3.2) and δ is a threshold that separates the slightly positive/negative returns belonging to a more neutral range from the more negative/positive returns, in alignment with the sentiments extracted from the text data. δ is set to 0.3%.

3.5 Feature selection

The feature extraction process yields 56 features in total, comprising both continuous and discrete features.

In order to extract the most relevant features, a correlation filter is applied, since using a large number of features would lead to overfitting and increase computation time. For an explanation of the correlation coefficient, see [25, pp. 58-59]. A total number of 10 features are selected as in previous literature [37], including the aggregated sentiment feature. These are employed in the analysis for this thesis. The features are selected based on the highest absolute correlation coefficient with the output variable computed as per equation (3.1).

An overview of features used in the selection process as well as their correlation matrix are presented in Table A.1 and Fig. A.1 in the Appendix. The features used for this analysis are described in the following subsections 3.5.1 and 3.5.2.

3.5.1 Technical indicators

Simple moving average

The simple moving average is a simple technical indicator which calculates the average price for a specified range of trading days. Here the default of 10 days is used.

$$SMA_{10} = \frac{C_t + C_{t-1} + \cdots + C_{t-9}}{10}$$

,where C_t is the closing price at day t. In trend deterministic context, if current price is above the moving average, the variable takes the value 1, otherwise -1.

Weighted moving average

The weighted moving average is a simple technical analysis tool where more weight is put on the more recent data. The default value of 10 days is used.

$$WMA_{10} = \frac{nC_t + (n-1)C_{t-1} + \cdots + C_{t-9}}{n + (n-1) + \cdots + 1}$$

If the current price is above the weighted moving average, the trend deterministic variable takes the value 1, otherwise -1.

Stochastic oscillator

The stochastic oscillator is a momentum indicator that compares the current closing price to a range of prices over a specific period of time:

$$\%K = \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}}$$

, where HH_t and LL_t are the lowest low and the highest high in the past t days. The default value of 14 days is used. If the value at time t is higher than the value at $t-1$, the trend deterministic variable takes the value 1, otherwise -1.

Relative Strength Index

The Relative Strength Index (RSI) is a popular momentum indicator used to evaluate the overbought or oversold position in a security, based on the closing prices of a recent trading period. The default period n is 14 days. The RSI takes values between 0 and 100.

$$RSI = 100 - \frac{100}{1 + (\sum_{i=0}^{n-1} Up_{t-i}/n)/(\sum_{i=0}^{n-1} Dw_{t-i}/n)}$$

, where Up and Dw are upward and down price change at time t, respectively. When the value of RSI is below 30, the security is considered oversold. Then there is a likelihood that the price will go up, and the trend deterministic value is 1. If RSI is over 70, the security is overbought, which means the price might drop in the near future. Therefore, the trend deterministic value is -1.

Williams %R

Williams %R is a momentum indicator that takes values between -100 and 0 and, similar to RSI, measures overbought and oversold levels for a security. It compares an asset's closing price to the high-low range over a specified period. The period used in this analysis is the default 14 days.

$$Williams\ \%R = \frac{H_n - C_t}{H_n - L_n} \times 100$$

, where L_t and H_t are the low and high prices at time t. If the indicator is ascending compared to previous period, then the trend deterministic value is 1. Otherwise it is -1.

3.5.2 Additional features

- Chicago Board Options Exchange Volatility Index (VIX) is a real-time market index that measures the market's volatility expectations over the coming 30 days. It is used to measure the level of fear and stress in the market.
- The discretized OMXS30 return according to equations (3.3) and (3.4).
- The discretized S&P500 previous day return according to equations (3.3) and (3.4).
- The discretized HK50 current day return according to equations (3.2) and (3.4).
- The aggregated sentiment values for each text document.

3.6 Model selection

For predicting the daily trend of OMXS30, XGBoost is the selected classification algorithm, given its popularity and good results in supervised learning challenges [10]. It will be applied on three complementary datasets:

1) the numerical dataset consisting of nine features derived from price data.
2) the augmented dataset being the same as above, plus the additional sentiment feature extracted from the news data (10 features in total).
3) the dataset consisting of only the sentiment feature is used for the baseline scenario.

Comparing model performance on these datasets will answer the second research question whether augmenting the numerical dataset with features extracted from text enhances the model's predictive performance.

For comparing performance metrics on the three test sets, an optimal set of hyperparameters for XGBoost first needs to be identified [30]. A combined procedure using cross-validation and grid search

will be performed on the numerical dataset to identify the hyperparameters that yield best classification performance without overfitting or underfitting the training dataset. Finally, the model with best hyperparameters will be fitted and evaluated on the remaining two datasets.

The cross-validation and grid search procedures are explained in the following two subsections.

3.6.1 Cross-validation

In statistics, cross-validation is normally employed to obtain an unbiased estimation of model performance, but it can also be used during model selection for identifying the best hyperparameters for the final model. For classification tasks, k-fold cross validation is typically employed [25, pp.73]. In this approach, the dataset is split into *k* partitions, the algorithm is trained on *k-1* partitions and then tested on the remaining subset.

The average of the *k* outcomes is then used as an estimator of model performance of the algorithm. Cross-validation with three, five or ten folds are common choices. However, this method is not suitable as is for time series data, where the ordering of the samples is important and must be maintained. One suitable approach for time series is the rolling-origin-recalibration validation method, described in [56]. Based on this method, we use five validation subsets for model selection and comparative performance evaluation. A depiction of the cross-validation approach is provided in Fig. 3.4.

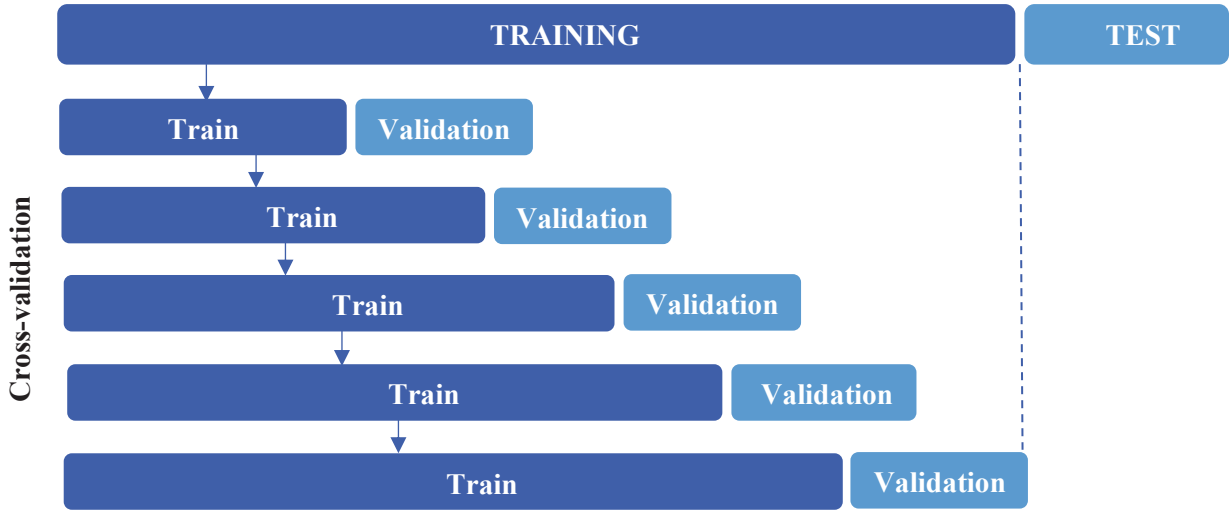

Fig. 3.4. Cross-validation process of a 5-fold time series splitting method

3.6.2 Grid search

To perform a fair comparison of classification performance when using three different datasets, it is necessary that the models use the same hyperparameters. Furthermore, the hyperparameters must be optimized, so that a model is chosen that performs well on the test set and does not overfit or underfit the training data. Grid search is a commonly used method for exploring the hyperparameter space [25, pp. 76]. The main idea is to generate a grid of all hyperparameter combinations from a predefined space. In combination with cross-validation, the use of grid search can mitigate the risk of overfitting, while selecting suitable hyperparameters that enhance classifier performance. The method entails that each set of hyperparameters from the grid is evaluated and validated in accordance with a time series split cross-validation scheme.

It must be taken into account that evaluating each hyperparameter set through grid search is a time consuming and computationally expensive procedure, even more so with cross-validation. However, since each hyperparameter set is independent, grid search can easily be parallelized.

To achieve a reasonable balance between resource consumption and search breadth, 1,200 parameter combinations was used for model selection. Ultimately, whether the grid search is successful depends on finding the best set of hyperparameters among the predefined search values.

Before running the expanding grid search with 1,200 combinations, an initial grid search is done separately to identify the optimal column sampling rate, which is useful for further reducing overfitting. A column sampling rate of 0.1 is identified from the range {0.1, 0.2….,1}. The remaining hyper-parameter candidates selected for the experiments are presented in Table 3.4.

Table 3.4. XGBoost hyperparameter candidates

Hyper-parameter	Search space	Default value
learning_rate (shrinkage)	{0.005, 0.01, 0.05, 0.1, 0.3}	0.3
n_estimators	{50, 100, 200, 500}	100
max_depth	{3,4,5,6}	6
gamma	{0, 0.1, 0.2}	0
min_child_weight	{1,2,4,8,16}	1

Finally, since we are dealing with a binary classification problem, in XGBoost we use the logistic loss. Accuracy is chosen as evaluation metric.

3.7 Feature importance

Feature importance are techniques where a score is assigned to each explanatory variable based on how useful they are at predicting the output variable [57]. For gradient boosting algorithms, importance provides a score that shows how valuable each feature was in the construction of the decision trees within the gradient boosting model. The score is calculated for each feature in the dataset, so that features can be ranked accordingly and compared with one another. The higher the score, the more a particular feature is used as a criterion for node splitting in the trees.

For this thesis, feature importance is computed with the XGBoost package, using as criterion the average gain across all splits that the feature is used in. The gain is a quantity that measures the reduction in training loss when using a feature for splitting [30].

Computing feature importance for all features will reveal further to what extent the sentiment feature is useful in predicting the up/down movements of OMXS30 compared to the features extracted from time series.

3.8 Performance evaluation

In order to answer the research questions, several metrics are computed on the three test sets for comparing classification performance. These are presented in the following subsections.

3.8.1 Accuracy

Accuracy is the rate of correct classifications divided by the total number of predictions made:

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

where: *TP* (true positives) are correctly classified up movements,
TN (true negatives) are correctly classified down movements,
FP (false positives) are down movements incorrectly classified as up movements,
FN (false negatives) are up movements incorrectly classified as down movements.

3.8.2 Confusion matrix

The confusion matrix is a table that describes classifier performance on the test data when the true values are known. True positives, true negatives, false positives and false negatives are displayed for each of the two categories. Fig. 3.5 shows an example of confusion matrix for binary classification.

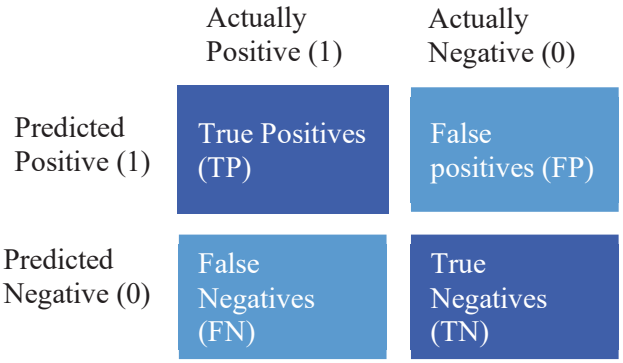

Fig. 3.5. Confusion matrix for binary classification

3.8.3 ROC curve

The ROC (Receiver Operating Characteristic) curve measures the performance of a classifier at various threshold settings. The curve plots two parameters, the true positive rate (TPR) and false positive rate (FPR). The true positive rate measures the ability of a classifier to find all the positive instances. It is also called recall or sensitivity and is defined as:

$$TPR/Recall/Sensitivity = \frac{TP}{TP + FN}$$

The false positive rate is defined as follows:

$$FPR = 1 - Specificity = 1 - \frac{TN}{TN + FP} = \frac{FP}{TN + FP}$$

Plotting the false positive rate on the x-axis and the true positive rate on the y-axis yields a curve similar to the ones depicted in Fig. 3.6.

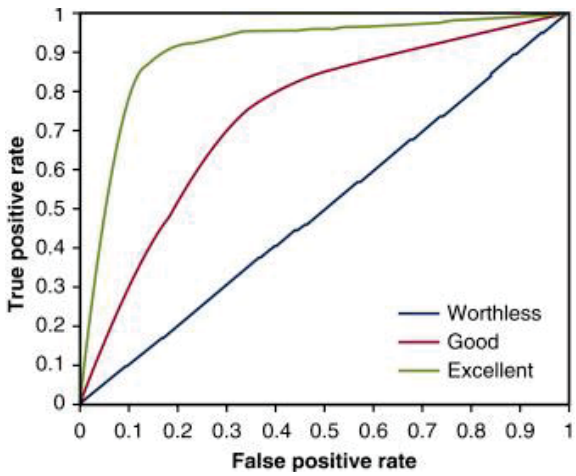

Fig. 3.6. ROC curves example
Source: https://www.jtcvs.org/article/S0022-5223(18)32875-7/fulltext

ROC is a probability curve and the area under it (AUC) measures the degree of separability between classes. AUC takes values between 0 and 1. The closer the ROC follows the left-hand border, the higher the AUC and the better the model is at making accurate predictions. An AUC of 0.5 means that the model is no better than random chance.

3.8.4 Matthews correlation coefficient

This metric shows how correlated model predictions are to the true values. As a correlation coefficient, it takes values between -1 and 1. The value of 1 is achieved when the classifier is perfect (*FP*=*FN*=0), and -1 for the opposite case. A value of 0 indicates that the classifier is not better than random chance.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}$$

4 RESULTS AND ANALYSIS

The set of parameters that was identified as best performing by grid search on the dataset with numerical features is presented in Table 4.1 below.

Table 4.1. Optimal grid search results for XGBoost hyperparameters

XGBoost hyperparameter	Value
learning_rate	0.3
max_depth	3
gamma	0
min_child_weight	16
n_estimators	100

4.1 Feature importance

Fig. 4.1 shows that the 10-day weighted moving average has the highest importance score according to gain. The sentiment feature is only 6th in importance with an F score of 5.33. The results suggest that the sentiment feature is not expected to have a very high contribution to the reduction in accuracy error in the model.

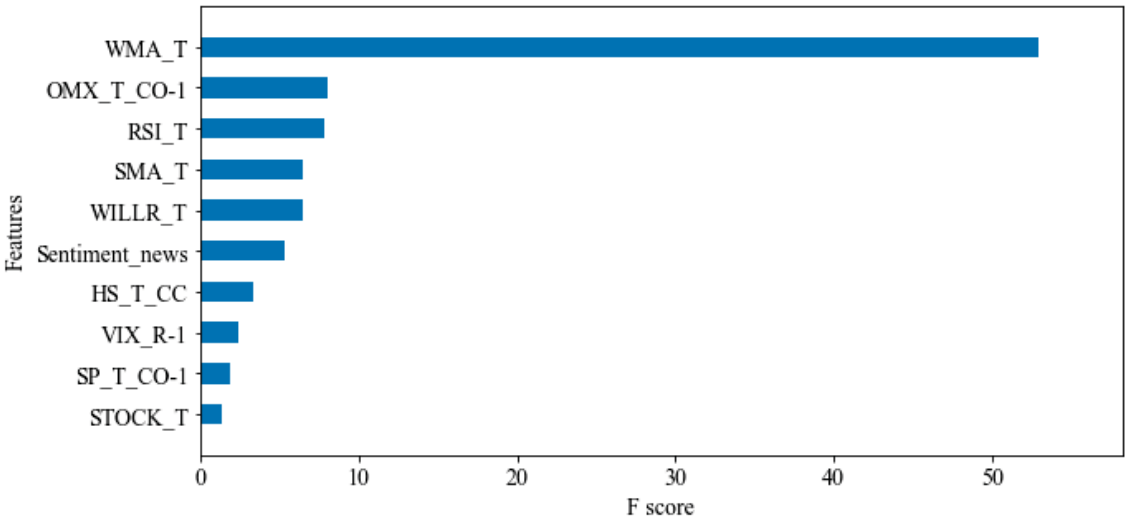

Fig. 4.1. Feature importance

4.2 Comparative analysis of performance metrics

Classification metrics indicate that there is a slight positive difference between the model using both numerical and text features and the one using only numerical features.

Fig. 4.2 shows the accuracy of the algorithm given the three datasets. When using the entire dataset with the sentiment feature included, the accuracy on the test set is 73.71%. The accuracy obtained for the model when only using numerical features is slightly lower, at 73%.

An interesting result is that the baseline case using only the sentiment feature yields an accuracy of 63.85%. This is lower than the first two models, however significantly better than random chance.

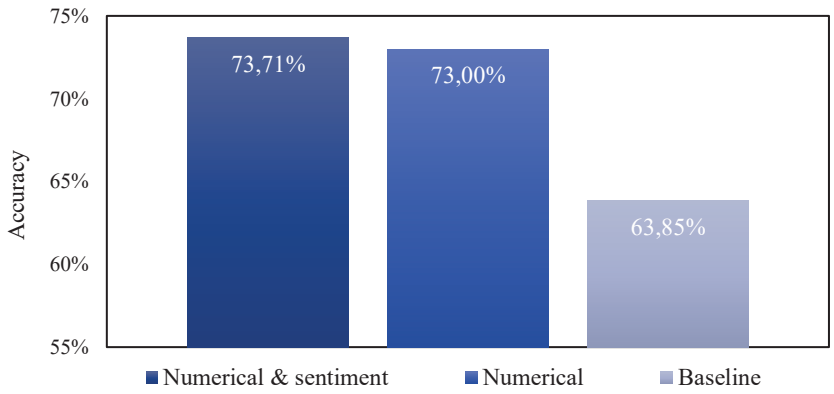

Fig. 4.2. Comparative test set accuracy

The boxplots in Fig. 4.3 depict the cross-validated accuracy obtained on the training dataset in each of the three cases. Using the augmented dataset achieved a cross-validated mean accuracy of 71.85%, with maximum accuracy of 74.84%, minimum of 69.72% and a standard deviation of 1.69%. The cross-validated mean accuracy for the numerical dataset was 71.39%, with maximum accuracy of 73.13%, minimum of 68.87% and a standard deviation of 1.52%. The small difference in mean accuracy of 0.46% is consistent with the results obtained on the test set and presented in Fig. 4.3.
The baseline case where only the sentiment feature was used achieves the poorest performance, with mean accuracy of 61.74%, maximum of 63.75%, minimum accuracy of 59.27% and standard deviation of 1.46%.

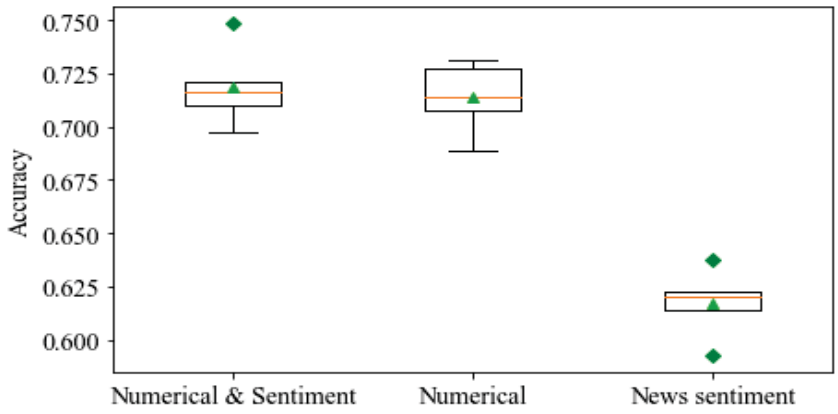

Fig. 4.3. Comparative cross-validated accuracy for the training set. The green arrows show the mean, while the orange line is the median.

To test whether the training accuracy mean for the two datasets (with and without the sentiment feature) is statistically significant the paired t-test is used. Results in Table 4.2 indicate that there is a small positive significant difference of 0.31% in average training accuracy between the model using the augmented dataset and numerical dataset.

Table 4.2. Cross-validated mean accuracy with 1,200 parameter combinations

Cross-validation test fold	Dataset with sentiment feature	Dataset without sentiment feature	Paired t-test, p-value	Mean difference, %
1	0.6901	0.6938	3.17e-09 ***	-0.37%
2	0.6765	0.6733	1.98e-11 ***	0.32%
3	0.7181	0.7016	3.09e-318 ***	1.65%
4	0.6908	0.6911	0.6129	-0.03%
5	0.7164	0.7161	0.6078	0.03%
Average	**0.6983**	**0.6952**	**1.21e-16*** **	**0.31%**

***: significance at 5% level

Fig. 4.4 - 4.6 show the confusion matrix for each of the three cases. The model with the augmented dataset has correctly classified three extra instances as true positives and two extra instances as true negative on the test set, which is a very modest improvement compared to the numerical dataset.

Both models misclassify a large proportion of the down movements. The confusion matrix in Fig. 4.4 shows that the model with augmented dataset classifies 31.4% of down movements as up movements and 21.77% of the up movements as down movements.

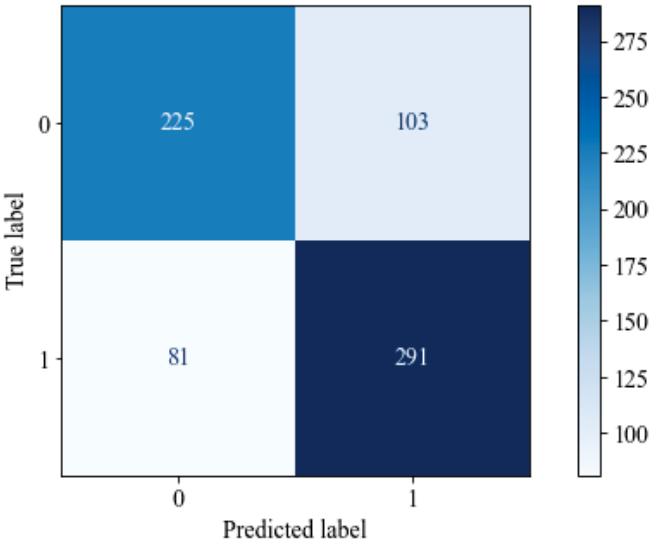

Fig. 4.4. Confusion matrix, numerical & sentiment

The model with just numerical features classifies 32.01% of down movements as up movements and 22.58% of up movements as down movements, as the confusion matrix in Fig. 4.5 indicates.

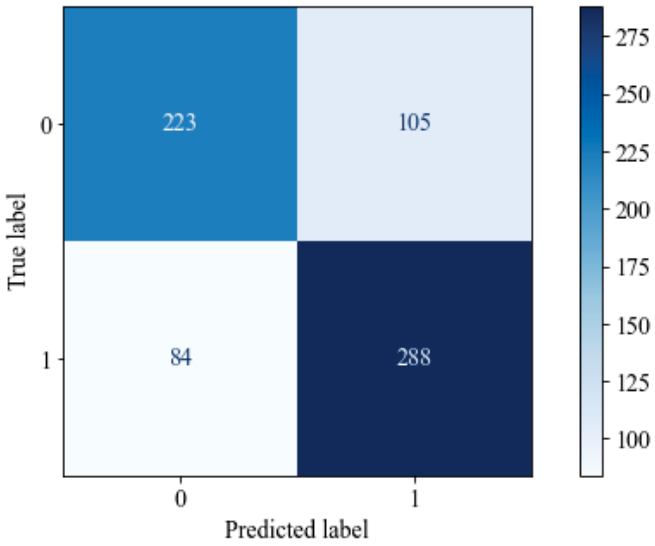

Fig. 4.5. Confusion matrix, numerical

The basic model misclassifies 46.64% of the down movements and 26.88% of the up movements, as displayed in Fig. 4.6.

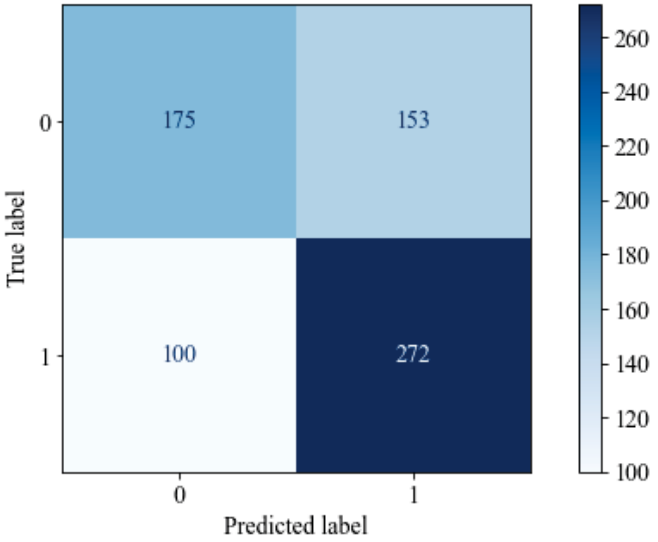

Fig 4.6. Confusion matrix, baseline

Figures 4.7 - 4.9 display the ROC curves and AUC for the two classes (daily up/down movements of OMXS30). Fig. 4.7 shows that the model with the augmented dataset has a good performance, with an AUC of 0.8.

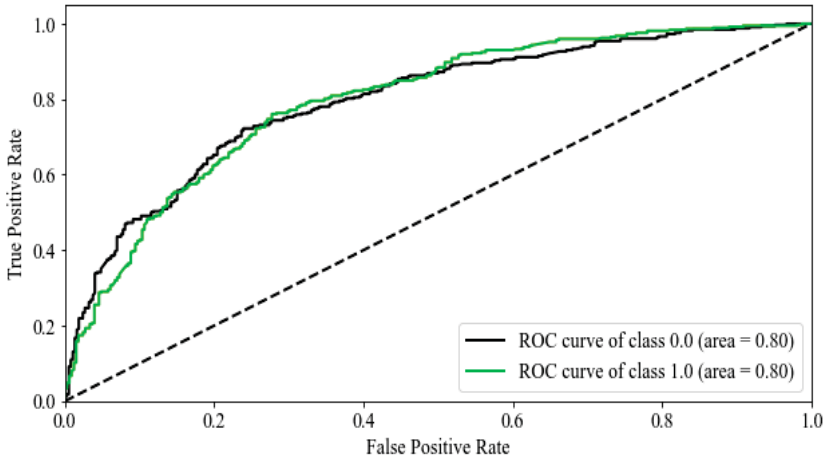

Fig. 4.7. ROC curves, numerical & sentiment

The model with numerical dataset has a slightly lower AUC of 0.79, as shown in Fig. 4.8.

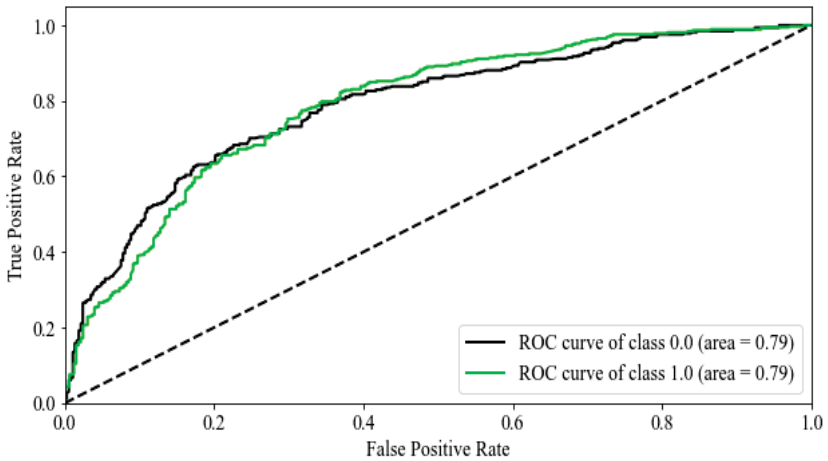

Fig. 4.8. ROC curves, numerical

Figure 4.9 displays the ROC curve and AUC for the baseline model. The classification performance is poor, with an AUC of 0.67.

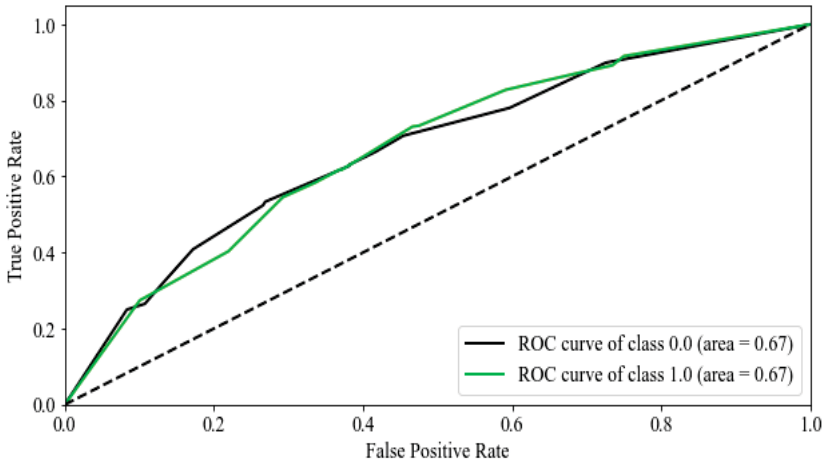

Fig. 4.9. ROC curves, baseline

In Fig. 4.10 MCC is displayed for the three models. The model trained on the augmented dataset achieves a MCC of 0.47, while the simple model achieved a slightly lower MCC of 0.46. This small positive difference is consistent with previous performance metrics results reported in Fig. 4.2 - 4.9.

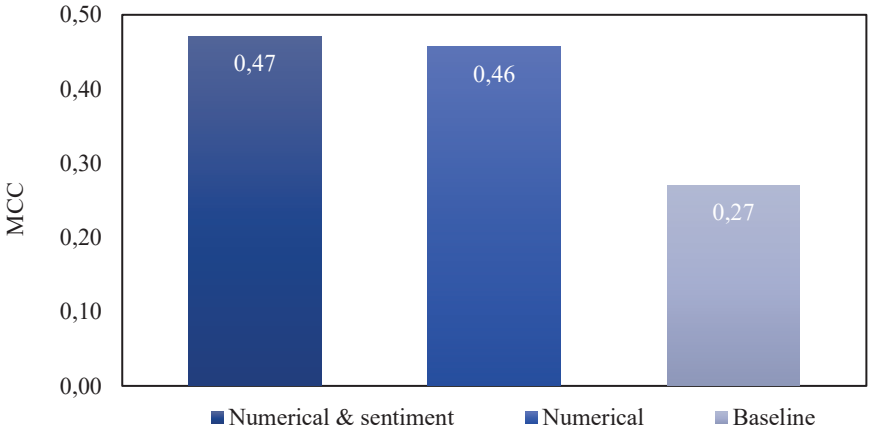

Fig. 4.10. Matthews Correlation Coefficient

5 DISCUSSION

The test data has 372 daily true positive movements and 328 true negative movements for OMXS30. Hence, for the XGBoost model to be viable, it must yield better performance than a baseline classifier that takes the movement direction average. A zero-rule classifier which uses mode to select the majority class of the test set would predict up movements 372/700=53.14% of the time, and down movements 46.86% of the time.

To address the first research question, the results in Fig. 4.2 indicate that the test set accuracy for all three models is higher than for the zero-rule classifier. Namely, the model with the numerical dataset yields an accuracy of 73% on the test set, with 77.42% of the positive movements and 67.99% of the negative movements classified correctly, respectively. These are promising results for XGBoost, which make worthwhile the investigation of economic significance to finance practitioners. The classification performance is according to expectations, as expressed in subsection 3.1.1. The accuracy is higher than what was obtained in [33], where the obtained accuracy was 66%. However, the results are not as optimistic as those obtained in [35], where the accuracy was 82.66%. It should also be noted that a direct comparison of the results with the ones in the literature is not possible, due to the fact that different datasets and algorithm hyperparameters were used.

An interesting result is that the model achieves a better than the zero-rule classifier when only using the sentiment feature, an average accuracy of 63.85% on the test set, with 73.12% of the up movements and 53.35% of the down movements are classified correctly. This is an indicator of the significant power of sentiment data as sole predictor for the trend movement of OMXS30. These results are superior to the ones obtained in [45], where average accuracy for the chosen dataset did not exceed 60% when employing less sophisticated machine learning models such as DT and SVM. This could be attributed to the choice of algorithm, but also to the fact that the sentiment extraction from text in this thesis was done manually, and thus the quality of this data is deemed higher.

Regarding the second research question, the results obtained through the various classification metrics on the test set indicate a modestly higher accuracy for the model with the augmented set, 73.71% test set accuracy versus 73% for the model with historical price features only. The average difference in training accuracy is only slightly higher for the augmented model, of 0.31%. This is consistent with the results obtained for AUC and MCC in Fig.4.4 - 4.7, where small differences of 1.3% - 2.2% between values were obtained. These results may be correlated with the feature importance results displayed in Fig 4.1. It indicated that the sentiment feature is only 6^{th} in importance, therefore adding the sentiment feature plays a smaller role in building the boosted trees.

For the second research question, the results are not in line with the findings in literature where significantly higher differences in performance of at least 10% were obtained when using features extracted from text [42], [43], [44]. This could be attributed to several factors. First of all, the type of information and sentiments conveyed in the text may be better embedded in or overlap to some degree with the price data for the international stock indices used for the US and Asian markets. Indeed, the short texts that served as dataset for analysis provide information about the main global stock markets as well as the expected direction of movement at market open for OMXS30, which may be reflected to some extent in the selected features. As such, this textual data is problem specific, and the results might not be generalizable. Secondly, the time granularity used in this study is daily and the purpose of this thesis was to predict the movement of OMXS30 on a daily basis. The text data gives an indication of the OMXS30 price direction at market opening. However, this does not always coincide with the daily price direction. Trends may often reverse intraday due to macroeconomic releases, index constituents' corporate disclosures that have a significant stock price impact, or other high impact events. In order to achieve higher accuracy, it may be required to use higher time granularity and additional sources of information.

It may be argued that the small difference in classification performance is insufficient for drawing any inferences about the performance of the text augmented model compared to the model using the simple dataset in a practical trading setting. That is because, for comparison purposes, not only classification performance is important, but also the magnitude of the realised returns whose direction was correctly classified by the models. A rigorous back-testing of the two models in a realistic investment environment is, however, beyond the scope of this thesis.

5.1 Limitations and validity threats

A limitation for this study is that only a narrow subspace of the hyperparameter space was explored by grid search. Therefore, there is a possibility that the optimal set was not identified, with potential negative effects on classification performance and the importance of the sentiment feature in the model. Furthermore, due to time constraints, feature selection was not performed thoroughly and a single main algorithm, XGBoost, was chosen based on good performance and popularity in practice. A different set of features and/or algorithms might yield different results to the research questions.

An additional limitation is that for the purpose of this study, a small text dataset of ca 3,500 observations was used. As such, manual sentiment annotation was straightforward. However, this might not be possible in a practical trading setting where the requirement is to perform text analysis in real time for automated decision making on a broad range of assets. Hence, for larger scale projects it would be necessary to employ an automated method for financial sentiment extraction.

6 CONCLUSION AND FUTURE WORK

The purpose of this thesis was to examine the classification performance impact of augmenting the input feature set with sentiment features extracted from news when predicting the daily trend of OMXS30 stock index. Sentiment for relevant financial markets was extracted at sentence level from ca 3,500 daily financial newsletters and aggregated at document level. XGBoost was used as algorithm and four metrics were computed for assessing classification performance of the daily trend direction for OMXS30.

Results indicate that only using the sentiment feature yielded a good classification accuracy of nearly 64%. This proves the predictive importance of sentiment features extracted from text. When using only features derived from historical prices, the classification performance was satisfactory at 73%. Small positive differences were recorded for the model with the sentiment augmented dataset on all performance metrics. On the test set, XGBoost yielded a decent classification performance of 73.71% and was generally better at classifying up movements than down movements.

These results are promising and for future work, an economic evaluation of the model and trading simulation in a realistic investment environment may prove valuable for finance practitioners. In addition, refining the time granularity to intraday would model potential price trend reversals and make the model more responsive.

As stock markets in general are influenced by a multitude of factors both over short and longer time horizons, it would meaningful for further research to include and analyze additional textual sources related to macroeconomic variables well as the index constituents. Features extracted from these sources may further boost predictive performance.

REFERENCES

[1] E. Fama, "The Behavior of Stock-Market Prices", *The Journal of Business*, vol. 38, no. 1, p. 34, 1965.

[2] T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity", *Journal of Econometrics*, vol. 31, no. 3, pp. 307-327, 1986.

[3] R. Engle and C. Granger, "Co-Integration and Error Correction: Representation, Estimation, and Testing", *Econometrica*, vol. 55, no. 2, p. 251, 1987.

[4] S. Makridakis and M. Hibon, "ARMA Models and the Box-Jenkins Methodology", *Journal of Forecasting*, vol. 16, no. 3, pp. 147-163, 1997.

[5] R. Siedlecki and D. Papla, "Logistic Law of Growth as a Base for Method of Forecasting Stock Market Data", *SSRN Electronic Journal*, 2012.

[6] X. Zhong and D. Enke, "Forecasting daily stock market return using dimensionality reduction", *Expert Systems with Applications*, vol. 67, pp. 126-139, 2017.

[7] D. Gandhmal and K. Kumar, "Systematic analysis and review of stock market prediction techniques", *Computer Science Review*, vol. 34, p. 100190, 2019.

[8] Bustos, O. and Pomares-Quimbaya, A., 2020. "Stock market movement forecast: A Systematic review". *Expert Systems with Applications*, 156, p.113464.

[9] C. Krauss, X. Do and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500", *European Journal of Operational Research*, vol. 259, no. 2, pp. 689-702, 2017.

[10] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System", *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[11] D. Enke and S. Thawornwong, "The use of data mining and neural networks for forecasting stock market returns", *Expert Systems with Applications*, vol. 29, no. 4, pp. 927-940, 2005.

[12] X. Li, P. Wu and W. Wang, "Incorporating stock prices and news sentiments for stock market prediction: A case of Hong Kong", *Information Processing & Management*, vol. 57, no. 5, p. 102212, 2020.

[13] A. Khadjeh Nassirtoussi, S. Aghabozorgi, T. Ying Wah and D. Ngo, "Text mining for market prediction: A systematic review", *Expert Systems with Applications*, vol. 41, no. 16, pp. 7653-7670, 2014.

[14] W. Khan, M. Ghazanfar, M. Azam, A. Karami, K. Alyoubi and A. Alfakeeh, "Stock market prediction using machine learning classifiers and social media, news", *Journal of Ambient Intelligence and Humanized Computing*, 2020.

[15] "Market Index - Overview, Functions, and Examples", *Corporate Finance Institute*, 2020. [Online]. Available: https://corporatefinanceinstitute.com/resources/knowledge/trading-investing/market-index/. [Accessed: 4-Nov-2020].

[16] J. Chang, *Choice of market proxy in the capital asset pricing model*. Lap Lambert Academic Publ, 2011.

[17] B. Malkiel, *A random walk down Wall Street*. New York: Norton, 1973.

[18] E. Fama, "Efficient Capital Markets: A Review of Theory and Empirical Work", *The Journal of Finance*, vol. 25, no. 2, p. 383, 1970.

[19] Man, X., Luo, T., & Lin, J. (2019). Financial Sentiment Analysis(FSA): A Survey. *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, pp.617-622.

[20] J. Murphy, *Technical analysis of the financial markets*. Fishkill, N.Y.: New York Institute of Finance, 1999.

[21] B. Liu, *Sentiment analysis and opinion mining*. [S.l.]: Morgan & Claypool, 2012.

[22] J. Bollen, H. Mao and X. Zeng, "Twitter mood predicts the stock market", *Journal of Computational Science*, vol. 2, no. 1, pp. 1-8, 2011.

[23] H. Leung and T. Ton, "The impact of internet stock message boards on cross-sectional returns of small-capitalization stocks", *Journal of Banking & Finance*, vol. 55, pp. 37-55, 2015.

[24] F. Alzazah and X. Cheng, "Recent Advances in Stock Market Prediction Using Text Mining: A Survey", in *E-Business*, R. Wu and M. Mircea, Ed. 2020.

[25] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, 2019.

[26] L. Breiman, *Classification and regression trees*. The Wadsworth and Brooks-Cole statistics probability series. Chapman & Hall,1984.

[27] R. Nisbet, J. Elder and G. Miner, *Handbook of statistical analysis and data mining applications*. Amsterdam: Elsevier/Academic Press, 2009.

[28] J. Friedman, "Greedy function approximation: a gradient boosting machine", *The Annals of Statistics*, vol. 29, no. 5, pp. 1189-1232, 2001.

[29] J. Brownlee, "A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning", *Machine Learning Mastery*, 2020. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/. [Accessed: 5-Nov-2020].

[30] "Introduction to Boosted Trees — xgboost 1.4.0-SNAPSHOT documentation", *Xgboost.readthedocs.io*, 2020. [Online]. Available: https://xgboost.readthedocs.io/en/latest/tutorials/model.html. [Accessed: 6-Nov-2020].

[31] J. Friedman, "Stochastic gradient boosting", *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367-378, 2002.

[32] S. Shen, H. Jiang and T. Zhang, *Cs229.stanford.edu*, 2012. [Online]. Available: http://cs229.stanford.edu/proj2012/ShenJiangZhang-StockMarketForecastingusingMachineLearningAlgorithms.pdf. [Accessed: 21- Oct- 2020].

[33] D. Lv, S. Yuan, M. Li and Y. Xiang, "An Empirical Study of Machine Learning Algorithms for Stock Daily Trading Strategy", *Mathematical Problems in Engineering*, vol. 2019, pp. 1-30, 2019.

[34] V. Vismayaa, K. Pooja, A. Alekhya, C. Malavika, B. Nair and P. Kumar, "Classifier Based Stock Trading Recommender Systems for Indian stocks: An Empirical Evaluation", *Computational Economics*, vol. 55, no. 3, pp. 901-923, 2019.

[35] E. Ampomah, Z. Qin and G. Nyame, "Evaluation of Tree-Based Ensemble Machine Learning Models in Predicting Stock Price Direction of Movement", *Information*, vol. 11, no. 6, p. 332, 2020.

[36] M. Ballings, D. Van den Poel, N. Hespeels and R. Gryp, "Evaluating multiple classifiers for stock price direction prediction", *Expert Systems with Applications*, vol. 42, no. 20, pp. 7046-7056, 2015.

[37] J. Patel, S. Shah, P. Thakkar and K. Kotecha, "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques", *Expert Systems with Applications*, vol. 42, no. 1, pp. 259-268, 2015.

[38] L. Huang, “Machine Learning on Stock Price Movement Forecast: The Sample of the Taiwan Stock Exchange”, *International Journal of Economics and Financial Issues*, 2019, 9(2), pp.189-201.

[39] P. Chakraborty, U. S. Pria, M. R. A. H. Rony and M. A. Majumdar, "Predicting stock movement using sentiment analysis of Twitter feed," *2017 6th International Conference on Informatics, Electronics and Vision & 2017 7th International Symposium in Computational Medical and Health Technology (ICIEV-ISCMHT)*, Himeji, 2017, pp. 1-6.

[40] T. Geva and J. Zahavi, "Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news", *Decision Support Systems*, vol. 57, pp. 212-223, 2014.

[41] Abdul-Rahman S., Mutalib S. ,"Mining Textual Terms for Stock Market Prediction Analysis Using Financial News". In: Mohamed A., Berry M., Yap B. (eds) Soft Computing in Data Science. SCDS 2017. *Communications in Computer and Information Science*, vol 788. Springer, Singapore, 2017.

[42] A. A. Bhat and S. S. Kamath, "Automated stock price prediction and trading framework for Nifty intraday trading," *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, 2013, pp. 1-6.

[43] T. -. Teoh et al., "From Technical Analysis to Text Analytics: Stock and Index Prediction with GRU", *IEEE Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM),* Bangkok, Thailand, 2019, pp. 496-500.

[44] S. Bouktif, A. Fiaz and M. Awad, "Stock Market Movement Prediction using Disparate Text Features with Machine Learning", in *Third International Conference on Intelligent Computing in Data Sciences (ICDS)*, Marrakesh, 2019.

[45] H. Elbousty and K. Salah-ddine, "Financial News Analysis for Moroccan Stock Trend Predictions". *Test Engineering and Management*. vol.82. pp. 1712 -1717, 2020.

[46] n.d. *Python*. https://www.python.org/: Python Software Foundation.

[47] n.d. *Jupyter Notebook*. https://jupyter.org/

[48] C. Harris et al., "Array programming with NumPy", *Nature*, vol. 585, no. 7825, pp. 357-362, 2020.

[49] P. Virtanen et al., "SciPy 1.0: fundamental algorithms for scientific computing in Python", *Nature Methods*, vol. 17, no. 3, pp. 261-272, 2020.

[50] n.d. *Scrapy*. https://scrapy.org/

[51] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, no. 12, pp. 2825-2830, 2011.

[52] W. McKinney, "Data Structures for Statistical Computing in Python", in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 56-61.

[53] n.d. *Pandas TA*. https://github.com/twopirllc/pandas-ta

[54] J. Hunter, "Matplotlib: A 2D Graphics Environment", *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90-95, 2007.

[55] *mwaskom/seaborn*. https://doi.org/10.5281/zenodo.592845: Zenodo, 2020.

[56] C. Bergmeir and J. Benítez, "On the use of cross-validation for time series predictor evaluation", *Information Sciences*, vol. 191, pp. 192-213, 2012.

[57] J. Brownlee, "Feature Importance and Feature Selection With XGBoost in Python", *Machine Learning Mastery*, 2020. [Online]. Available: https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/. [Accessed: 02-Dec-2020].

APPENDIX

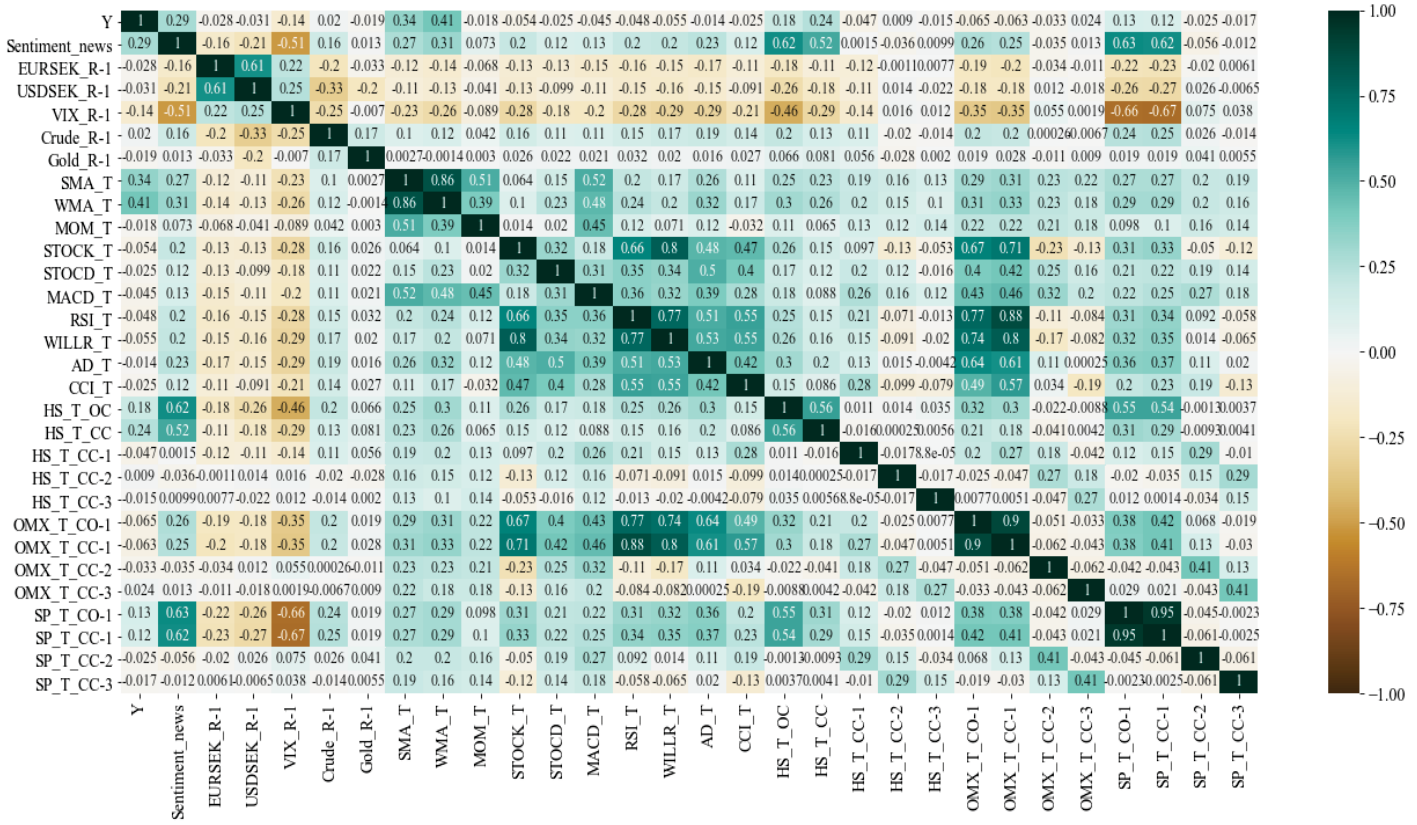

Fig. A.1. Correlation matrix for input set used in feature selection

Table A.1. Features annotation and description

Feature	Description	Selected
Y	Up/down movement of OMXS30	✓ (target variable)
Sentiment_news	Sentiment feature extracted from news documents	✓
EURSEK_R-1	Previous day return for EURSEK currency pair	
USDSEK_R-1	Previous day return of EURSEK	
VIX_R-1	Previous day return of VIX	✓
Crude_R-1	Previous day return of WTI oil futures	
Gold_R-1	Previous day return of Gold futures	
SMA_T	Simple moving average, trend deterministic	✓
WMA_T	Weighted moving average, trend deterministic	✓
STOCK_T	Stochastic K%, trend deterministic	✓
STOCKD_T	Stochastic D%, trend deterministic	
MACD_T	Moving Average Convergence Divergence, trend deterministic	
RSI_T	Relative Strength Index, trend deterministic	✓
WILLR_T	Williams R%, trend deterministic	✓
AD_T	Accumulation Distribution, trend deterministic	
CCI_T	Commodity Channel Index, trend deterministic	
HS_T_OC	Hang Seng stock index overnight return, discretized	
HS_T_CC	Hang Seng stock index daily return, discretized	✓
HS_T_CC-1	Hang Seng stock index daily return, lag 1 day, discretized	
HS_T_CC-2	Hang Seng stock index daily return, lag 2 days, discretized	
HS_T_CC-3	Hang Seng stock index current day return, lag 3 days, discretized	
OMX_T_CO-1	OMXS30 overnight return, discretized	✓
OMX_T_CC-1	OMXS30 current day return, lag 1 day, discretized	
OMX_T_CC-2	OMXS30 current day return, lag 2 days, discretized	
OMX_T_CC-3	OMXS30 current day return, lag 3 days, discretized	
SP_T_CO-1	S&P500 intraday return, lag 1 day, discretized	✓
SP_T_CC-1	S&P500 daily return, lag 1 day, discretized	
SP_T_CC-2	S&P500 daily return, lag 2 days, discretized	
SP_T_CC-3	S&P500 daily return, lag 3 days, discretized	

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden