Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Machine Learning for Stock Index Predictions

Bachelor Thesis

2024 **Matej Šembera**

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Machine Learning for Stock Index Predictions

Bachelor Thesis

Study Program:	Mathematics of Economy and Finance
Field of Study:	Mathematics
Department:	Department of Applied Mathematics and Statistics
Thesis Supervisor:	Mgr. Samuel Rosa, PhD

Bratislava, 2024 **Matej Šembera**

Abstrakt

Šembera, Matej: Predikovanie akciového indexu pomocou strojového učenia [Bakalárska práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: Mgr. Samuel Rosa, PhD., Bratislava, 2024, 47 s.

Predkladaná bakalárska práca sa zameriava na vytvorenie modelu využívajúceho strojové učenie na predpovedanie denných pohybov indexu S&P 500. Popisuje rôzne algoritmy strojového učenia, medzi ktoré patria k-najbližších susedov, náhodné lesy, logistická regresia a metóda podporných vektorov. Tiež je používaná aj neuronová sieť dlhá-krátka pamäť. Zaoberá sa aj princípmi výberu premenných pre model. Popisuje a zdôvodňuje výber premenných, pričom premenné sú rozdelené do tematických skupín. V rámci jednotlivých skupín porovnáva výsledky s inými akademickými prácami v oblasti strojového učenia. V závere sa vyhodnocuje kvalita modelu pomocou metriky vytvorenej autorom a porovnáva sa výkon modelu s výkonom indexu S&P 500.

Kľúčové slová: strojové učenie, S&P 500, predikcia, logistická regresia

Abstract

Šembera, Matej: Machine learning for stock index predictions [Bachelor thesis], Comenius University in Bratislava, Faculty of mathematics, physics and informatics, Department of applied mathematics and statistics; Supervisor: Mgr. Samuel Rosa PhD., Bratislava, 2024, 47 p.

The presented bachelor's thesis focuses on creating a model using machine learning to predict daily movements of the S&P 500 index. It describes the machine learning algorithms employed, such as K-Nearest Neighbors, Random Forest, Support Vector Machines, and Logistic Regression. Additionally, the neural network Long Short-Term Memory is also used. It addresses the principles of feature selection for the model and explains and justifies the choice of features, categorizing them into thematic groups. Within each group, it compares the results with other academic papers in the field of machine learning. In conclusion, the thesis evaluates the quality of the model using a metric created by the author and compares its performance with the S&P 500 index.

Keywords: Machine learning, S&P 500, predictions, Logistic Regression

Contents

Introduction

Mathematics has often been used to create new inventions. One such invention that rose dramatically in importance in the twenty-first century was machine learning. One of the main advantages that contributed to the rise of machine learning was the ability to learn patterns by itself, rather than relying on rules written by humans. This process of self-learning is much more time-effective and can also lead to the discovery of patterns that are invisible to the human eye.

For this reason, today, machine learning is used in solving problems across various domains, including medicine, astronomy, and biology. For instance, machine learning is used to process data from telescopes in order to discover new exoplanets.

Machine learning is not only used in the field of research but also in the private sector. One of the sectors that has shown a heavy interest in machine learning is finance. Today, more and more companies are trying to use machine learning to predict the stock market.

In this thesis, we aim to develop a model that will predict daily moves of the stock index S&P 500. Numerous research studies have also tried to predict daily moves of the S&P 500 index [12, 16, 17]. However, some of the studies are more than a decade old [17]. Some studies have attempted to predict weekly movements of the S&P 500 index [12]. Therefore, we aim to employ similar features to gain deeper insights.

The most crucial aspect of building a model that predicts the stock market is the selection of features, the importance of which is outlined in [14]. We will be using two methodologies, A and B, to select the right features, but we will also rely on common sense.

Feature selection holds significant importance, as does the careful selection of appropriate algorithms. Therefore, we will employ a range of machine learning algorithms, including K-Nearest Neighbors, Random Forest, Logistic Regression and Support Vector Machines. We explain each algorithm with reference to [4] as our primary source. Additionally, we will be using a neural network known as Long Short-Term Memory.

When explaining each algorithm, we will use knowledge from statistics and optimization since many of the algorithms we will be using are based on solving particular optimization problem. We will also be using knowledge from economics when selecting possible features for our model. We will compare the performance of these algorithms throughout the bachelor's thesis. For this reason, we will create our own metric since we did not find a metric that satisfied our needs.

1 Theoretical foundations

This section aims to explain the concepts used throughout this bachelor thesis.

1.1 Predictability of the S&P 500 index

The Standard & Poor's 500 index, abbreviated as the S&P 500, is an index that includes the 500 publicly listed companies in the U.S. with the highest capitalization. In this bachelor's thesis, we selected the S&P 500 index for machine learning forecasting due to its stability and its widespread recognition as a metric for evaluating the performance of the U.S. stock market.

Random walk theory suggests that changes in asset prices are random, making accurate predictions impossible. The idea comes from the efficient market hypothesis, which says that stock prices quickly adjust to reflect all available information, so acting on new information to gain profits is not possible.

On the other hand, many successful traders argue that markets can be predicted. Several investors and traders have successfully outperformed the S&P 500 index. For instance, Warren Buffett is recognized by many as an investor who consistently outperforms the S&P 500 index. Therefore, it is wise to explore the extent to which the stock market can be predicted.

1.2 Terminologies used throughout this bachelor's thesis

In the context of machine learning, and throughout this bachelor's thesis, we will use the following terminology:

- **Data point:** In our context, a data point refers to one trading day, represented by a row in a table for the respective day. The symbol used throughout this thesis to denote the ith data point is x_i. New data point is represented as x_{i+1}. The quantity of data points is represented by the variable m.

- **Feature:** A feature refers to a specific attribute represented by a value, used to predict the movement of the next day. In our table, each column contains values of a specific feature.

- **X_{ij}:** Represents the jth feature of the ith data point. The row of the ith data point is denoted as $X_i = [X_{i1}, X_{i2}, ..., X_{in}]$, where n represents number of features. The symbol j is consistently used to reference a feature.

- **Class:** All data points will be divided into two groups or classes. The first class, denoted by the number zero (0), refers to data points where the index declined on that day. The second class, denoted by the number one (1), refers to data points where the index rose on that day. The predicted class for the ith data point is denoted as y_i, where y_i can take values of either 1 or 0.

- **Hypothesis:** A hypothesis is a function that approximates the dependency of a predicted output on features [5].
- **Algorithm:** An algorithm is a method used to construct a hypothesis predicting the probability of a data point belonging to class one. The probability of an $(i+1)$th data point belonging to class 1 is represented as $p(x_{i+1})$.
- **Model:** Refers to a combination of a specific algorithm and features.

1.3 The principles of machine learning

Machine learning is an umbrella term for algorithms that can learn patterns from historical data to solve specific problems, without being explicitly programmed to perform the task by humans. However, it does come with challenges; notably, these algorithms can sometimes learn irrelevant patterns. For example, let us assume there is a significant negative correlation between the number of fresh lemons imported from Mexico to the USA and the total US highway fatality rate. Using common sense and reasoning, humans can understand that this correlation is likely coincidental. However, machine learning algorithms might view these as some numbers, potentially mistaking the correlation for causation. This could lead to a situation where the model tries to predict the US highway fatality rate based on lemon imports. Two of the most known challenges in machine learning are overfitting and underfitting.

1.4 Overfitting and underfitting

Before discussing overfitting and underfitting, it is necessary to first define the term accuracy. Accuracy is a metric commonly used to measure the performance of machine learning models. It is calculated as Accuracy $= \frac{\text{total number of correct predictions}}{\text{total number of predictions}} \times 100$.

Overfitting is an undesirable outcome in machine learning when a model cannot generalize and fits too closely to the training data. Training data refers to data used to train our model. A primary indicator of overfitting is when a model accurately predicts outcomes on the training data but performs unsatisfactorily on new, unseen data, referred to as test data. Test data refers to data used to evaluate the performance of our model. Mathematically, overfitting is defined as follows [10]: $\hat{h} \in H$ overfits the training data D_{tr} if there exists $h_0 \in H$ such that $\text{accuracy}_{D_{\text{tr}}}(\hat{h}) > \text{accuracy}_{D_{\text{tr}}}(h_0)$ and $\text{accuracy}_{D_{\text{te}}}(\hat{h}) < \text{accuracy}_{D_{\text{te}}}(h_0)$,

where

- H is the hypothesis space. In this context, the hypothesis space refers to the entire set of all possible hypotheses that can be considered,

- $\hat{h}$ is the learnt hypothesis that our model learnt on the training data,
- h_0 is another hypothesis in H,
- D_{tr} represents the training data,
- $\text{accuracy}_{D_{\text{tr}}}(\hat{h})$ and $\text{accuracy}_{D_{\text{tr}}}(h_0)$ are the accuracies of the hypotheses $\hat{h}$ and h_0 respectively, on the training data D_{tr},
- D_{te} represents the test data,
- $\text{accuracy}_{D_{\text{te}}}(\hat{h})$ and $\text{accuracy}_{D_{\text{te}}}(h_0)$ are the accuracies of the hypotheses $\hat{h}$ and h_0 respectively, on the testing data.

On the other side underfitting occurs when the machine learning model cannot capture the pattern in the training data. The sign of underfitting is mainly low performance on both training and testing data.

Similarly, underfitting can be defined as follows [10]: $\hat{h} \in H$ underfits the training data D_{tr} if there exists $h_0 \in H$ such that $\text{accuracy}_{D_{\text{tr}}}(\hat{h}) < \text{accuracy}_{D_{\text{tr}}}(h_0)$ and $\text{accuracy}_{D_{\text{te}}}(\hat{h}) < \text{accuracy}_{D_{\text{te}}}(h_0)$.

1.5 Machine learning algorithms

In this bachelor thesis we will use several machine learning algorithms. In this section we will explain each algorithm. It is worth noting that the primary source of information for all algorithms, with the exception of LSTM, is [4]. Source for LSTM is [7].

1.5.1 K-Nearest Neighbors

The K-Nearest Neighbors algorithm, abbreviated as KNN, determines the label of a new data point based on the classifications of its nearest neighbors. The nearest data points are referred to as neighbors. The number of neighbors to consider is specified by the hyperparameter k. For instance, if $k = 3$, the algorithm considers the three nearest neighbors of the new data point. To determine the nearest neighbors, the Euclidean distance is employed to calculate the distance between two data points. The Euclidean distance between data points x_1 and x_2 is calculated as $\sqrt{\sum_{j=1}^{n}(x_{1j} - x_{2j})^2}$.

Upon identifying the nearest neighbors, KNN calculates the probability that a new data point belongs to class 1. The conditional probability that a new data point x_{i+1} belongs to a class 1 is calculated as $P(y_{i+1} = 1|X = x_{i+1}) = \frac{1}{k}\sum_{i \in N_0} y_i$, where N_0 represents the set of k nearest neighbors to x_{i+1}.

Visualization of KNN

Figure 1 visualizes the KNN algorithm, considering two features. The feature on the x-axis represents the percentage change in the value of S&P 500 index one day prior, referred to as $D1B$. The feature on the y-axis represents the change in the value of S&P 500 index two days prior, referred to as $D2B$. More detailed explanation of features $D1B$ and $D2B$ is provided in Subsection 2.5.1. The visualization is created using six neighbors.

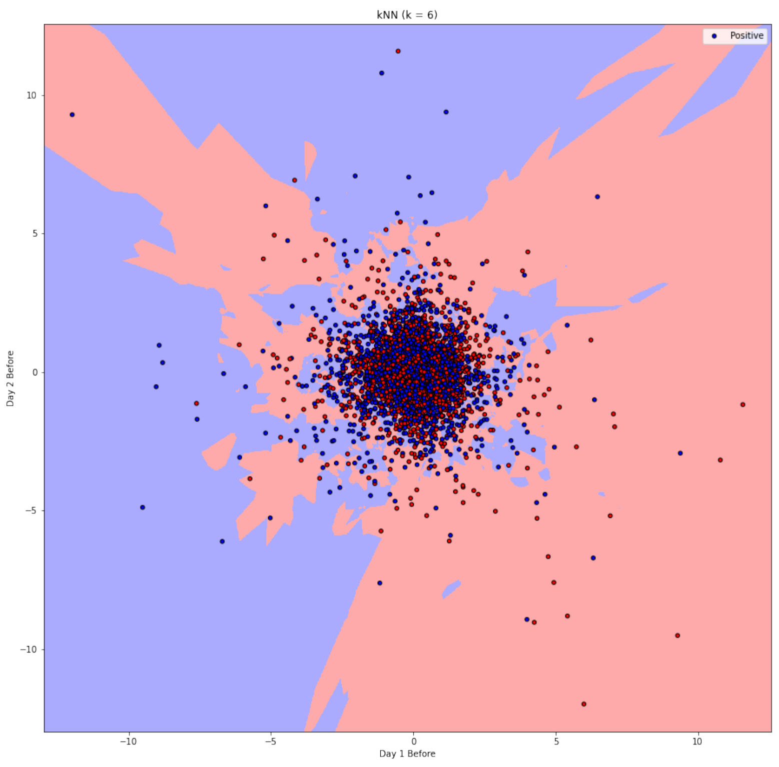

Figure 1: Visualization of the KNN algorithm on the validation set. We used 6 neighbors for creating this visualization, and the threshold was set to 0.5. The definition of a validation set is provided in Subsection 2.1, and the definition of a threshold is in Section 2.

The colors in Figure 1 have the following meanings:

- **Blue dots**: Data points belonging to class zero.
- **Red dots**: Data points belonging to class one.
- **Light Red**: Represents an area in which, if a new data point is assigned, its predicted class will be one.
- **Light Blue**: Represents an area in which, if a new data point is assigned, its predicted class will be zero.

In Figure 2, the KNN algorithm is visualized with a higher number of neighbors, specifically 15. The light blue and light red areas appear smoother, suggesting a more gradual transition.

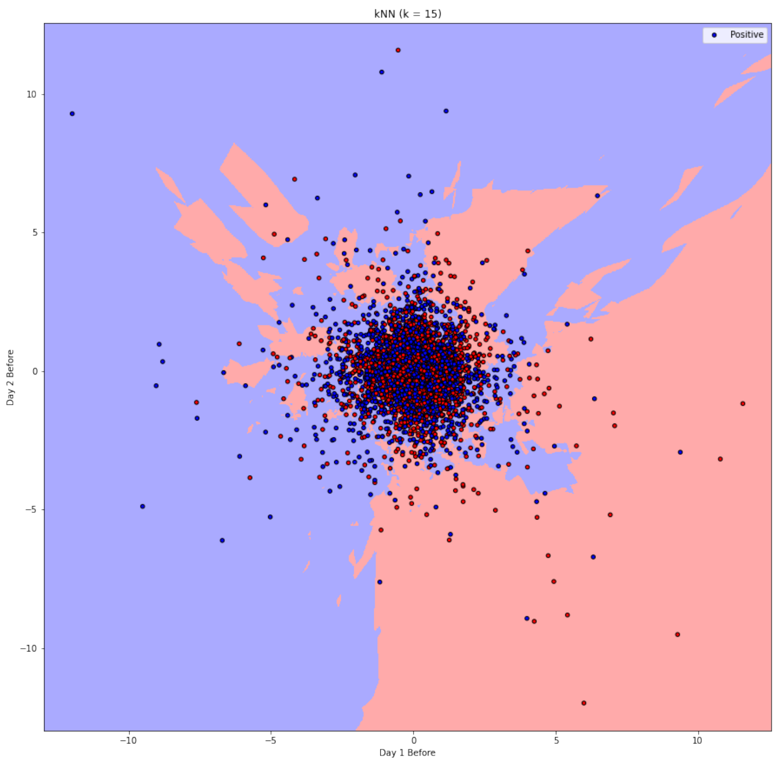

Figure 2: Visualization of the KNN algorithm on the validation set. We used 15 neighbors for creating this visualization, and the threshold was set to 0.5.

Most data points are concentrated towards the center, indicating that, on the majority of the days, the index experienced minimal percentage change.

1.5.2 Random Forest Algorithm

The Random Forest algorithm, abbreviated as RF, is a type of decision tree algorithm. It constructs multiple decision trees. Each decision tree creates its own prediction of the class to which a new data point belongs. The RF algorithm determines the class of a new data point by taking the mode of the classes produced by individual decision trees. The term 'number of estimators' refers to the number of decision trees in the RF algorithm.

Decision trees are constructed with nodes, each representing a point where the tree divides. At each node in each decision tree, RF considers a random subset of features. Specifically, at each node, a decision tree splits the predictor space into two regions, $Region_i(j,s) = \{X \mid X_j < s\}$ and $Region_{i+1}(j,s) = \{X \mid X_j \geq s\}$, where s represents a parameter. At each node the algorithm identifies combinations of s and j that minimize the Gini index or Entropy.

The Gini index is defined as $G = \sum_{c=1}^{C} \hat{p}_{ic}(1-\hat{p}_{ic})$, where C denotes the number of classes (in our case, $C = 2$) and $\hat{p}_{ic}$ denotes the proportion of training observations in $Region_i$ assigned to the cth class. Entropy is calculated as $E = -\sum_{c=1}^{C} \hat{p}_{ic} \log \hat{p}_{ic}$.

As a result, the RF algorithm splits the predictor space into multiple non-overlapping subspaces, denoted from R_1 to R_r. The maximal depth of a decision tree refers to the longest path from the tree's origin to a subspace. For example, the maximum depth of the decision tree in the bottom-left corner of Figure 4 is 3. Additionally Figure 4 provides a visual representation of the RF algorithm, using the same data that was used to visualize the KNN.

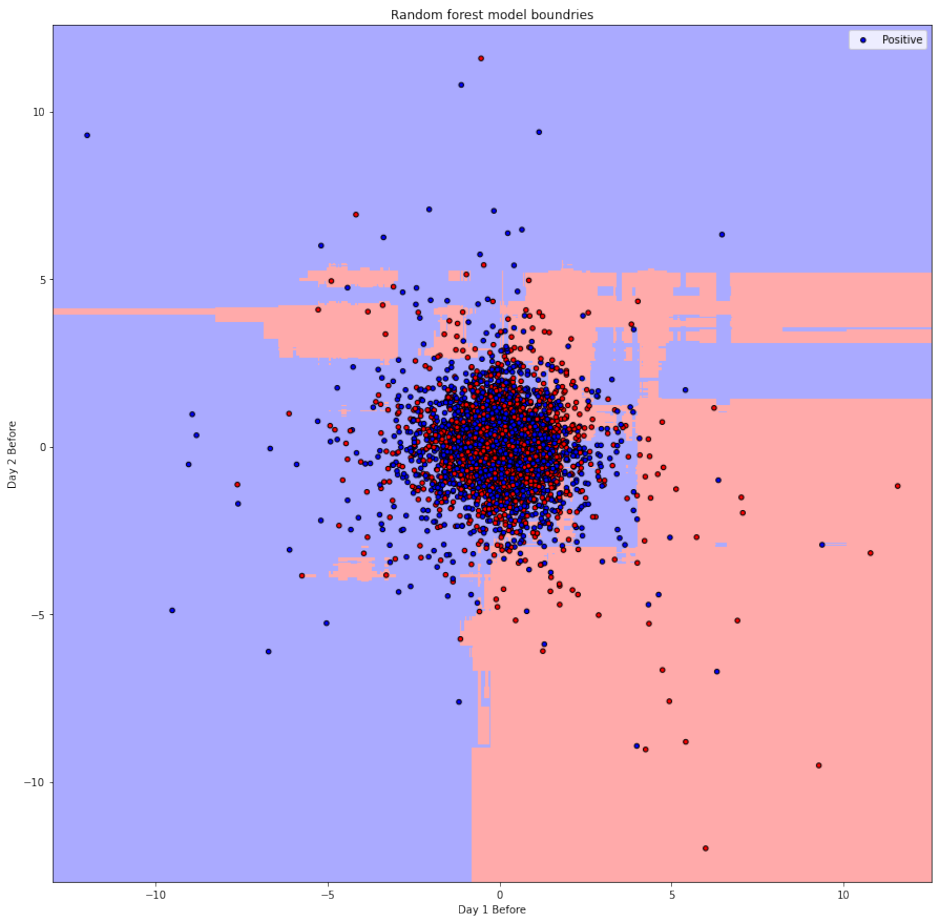

Figure 3: Visualization of the RF algorithm on the validation set. The threshold was set to 0.5.

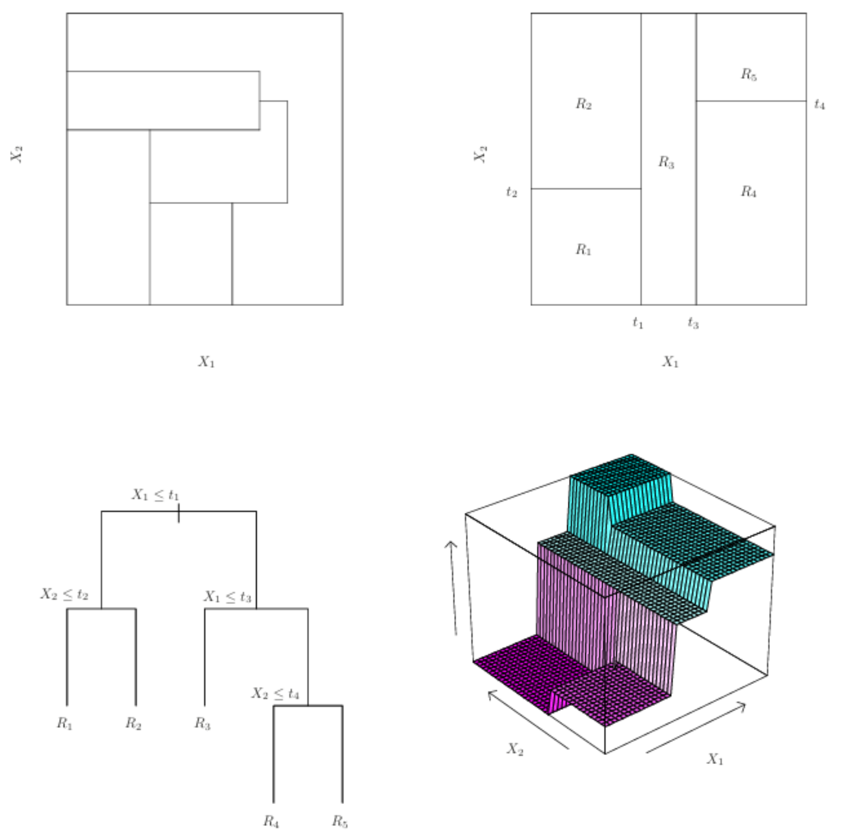

Figure 4: Visualization of decision trees and subspaces. Source: [4]

1.5.3 Support Vector Classifier

A hyperplane is a geometric term used to describe a subspace that has one fewer dimension than the surrounding space. Hyperplane can be mathematically defined as $\beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \beta_0 = 0$. Here, $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients of the respective features, and β_0 is the intercept. The inequalities that determine the subspaces on either side of the hyperplane can be represented as $\beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \beta_0 < 0$ and $\beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \beta_0 > 0$.

The concept of a hyperplane is used in Maximal Margin Classifier, a machine learning algorithm that uses hyperplanes as decision boundaries to classify data points. Figure 5 illustrates how data above the hyperplane are classified as red, and data below the hyperplane are classified as blue.

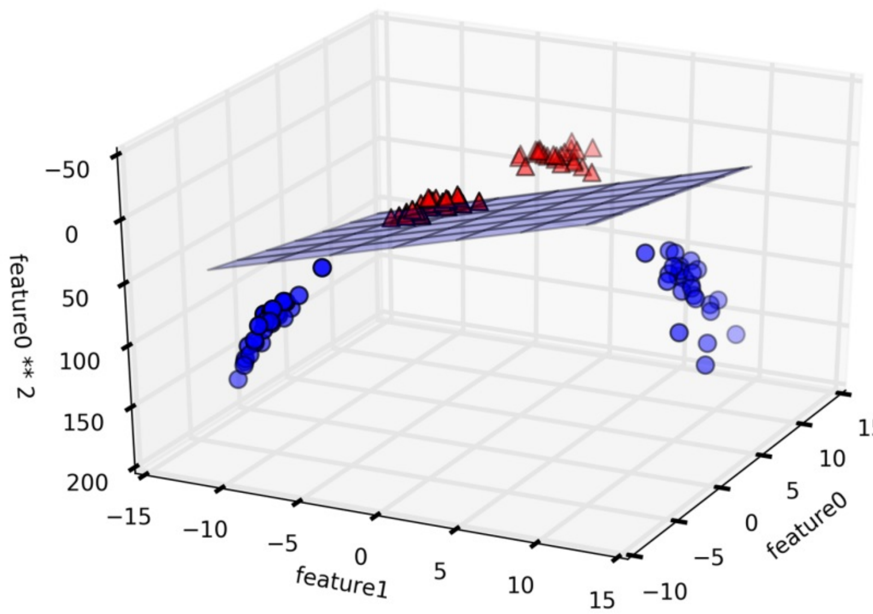

Figure 5: Visualization of Maximal Margin Classifier. Source: [11]

Maximal Margin Classifier aims to identify a hyperplane that can separate data without error, referred to as a hard margin. The term margin denotes the distance from the hyperplane to the nearest classified data point. However, challenges arise when the data is noisy and cannot be cleanly separated by a hyperplane. In contrast, the Support Vector Classifier (SVC), or Soft Margin Classifier, is more flexible.

While the Maximal Margin Classifier tries to put all predictions not only on the correct side of the hyperplane but also outside the margin, the Soft Margin Classifier allows certain points to be misclassified. It allows certain points to lie on the wrong side of hyperplane or margin.

In Figure 6, the dashed lines refers to the margin. The numbers represent individual data points, with blue and red numbers representing different classes. Observing the image on the left, we note data points labeled 5, 2, and 9 are positioned along the margin's boundary, while data points labeled 1 and 8 are located within the margin itself, indicating not an ideal classification.

Redirecting our focus to the image on the right, points 12 and 11 are not only found on the incorrect side of the margin but are also situated on the wrong side of the hyperplane, demonstrating instances where the classifier's predictions are incorrect. In this graphical representation, the solid line refers to the hyperplane. Points 3, 4, 5, 6, and 10 are correctly positioned with respect to the margin, showing accurate classifications.

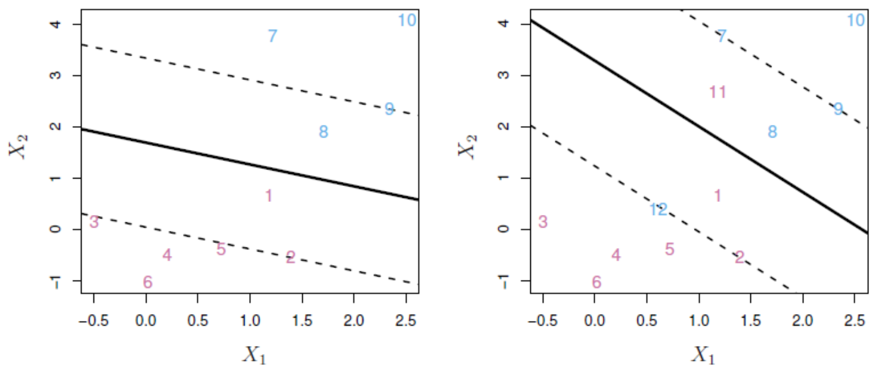

Figure 6: Visualization of SVC. Source: [4]

Mathematics behind SVC

In SVC, the selection of the hyperplane is solution of the following optimization problem:

$$\max_{\substack{\beta_0,\beta_1,\ldots,\beta_n,\\ \varepsilon_1,\ldots,\varepsilon_m,M}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{n} \beta_j^2 = 1, \tag{1}$$

$$y_i\left(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n\right) \geq M(1-\varepsilon_i) \quad \forall i, \tag{2}$$

$$\varepsilon_i \geq 0 \quad \forall i, \tag{3}$$

$$\sum_{i=1}^{m} \varepsilon_i \leq C. \tag{4}$$

In this optimization problem, M represents the margin of the hyperplane. The objective is to maximize this margin to construct a model that generalizes well. The variable ε_i, known as the slack variable, indicates the position of the ith data point.The slack variable, can take on the following values:

- $\varepsilon_i = 0$ if the ith data point is correctly classified, lying on the correct side of the margin.
- $\varepsilon_i > 0$ if the ith data point is misclassified, located within the margin.
- $\varepsilon_i = 1$ if the ith data point is on the wrong side of the hyperplane.

Equation (1) ensures the normalization of the coefficients, β. This standardization enables a consistent comparison of models across various scenarios. Equation (2) ensures the correct classification of each data point with y_i representing its class, denoted by either one or minus one. In the preceding subsections, we assumed that y_i could only take values of 0 or 1. However, in SVC, y_i takes on values of -1 or 1. The term $M(1-\varepsilon_i)$ adjusts the margin. The parameter C in Equation (4) determines the tolerance level for misclassifications. A C value of 0 signifies no tolerance for misclassifications.

Although using SVC in financial market predictions might seem unusual, their inclusion was motivated by the positive outcomes reported in study [13], where Support Vector Machines (SVMs) were used to predict stock market moves. Note that an SVC is a specific type of SVM.

1.5.4 Logistic Regression

Logistic Regression is a machine learning algorithm that uses the logistic function to calculate probabilities that a data point belongs to class one. The logistic function is defined as $p(\mathbf{x_{i+1}}) = f(\mathbf{X}) = \frac{e^{\beta_0+\beta_1 x_{i+1,1}+\cdots+\beta_n x_{i+1,n}}}{1+e^{\beta_0+\beta_1 x_{i+1,1}+\cdots+\beta_n x_{i+1,n}}}$.

The coefficients are calculated via the Maximum Likelihood Estimation (MLE) method. In essence, MLE tries to maximize the likelihood that the model correctly classifies the classes, so that the predicted probabilities align closely with the actual class labels—as close as possible to zero for the class 0 and as close as possible to one for the class 1. We want to find such coefficients that maximize function $\prod_{i:y_i=1} p(x_{i+1}) \prod_{i:y_i=0}(1 - p(x_{i+1}))$.

1.5.5 Long Short-Term Memory

Artificial Neural Networks (ANN) are computational models that are a subset of machine learning, inspired by the structure of neural networks present in the human brain. ANNs consist of interconnected nodes, or artificial neurons, organized into layers. The network of layers is then used to recognize patterns.

One of the many algorithms within ANN is the Long Short-Term Memory (LSTM), known for its superior capability to understand long-term dependencies. The LSTM algorithm's structure is complicated, and this thesis provides a simplified explanation. A more comprehensive understanding of LSTM is available in [15]. The LSTM algorithm has several components, including the hidden state, cell state, forget gate, input gate, and output gate. Each component is discussed in [7]. We will explain each component using several distinct approaches.

- **In Simplified Terms:** Each component will be described in terms that are straightforward and easy to understand.
- **Illustrative Example:** An example relevant to the context of this bachelor thesis will be provided for each component.

Components

Hidden state

- The hidden state can be seen as the short-term memory of the LSTM, remembering recent data.

- In our context, it involves remembering information of the recent trading days.

Cell state

- Cell state serves as long term memory of LSTM algorithm. It remembers patterns.
- In our case it can for example remember if the S&P 500 index is in uptrend or downtrend.

Forget gate

- This component decides what information should be discarded in the cell state. It selects information that is no longer relevant.
- For example, after major drop on Monday, the forget gate decided that the stock market is no longer in uptrend. It discards the uptrend memory from the cell state.

Input gate

- Input gate decides which information is relevant and should be stored in a cell state.
- Suppose that one pattern emerged when predicting daily moves of S&P 500 index. It stores the pattern inside the cell state.

Output gate

- Decides what the next hidden state should be.
- In our case for example new information emerged and it changes the way the hidden state interprets data.

Design of LSTM

LSTM algorithms are constructed using layers. Each layer learns long-term dependencies, making the selection of the correct number of layers crucial when building LSTM. For example [6] discusses how ANN frequently yield inconsistent performance because of their incorrect design. We have implemented two distinct LSTM algorithms. The first algorithm, referred to as LSTM, consists of a single layer.

Second algorithm, referred to as LSTM Improved, incorporates four layers. Additionally, to each layer, a dropout rate of 20% is implemented. Dropout is a technique where 20% of the neurons within a layer are dropped, a strategy used to avoid overfitting.

2 Model

In this section, the primary objective is to develop a model that predicts if S&P 500 index will increase or decrease on the following day. The following day is denoted as x_{i+1}. To represent this model mathematically, we can define it as $p(x_{i+1}) = f(X_{i+1,1}, X_{i+1,2}, \ldots, X_{i+1,n})$, where f represents the specific algorithm applied to the features to compute the probability that on day x_{i+1} the S&P 500 index will increase.

The probabilities are subsequently transformed to classes. To facilitate this transformation from probabilities to binary outcomes, a threshold value will be incorporated. This threshold serves as a decision boundary in our model. If $p(x_{i+1}) >$ threshold, then the model's prediction for day x_{i+1} is 1. If $p(x_{i+1}) \leq$ threshold, then the model's prediction for day x_{i+1} is 0.

2.1 Data partitioning

Prior to testing our model, we must divide our data into three distinct sets - training, validation, and test set. The training set is data that is used to train our model. The validation set is data that is used as an experimental playground for finding the right features and for testing different algorithms. After we find the right features and algorithm for our model we evaluate performance of our model on the test set. The reason we do not use the validation set for evaluating our model is that our model had already seen the data and could use this to its advantage. Therefore we evaluate our model on a new data that the model have not seen before - a test set. The interval of validation set is 26.9.2017 - 22.11.2021. The interval of test set is 23.11.2021 - 24.1.2024.

The training set interval is different across feature groups as we aim to train the model with information that predates the specific data we intend to use for evaluating our model's performance. The intervals of the training sets are explained in Subsection 2.5.

Additionally, we evaluated our model's performance not just on the entire validation set, but also on two distinct validation subsets. In total, we evaluate our model's performance on these three intervals:

- **whole validation set:** 26.9.2017 - 22.11.2021
- **first validation subset:** 18.2.2020 - 23.3.2020
- **second validation subset:** 24.3.2020 - 22.11.2021

We chose this approach because each subset reflects different market conditions. For example, the subset from 18.2.2020, to 23.3.2020, represents the stock market crash because of the COVID-19 pandemic. The subset from 24.3.2020, to 22.11.2021, captures the subsequent bull market that emerged after the 2020 crash.

2.2 Model evaluation

Not only is it necessary to define on which data our model will be trained, but to also define how the performance of such model is measured. There are numerous ways to evaluate the success of a model, such as accuracy.

Accuracy proves to be valuable in many projects, calculating the percentage of successful predictions. However, we will not use accuracy as our primary performance metric in this thesis. The rationale behind this decision lies in the inherent limitations of accuracy as a metric.

Firstly, accuracy assumes that each prediction is of equal importance. In the context of our study, this assumption does not hold. For instance, consider day 1 where the market falls by 5 percent, and day 2 where it drops by 0.3 percent. Let us imagine two models: Model 1 accurately predicts day 1 but incorrectly predicts day 2, while Model 2 does the reverse. While both models have an accuracy of 50%, from our perspective, the model that accurately predicted day 1 (Model 1) is more valuable due to significantly higher decrease on day 1 than compared to day 2.

Secondly, accuracy as a metric assumes an equal distribution of outcomes. In our case, the distribution of days when the index goes up versus when it goes down is not necessarily equal. For example the percentage of days when S&P 500 index increased in our validation set is 56.96%. If our model would only predict that S&P 500 index will increase on the following day, the accuracy of such model is 56.96%, which could give us an illusion that our model can predict the market as the accuracy exceeds 50%. Though such model does not have any added value, since it always predict an increase on following day.

The issue of equal distribution of outcome can be avoided by using other metrics, such as the F1-score. Source [3] discusses F1-score. While accuracy focuses on the overall correctness of predictions, the F1-score is a harmonic mean of precision and recall. The definitions of recall and precision are as follows:

- Recall, often referred to as sensitivity or true positive rate, quantifies the proportion of actual positives that are correctly identified. It is calculated as $\text{Recall} = \frac{\text{number of true positives}}{\text{number of true positives + number of false negatives}}$.
- Precision, on the other hand, evaluates the accuracy of positive predictions calculated as $\text{Precision} = \frac{\text{number of true positives}}{\text{number of true positives + number of false positives}}$.

Combining these two, the F1-score is calculated as $F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$. However, the F1-score as accuracy assumes the equal importance of each outcome.

2.2.1 NODR

Thus, after being unable to identify a suitable metric, we have developed a customized performance metric which we call Number of Dollars Returned, abbreviated as NODR. This metric takes the form of an investment strategy, based on the following assumptions:

- At the beginning of the period for which we are calculating NODR, we invest 1 dollar.
- When the model predicts 1, we interpret this as a buy signal for the day x_i. In contrast, a prediction of 0 indicates that we should abstain from buying the index on the day x_i.
- We will calculate the daily gain by comparing the closing price of a particular day with the closing price of the previous day. We are holding the index just for one day.
- There are no transaction fees involved in buying and selling the index.
- Each time we buy the index, we invest 100% of our available capital.
- If on day x_i our model predicted a 0, indicating a decline, then we do not buy the index on day x_i, and on day x_{i+1} we have the same amount of money as on day x_i.

The result of this metric represents the total number of dollars returned based on the strategy described above. For clarity, if we say the NODR of a model with specific features from 1.1.2020 to 1.12.2020 is 2, it means we invested one dollar in January, followed the rules above, and by December 1st, we had 2 dollars.

To assess the effectiveness of our model-based strategy, we will compare its results with the returns of a straightforward buy-and-hold approach, where we purchase the S&P 500 index at the start of the test period and retain it throughout. Within the context of our metric, this buy-and-hold strategy is equivalent to making a continuous prediction of 1. We will denote its performance as NODR BUY AND HOLD.

This performance metric does not have some of the limitations associated with traditional measures like accuracy. It does not require the assumption that each prediction carries equal weight - the outcome of each prediction is reflected in the final result. It also compares the results to a buy-and-hold strategy. However even NODR has its own limitations.

First limitation of NODR - Calculating the return of the S&P 500 index

We are comparing the Close price from the previous day to the Close price of the target day to determine whether the index has risen or fallen. However, it is common for the opening price on the target day to differ from the previous day's closing price. While the appropriate way to calculate daily return of the S&P 500 index would involve calculating the difference between the opening and closing prices on the target day, we calculate returns as the difference between the closing prices of the target day and the previous day. Despite its impracticality in real-world scenarios—since buying an index at the closing price after trading hours is not possible we have opted for this approach.

The reason behind this choice is that it simplifies comparison between our model and the S&P 500 index in a basic buy-and-hold strategy. For instance, during a market downturn, the opening price is usually lower than the previous day's closing price. Even if our model only predicted a 1 (providing no added value), it would report a higher return than the S&P 500 index, merely because it was buying at the opening price. The same logic applies during an upward market trend, putting our model at a disadvantage. This issue is nonexistent when we compare closing prices.

Second limitation of NODR - Compound effect

As accuracy assumes equal importance of each day, we wanted to create a new metric that gives each day the appropriate weight when evaluating our model. Though we have given the appropriate weight each day by calculating how our investment changes while using our investment strategy it shall be mentioned that the earlier days have more weight in the final result. For example lets assume that the market will increase in next 30 days by 5 percent and that our model makes only one false prediction on the first day. The NODR will be 4.11. Though if we just buy and hold the index the NODR would be 4.32. The difference is 5.10%, which is slightly higher then 5%, which we did not obtain on first day due to wrong prediction. If we made a mistake on the last day, then the loss would be 5%. Mistakes that occur earlier in the period, during which we are calculating the NODR, carry more weight in the final value of NODR than mistakes occurring later in that period. It shall be mentioned that we used only 30 days. The difference would be much higher if we used 2 years.

We do not see this as a major obstacle due to the fact that on the test set we used a shorter interval than on validation set. In the example earlier we assumed that the index will increase continuously for 30 days. In reality this is considered unusual as the index has also downward movements. These downward movements reduce the compound effect.

It is important to note that our primary objective is to determine if daily movements in the S&P 500 index can be predicted, not to build a profit-generating model. We have adopted this investment strategy to emphasize the significance of each day's market movement, rather than treating all days as equal, as one would with an accuracy metric.

2.3 Constructing our model

Having established our model definition and evaluation metrics, we now turn our attention to the construction of the predictive model itself. This section outlines the principles we will follow for building machine learning models for financial forecasting.

In their work Suraphan Thawornwong and David Enke [14] describe fundamental principles for constructing such models. Several other studies we cite extensively reference this work [8, 9, 16], which underscores its significance in the field. A central premise in their approach is the importance of feature selection. They outline that selection of input features can greatly effect the model performance. Their research also emphasizes the necessity of feature elimination, stating that unimportant features can introduce confusion to models.

Despite advocating that effective market predictions are possible with appropriate feature selection, they acknowledge the inherent challenges due to the market's range of unpredictable factors. They have stated, that even with correct feature selection the markets remain difficult to predict. This conclusion is supported in other studies as well [1, 2].

2.4 Feature selection process

In the preceding sections, we emphasized the significance of feature selection for the successful implementation of our predictive model. The best method would be to try each combination of features. However the number of all combinations would be 2^n where n is the number of features. Due to the high number of combinations we decided to use a different approach.

We will now explain the process adopted in this study to find the most influential features. This process comprises two distinct methodologies for feature selection. Methodology A was designed and developed by the author of this thesis. Methodology B is externally sourced and widely acknowledged in the field of machine learning and statistical modeling.

It should be noted that prior to creating these methodologies we used various methods that are commonly used in feature selection for example like correlation matrix or extracting coefficients from Logistic Regression model. However these methods were inefficient and it was common that according to these methods none of the features were important. This was due to high level of randomness and low ability to predict the output. These methods are commonly used in projects like predicting heart diseases, where there is high correlation between the features and the outcome.

In both methodologies, we used NODR on the validation set to calculate the effectiveness of a model. In both methodologies we employed Logistic Regression to predict the labels of new data points, which were then used to calculate the NODR. We favored Logistic Regression over other algorithms because of its ability to deliver consistent results and its lack of need for hyperparameter optimization, unlike KNN and RF.

2.4.1 Methodology A

In this method, we sequentially add one feature at a time and then compute the NODR. If NODR is higher then in previous steps, it implies that the included feature enhances the model's performance and hence, is considered as useful. However if the method classifies certain feature as unimportant, the method will later not use that feature in further steps. Methodology A can be defined as:

Consider a feature vector X, where $X = [X_1, X_2, ..., X_n]$ represents a set of previously selected features. Let $\text{NODR}(X)$ be a function that calculates the NODR value for a given feature vector X on the validation set. We aim to evaluate the usefulness of p new features $X_{n+1}, \ldots, X_{n+p}$.

1. **Initialization:**

   - If $X = \emptyset$:
   $$\text{NODR}_{\text{max}} = \text{NODR BUY AND HOLD}$$
   - Else:
   $$\text{NODR}_{\text{max}} = \text{NODR}(X)$$

2. **Iterating over each new feature:** For each new feature X_{n+i}, where $i = 1, \ldots, p$, evaluate its importance using the following steps:

   (a) **Creating new feature vector**: Create an extended feature vector by adding the new feature to the current vector.
   $$X' = [X_1, \ldots, X_n, X_{n+i}]$$

   (b) **Compute new NODR**: Calculate NODR for the extended vector.
   $$\text{NODR}_{X'} = \text{NODR}(X')$$

   (c) **Evaluate feature's importance**:

   - If $\text{NODR}_{X'} > \text{NODR}_{\text{max}}$:
     - Consider X_{n+i} as a useful feature.
     - Update the NODR_{max}: $\text{NODR}_{\text{max}} = \text{NODR}_{X'}$.
     - Update the primary feature vector: $X = X'$.
     - In subsequent iterations, when evaluating new feature X_{n+i+1} that is after current feature X_{n+i} we will use the updated vector X and value NODR_{max}.
   - Else:

  - Disregard X_{n+i} as an unimportant feature.
  - Keep NODR_{max} and the primary feature vector X unchanged.

3. **Conclusion:** We obtain a new feature vector that includes all initial features $X_1, \ldots, X_n$ as well as the new features deemed useful by methodology A.

Let us assume that we have a initial feature vector X and new features C, D, E, and F. These features are just illustration and do not have any meaning. For these features, we will demonstrate how methodology A works.

Feature combinations used by methodology A	**NODR**
X	1.33
X, C	1.34
X, C, D	1.31
X, C, E	1.31
X, C, F	1.33

Table 1: Demonstrating methodology A

Table 1 shows that the model's performance improved upon the inclusion of feature C. It also improved after replacing E with F, declined with the addition of feature D, and remained unchanged after the replacement of feature D by feature E. After considering features D and E as non-essential, our method disregards it in the subsequent steps - therefore after classifying D as non essential we later tried combination of features X, C, E and not X, C, D, E. According to feature methodology A, only feature C is important. After classifying C as an important feature we change the feature vector of our model to $\mathbf{X} = [X_1, \ldots, X_n, C]$. It does classify feature F as unimportant because combination X, C, F yielded a lower NODR compared to combination X, C.

It's important to note that if the NODR remains the same after the inclusion of a feature, such a feature is not classified as important. This aligns with our earlier emphasis on using only the most crucial features.

However, this methodology has a disadvantage. The methodology does not account for the magnitude of improvement observed upon the addition of a feature. Even if the improvement is marginal (e.g., 0.01), it still considers the feature as important. Another disadvantage of methodology A is that the outcome of methodology depends on the sequence of features. Different sequences can yield different results.

2.4.2 Customized feature selection methodology B

Forward step-wise feature selection is commonly used for finding important features. Our implementation of this method aligns with the description in [4]. However we slightly modified the method, namely using NODR as a factor based on which we are going to eliminate the features. Feature methodology B can be defined as:

Suppose we have a vector X of previously selected features. We aim to evaluate the importance of p new features $X_{n+1},\dots,X_{n+p}$. Methodology B is done in rounds. The variable *round* denotes the current round.

1. **Initialization:**
   - If $X = \emptyset$:
     $$\text{NODR}_{\max} = \text{NODR BUY AND HOLD}$$
   - Else:
     $$\text{NODR}_{\max} = \text{NODR}(X)$$
   - Set round $= 1$.
2. **Rounds:** In each round follow these steps.
   (a) **Formation of new feature vectors**: For each new feature X_{n+i}, form a new feature vector, provided that feature X_{n+i} is not already included in X. Label these vectors with upper indices $X^{round,n+i}$, where $n+i$ equals to the index of newly added feature. The vectors are thus defined as:
   $$X^{round,n+i} = [X, X_{n+i}] \quad \text{if } X_{n+i} \notin X$$
   (b) **Evaluation of new vectors**: Calculate the NODR for every new vector.
   $\text{NODR}^{round,n+i} = \text{NODR}(X^{round,n+i})$
   (c) **Determine maximum NODR value**: Find the maximum NODR value of all new features vectors considered in this round and label the maximum value as NODR^{round}.
   (d) If $\text{NODR}^{round} > \text{NODR}_{\max}$:
      - Update $\text{NODR}_{\max}$:
        $$\text{NODR}_{\max} = \text{NODR}^{round}$$
      - Update the primary feature vector X to the new feature vector that had the highest NODR value: $X = X^{round,n+i}$.
      - round $=$ round $+ 1$.
      - Progress to the next round.
   (e) If $\text{NODR}^{round} \leq \text{NODR}_{\max}$: Terminate the methodology B, with X being the best combination of features, containing all initial features and features that methodology B classified as important.

Feature combinations used by methodology B	**NODR**
X, C	1.34
X, G	1.32
X, U	1.31
X, I	1.34

Table 2: Demonstrating methodology B - first round

Let's apply methodology B to an illustrative example, using features C, G, U, and I, and the initial feature vector X. In the first round, the method employs each feature independently to forecast the S&P 500 index.

In Table 2 we can see the result of round one. In this case, features X, C and X, I tie for the best outcome. When a tie occurs, the method selects the first feature combination, in this instance, X, C. Subsequently, the algorithm continues to round 2, which can be seen in Table 3. We continue to the second round because the feature combination X, C yields a higher NORD value than the NORD value of vector X. The NORD value of vector X is 1.33.

Feature combinations used by methodology B	**NODR**
X, C, G	1.35
X, C, U	1.33
X, C, I	1.41

Table 3: Demonstrating methodology B - second round

The combination X, C, I yields the highest NODR. Consequently, the method continues to round 3, shown in Figure 4.

Feature combinations used by methodology B	**NODR**
X, C, I, G	1.35
X, C, I, U	1.39

Table 4: Demonstrating methodology B - third round

Although the combination X, C, I, U yields highest NORD value in this round, its NORD value is lower than the previous round's best combination, X, C, I (1.39 versus 1.41). Therefore, the method stops and identifies X, C, I as the best feature combination, thus classifying features C and I as important. Additionally to the methodologies that are listed here we will also apply common sense.

2.5 Selecting important features for our model

Having explained our feature selection methodologies, we will now categorize our features into groups. Within each group, we will select important features for our model. The feature groups that we will later examine are listed here:

- Prior day percentage changes
- Open, high, low and close
- Opening price of the targeted day
- Volume
- Cboe volatility index
- Currency exchange rates
- Indices

In each feature group the feature vector X is determined by previous feature group, while the feature group Prior day percentage changes starts with empty feature vector X.

Please note the following facts:

- We used Python as our primary programming language.
- Code of this thesis is available on page 45.
- In methodologies A and B, we use Logistic Regression to calculate the NODR. In both methodologies, NODR is calculated on a validation set.
- We used the yfinance, a Yahoo Finance library, to extract data.
- We will use two distinct training sets. The first spans from 30.1.2001 to 25.9.2017. The second covers the period from 1.12.2003 to 25.9.2017. The necessity for two training sets arises from the inclusion of the feature Euro-Dollar percentage change in our model. We were unable to find data for the Euro-Dollar percentage change feature prior to 2003, as the Euro went into circulation in year 2002. For the feature groups currencies and indices, we employed the second training interval, while for other feature groups, we used the first training interval.
- Many of the features we used involve calculating percentage changes from previous days. Each time, we applied the following equation: Percentage change on day $i = \frac{\text{Value on day } i - \text{Value on day } i-1}{\text{Value on day } i-1} \times 100$.

For each feature group, we followed these steps:

1. We compute the features.
2. We apply methodologies A and B to identify the important features.

3. We use common sense to select important features from the outcomes of methodologies A and B. When applying common sense to identify important features, we consider only those features selected by methodologies A and B, with the exception in the feature group prior day percentage changes. Details of feature selection in feature group prior day percentage changes is explained in Subsection 2.5.1.

4. If we identify new important features for our model, we follow next steps. If we do not identify new important features, we continue to the next feature group.

5. We determine the hyperparameter values for KNN and RF. The values are found as follows: We calculate NODR for different hyperparameter values. Each time, we consider the same hyperparameter values:

   - For the variable number of estimators, we consider values 3, 5, 10, 50, 100, 150, 200.
   - For maximum depth, we consider values 1, 3, 5, 10, 20, 40, 80.
   - For neighbors, we consider values 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 25, 30, 50.

   We select the hyperparameters with the highest NODR value. The identified hyperparameter values are then used in subsequent steps.

6. We calculate the threshold matrix, which contains NODR values for various thresholds for each algorithm. The NODR values are computed on the validation set. For threshold, we consider values 0.2, 0.3, 0.4, 0.45, 0.5, 0.6, 0.7, 0.8. In subsequent steps, we employ the threshold for each algorithm with the highest NODR value.

7. After determining the thresholds, we compute the interval matrix, containing NODR values over different validation periods for each algorithm. The periods we use are described in Subsection 2.1.

We calculate the interval matrix only after identifying new important features.

2.5.1 Prior day percentage changes

This concept is borrowed from the book [4], where the authors predict the movement of the S&P 500 index one day ahead using the percentage changes from five days prior. We found this approach logical and plan to use it to predict the movement of the S&P 500 index based on the index's behavior a few days earlier. In our case, we used the nine days preceding the day we want to predict. These features are denoted as Day_1_Before through Day_9_Before. Later we will use shortcuts for features Day_1_Before through Day_9_Before. The shortcuts are D1B through D9B.

Methodologies A and B

In this subsection we will apply methodologies A and B in order to identify the most important features. Table 5 shows the result of methodology A. According to the methodology A the important features are D1B, D4B and D5B. In order to compare the results, the The NODR BUY AND HOLD for the validation period is 1.88.

Combinations of features	NODR
D1B	2.81
D1B, D2B	2.66
D1B, D3B	2.62
D1B, D4B	2.86
D1B, D4B, D5B	3.24
D1B, D4B, D5B D6B	3.14
D1B, D4B, D5B D7B	1.84
D1B, D4B, D5B D8B	2.81
D1B, D4B, D5B D9B	2.81

Table 5: Applied methodology A on prior day percentage changes

According to the methodology B the important features are D1B, D4B, D5B.

Applying common sense

It seems intuitively inconsistent to include D1B, D4B and D5B but skip D2B and D3B. This is also serves as an example why sometimes the methodologies can come to false conclusions and it outlines the importance to use common sense. We will be adding features one by one until we established a cutoff point. Currently we deem D1B as an important feature, as it delivered significantly better results across all algorithms. The result of the change in NODR can be seen in Table 6.

Algorithm	**26.9.2017-22.11.2021**	**18.2.2020-23.3.2020**	**23.3.2020-22.11.2021**
RF	+0.30	+0.23	-0.12
Logistic Regression	+0.93	+0.23	+0.24
KNN	+0.47	+0.27	-0.04
SVC	+0.24	+0.08	0.00
LSTM	+0.89	+0.23	+0.17
LSTM Improved	+0.96	+0.66	+0.22

Table 6: The change of NODR in selected intervals after incorporating feature D1B. It compares model with features D1B to NODR BUY AND HOLD. The value of NODR BUY AND HOLD is the same for all algorithms in the given interval. The value of NODR BUY AND HOLD is different for each interval. When calculating the NODR values in the table, we used the hyperparameter values for max depth, number of estimators, and thresholds that returned the highest NODR value.

Our conclusion aligns with [4], who also deemed D1B as an important feature. However, these authors additionally recognized D2B as an important feature. We decided to include D2B into our model. In Table 7, the changes in the NODR values within the interval matrix after incorporating feature D2B are displayed. We can see an improvement in all our algorithms, except for Logistic Regression. Since Logistic Regression is integral to all our feature selection methodologies, none of them selected D2B as an important feature.

Algorithm	**26.9.2017-22.11.2021**	**18.2.2020-23.3.2020**	**23.3.2020-22.11.2021**
RF	+0.53	+0,04	0.18
Logistic Regression	-0,15	0,00	-0.09
KNN	+0.56	+0.12	+0.10
SVC	+0.18	+0.08	+0.06
LSTM	+0.25	+0.04	+0.10
LSTM Improved	-0.03	+0.00	+0.01

Table 7: The change of in NODR after incorporating feature D2B. It compares model with features D1B, D2B to model with just feature D1B.

We decided to include D2B as a feature to our model. After falsely believing that feature D2B is unimportant, we also tried the combination of features D1B, D2B, D3B. Adding D3B as a feature resulted in a strong increase in performance of Logistic Regression. Therefore we will include D3B as well. This is also supported by logic that the behavior of an index three days prior are important information for our model. When it comes to deciding whether we should incorporate features D4B or D5B into our model, it is a challenging call to make. Nevertheless, we are inclined to believe that the difference will not be significant, regardless if we decide to include them as well.

Final conclusion

We decided that important features for our model are D1B, D2B and D3B. The X changed from an empty set to $X = [D1B, D2B, D3B]$. In Figure 7 we can see a graph that illustrates our investment if we chose to buy and sell the index, based on predictions created by the Logistic Regression on the validation set compared to the S&P 500 index. At first look, it may seem as if our model have an ability to forecast the S&P 500 index. However, this interpretation can be misleading.

The primary reason for this confusion is that our model showed impressive predictive performance during the stock market crash in 2020, while yielding slightly better results in the period preceding the crash. Moreover, it is evident that our model slightly outperformed the S&P 500 during an upward trending market phase (from 23.3.2020 to 22.11.2021) - achieving a NODR 2.14 compared to NODR BUY AND HOLD 2.03.

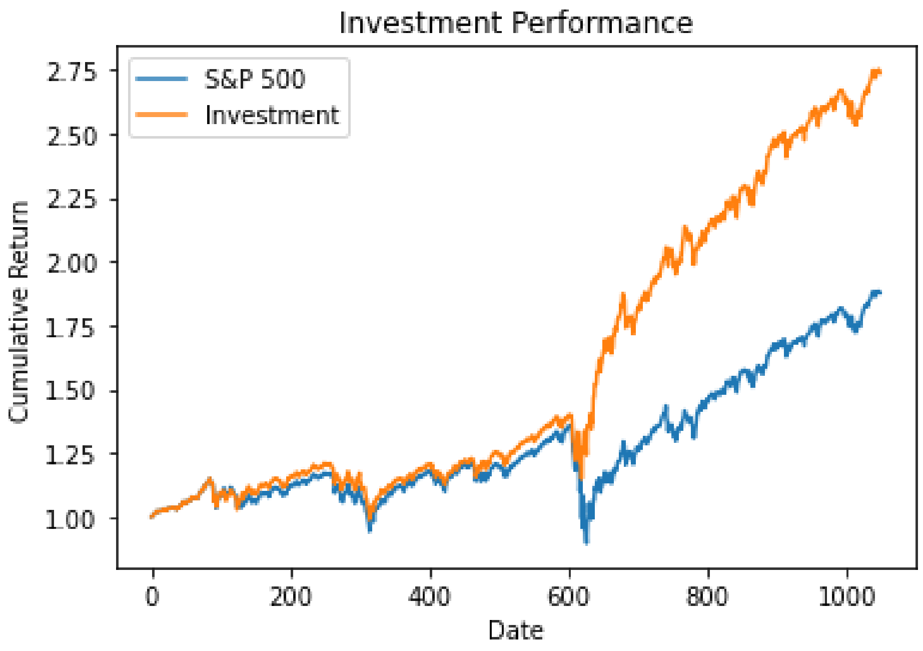

Figure 7: Evolution of our investment if we invest according to predictions generated by our model following the rules outlined in Subsection 2.2.1, compared to the S&P 500 index on the validation set (26.9.2017 - 22.11.2021). Another way to describe the graph is that it compares the NORD value over time to the S&P 500 index. The model consists of features D1B, D2B, D3B, and Logistic Regression as an algorithm, with a threshold value of 0.5 used.

It is crucial to consider that market conditions in our validation set may be different than those in our test set. Therefore the NODR on a validation set should not be the sole metric we rely on for feature selection, as this could potentially lead to the creation of models that are only good at predicting stock market crashes. We must also evaluate how our models performed during periods of upward market trends. This comprehensive approach promotes the development of a model that is capable of accurately predicting market movements, irrespective of whether the market conditions align with those seen in our test set. We also decided not only to visualize the graph but also to visualize confusion matrix.

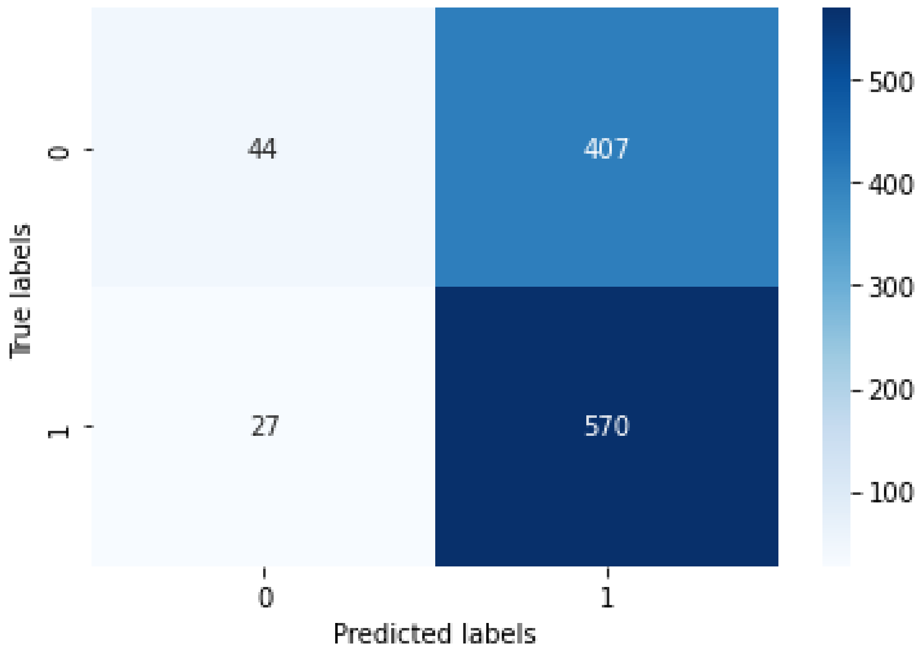

Figure 8: Confusion matrix of the model with features D1B, D2B, D3B. The predictions were made using Logistic Regression with a threshold of 0.5. The predictions were made on the validation set.

The confusion matrix displayed in Figure 8 provides a breakdown of our model's predictions, indicating the number of correct and incorrect classifications. The x-axis represents the true classes, while the y-axis shows our model's predicted labels. From the visualization, it is visible that our model has a tendency towards predicting class 1. Further, the model's accuracy score surpasses the 50% mark, coming in at 58.58%. It shall be mentioned however that the number of days when market rose was 56.96%. Therefore, from this point of view, the added value of our model is 1.62%.

2.5.2 Open, high, low, close from previous days

In addition to the existing data, we have access to the opening price, the highest price and lowest price for a given day. We created additional features based on these metrics. For instance, Open percentage change day one before, abbreviated as O represents the percentage change in the opening price a day ago. We followed the same calculation for the highest and lowest prices, abbreviated as H and L. We chose not to incorporate the closing price, as we already used such feature in preceding feature group (D1B). Upon applying our selection methodologies. Methodologies A and B classified features O, H and L as unimportant.

As methodologies A and B did not select any features as important, we will not include O, H and L in our model. The X remains the same as in previous feature group, $X = [D1B, D2B, D3B]$.

2.5.3 Opening price of the targeted date

Thus far, our approach to predicting the S&P 500 index has relied exclusively on data from previous days, excluding information on the target day. In this section, we introduce a feature named Change_Open_Close, representing the percentage change between the closing price a day before and the opening price of the target day. The theory suggests that if the opening price is higher than the previous day's closing price, the market is likely to rise that day. The NODR for Logistic Regression increased significantly from 2.7 to 20 after incorporating the feature Change_Open_Close. Therefore we find that this feature proves highly effective in predicting the movements of the S&P 500 index.

However there is a logical error. We are calculating the gains as a difference between close price and close price from previous day, not as difference between close price and open price of the same day. Let us use the following example:

Close price previous day: 50

Open Price: 52

Close price: 51

According to our investing strategy the index gained 2%. Adding a feature Change_Open_Close gives the model advantage, that the model knows the index already increased by 2 dollars overnight. Hence the probability that index close price will be lower then 50 dollars is smaller. Therefore after adding feature Change_Open_Close we gave our model knowledge what will happen in the future.

This subsection is a demonstration of how a logical error can improve our model's performance, where we used data from the next day. The study [13] suggests an accuracy of over 70% when predicting the next day's movement of the S&P 500 index. While the findings are interesting, such a high accuracy rate raises questions given the typically unpredictable nature of movements of the S&P 500 index. We examined the features in study [13], however not enough information was given.

2.5.4 Volume

Volume is an indicator that many people use to decide whether to buy or sell an index. In this subsection, we will first define volume and explain its common uses. Second, we will determine whether we should include volume as a feature in our model.

Definition of Volume and how volume is used in financial world

Volume is defined as the number of shares traded between the opening and closing hours. We used Volume as a feature, depicting the volume from the preceding day. We created an additional feature called Change of Volume, abbreviated as CV, which calculates the percentage change of volume from the previous day. Our model can use CV to filter randomness. Let us assume that an index rose by 0.2%. This could be random and not directly attributed to any significant change in the stock market. A significant drop in volume that day might indicate that the rise was random. Conversely, a higher volume could confirm the price movement.

This feature can also help our model detect abnormalities. If the volume rises significantly, say by 50%, it might suggest a major event that day, like an announcement of a change in interest rates by the Federal Reserve Board. Predicting whether the index will rise or fall in such scenarios is challenging, as the decisions of the Federal Reserve Board are uncertain.

Methodologies A and B

We applied methodologies A and B to determine which features are important to our model. According to methodology A, both CV and Volume are deemed as unimportant features. Methodology B classified the features CV and Volume as unimportant. Since both methodologies classified CV and Volume as unimportant, we will not include them in our model.

Comparison to other studies

Study [17] found that volume enhances model performance only in medium and long-term horizons. In contrast, volume did not yield improved performance in the short horizon and often led to irregular results. Our findings align with those in study [17]. We will not include CV or Volume in our model, therefore the X does not change - $X = [D1B, D2B, D3B]$.

2.5.5 Cboe volatility index

In this subsection we will look at Cboe volatility index (VIX) and if VIX can improve our model's performance.

Definition of VIX and its use in financial world

The VIX is real time index that represents market's expectations for the relative strength of near term price changes of S&P 500 index [12]. In simpler terms the VIX generates a 30-day forward projection of volatility. The volatility can be described as how strong the prices movements are. High volatility is often associated with insecurity.

Methodologies A and B

We introduced a new feature called VIX_change, which represents the percentage change of the VIX from the previous day. According to both methodologies A and B, this feature is deemed as unimportant. We attempted to combine VIX_change with various other features and observed inconsistent results. At times, the VIX_change feature enhanced the model's performance, while at other times, it led to a decline. Due to this inconsistency, we have chosen not to incorporate this feature into our model, therefore the X does not change - $X = [D1B, D2B, D3B]$.

Comparison to other studies

Study [12] explores the use of the VIX index in predicting the weekly movements of the S&P 500 index. The study concluded that the VIX index enhanced the model's performance. However we observed irregular outcomes from the VIX. We speculate that our differing conclusions arise from the distinct prediction intervals: the study aimed at weekly movements, where as our focus is on daily movements. This difference is significant. We believe that using the VIX to predict daily movements might be unsuitable due to the index's broader timeframe - it projects 30 day volatility. We believe it might be better to employ the VIX index for predicting weekly or monthly fluctuations of the S&P 500 index.

2.5.6 Currency exchange rates

The exchange rate of the US dollar with other currencies does not affect the S&P 500 index directly, but rather indirectly through exports and imports. For instance, if the value of the US dollar decreases compared to other currencies, exports will likely increase, and imports will decrease. This can influence companies' profits, such as increased sales to other countries. We introduced a new set of features named EUR, GBP, JPY, CHF, CNY, ZAR, and BRL. The abbreviations are explained as follows:

- **EUR:** Day-to-day percentage change in the Euro to US Dollar exchange rate.
- **GBP:** Day-to-day percentage change in the British pound to US dollar exchange rate.
- **JPY:** JPY: Day-to-day percentage change in the Japanese yen to US dollar exchange rate.
- **CHF:** Day-to-day percentage change in the Swiss franc to US dollar exchange rate.
- **CNY:** Day-to-day percentage change in the Chinese renminbi to US dollar exchange rate.
- **ZAR:** Day-to-day percentage change in the South African rand to US dollar exchange rate.

- **BRL:** Day-to-day percentage change in the Brazilian real to US dollar exchange rate.

We aimed to include all major currencies. We opted to exclude the Russian ruble due to its frequent fluctuations since the Ukrainian-Russian conflict in 2014. Although the Brazilian real and South African rand are not typically regarded as major currencies, we included them to balance the range of currencies, incorporating South America and Africa.

Conclusion

Methodology A identified the features EUR, GBP, JPY and CNY as important. Methodology B identified the features JPY and CNY as important. Both methodologies recognized the importance of features JPY and CNY, leading us to incorporate them into the feature vector. Additionally, methodology A identified the features EUR and GBP as important. However, we chose not to include the feature EUR in our model because the reason methodology A selected feature EUR was its first place in the sequence EUR, GBP, JPY, CHF, CNY, ZAR, BRL. We created a variation of this sequence where we switched the place of feature EUR with the place of feature GBP. In this new sequence variation, methodology A did not classify feature EUR as important.

We decided to include feature GBP in our model. In the initial round of methodology B, this feature had the second-highest NODR value after JPY (NODR 2.79 vs. NODR 2.80). Similarly, in the second round, feature GBP ranked second after CNY (NODR 2.95 vs. NODR 2.96). Given that feature GBP came close to being selected in methodology B, we have decided to incorporate it into our model. The X changes to

$X = [D1B, D2B, D3B, JPY, GBP, CNY]$.

Comparison with other studies

Study [16] reported CNY as the most influential feature in their model. Our results diverged. The JPY feature delivered a more substantial performance improvement in NODR values than CNY.

We believe that the disparity in findings arises from the time when the study was conducted, namely 2017. Different timeframes can significantly influence the importance of features. Additionally, Table 8 illustrates the change in NODR values after adding features JPY, GBP, and CNY to the feature vector.

Algorithm	26.9.2017-22.11.2021	18.2.2020-23.3.2020	23.3.2020-22.11.2021
RF	+0.52	+0.13	+0.22
Logistic Regression	-0.32	0.00	-0.19
KNN	+0.02	+0.18	-0.40
SVC	0.00	0.00	0.00
LSTM	+0.49	+0.13	-0.01
LSTM Improved	+0.42	+0.14	+0.03

Table 8: The change of NODR after incorporating features JPY, CNY and GBP. It compares model with features D1B, D2B, D3B to model with features D1B, D2B, D3B, JPY, CNY, GBP.

The NODR value for features D1B, D2B, and D3B, by using the Logistic Regression algorithm on a validation set, differs from those in the preceding sections. In the current section, the NODR value is 2.63, where as in the previous one, the NODR value was 2.74. The model achieving a NODR value of 2.63 was trained on data spanning from 1.12.2003, to 25.9.2017. The model yielding a NODR value of 2.74 was trained on data starting from 30.1.2001 until 25.9.2017. The different NODR values on the validation set is a direct result of these differing training sets, since models trained on different data sets yield different predictions.

The decision to change the training set was due to the incorporation of the EUR feature into our model. As Euro notes and banknotes began circulating in 2002, we did not obtain data on exchange rates before year 2003.

Table 8 illustrates the difference in NODR values between the interval matrices with features D1B, D2B, D3B, JPY, CNY, GBP, and the interval matrix with features D1B, D2B, D3B. Each of these algorithms in both internal matrices were trained on data spanning from 1.12.2003, to 25.9.2017. The selected hyperparameters and NODR values are different between the interval matrix with features D1B, D2B, D3B in this section and the corresponding matrix in the earlier sections, since the interval matrices were trained on different training sets.

2.5.7 Indices

In this section, we attempt to incorporate other indices into our model to predict the daily movements of the S&P 500 index. Several reasons for this decision are:

- **Diversified information**: Incorporating other indices helps the model understand sentiment across various sectors. For instance, the NASDAQ index tracks the technology sector in the USA, offering the model a more comprehensive view of the overall market.

- **Correlation among indices**: Major indices often correlate due to the interconnected nature of the global economy. For example, fluctuations in the index located in Korea could influence the S&P 500 index.
- **Risk factor modelling**: Different indices may respond differently to macroeconomic or global events. As an illustration, the Dow Jones Industrial Average is generally more stable than the S&P 500 index.

We have chosen to compute the percentage change for each index from the previous day. Here is a list of indices we aim to integrate into our model:

- **NASDAQ composite**: This index reflects the US technology sector.
- **Dow Jones industrial average**: This index represents the 30 largest companies on the US stock exchange.
- **Nikkei 225**: Monitors the 225 publicly-owned companies listed on the Tokyo stock exchange.
- **Hang Seng index**: Includes the most substantial, most liquid firms listed on the Hong Kong stock exchange.
- **FTSE 100**: Tracks the 100 biggest companies by capitalization on the London stock exchange.
- **DAX 30**: Features the 30 largest companies by capitalization on the Frankfurt stock exchange.
- **CAC 30**: Lists the 30 principal companies by capitalization on the Paris stock exchange.

According to both methodologies A and B, none of the indices are important. We have decided against incorporating any of these indices into our model. The vector X remains the same - $X = [D1B, D2B, D3B, JPY, GBP, CNY]$.

2.6 Summary of model selection

Table 9 presents the NODR value for each model with features D1B, D2B, D3B, JPY, GBP, CNY. Furthermore, the last row specifically displays the value NODR BUY AND HOLD in each period.

Algorithm	**26.9.17 - 22.11.21**	**18.2.20 - 23.3.20**	**23.3.20 - 22.11.21**
RF	1.64	0.69	1.81
Logistic Regression	3.02	0.89	2.36
KNN	2.19	0.79	2.11
SVC	1.87	0.66	2.04
LSTM	2.08	0.81	2.13
LSTM Improved	2.91	0.89	2.34
NODR BUY AND HOLD	1.87	0.66	2.04

Table 9: The NODR value of each algorithm on a validation set with features D1B, D2B, D3B, JPY, GBP, and CNY. Models were trained on a training set from 1.12.2003, to 25.9.2017.

Logistic Regression demonstrated the best overall performance on the validation set dated from 26.9.2017 to 22.11.2021. Additionally, it consistently delivered stable results. In contrast, models such as the LSTM often produced unpredictable outcomes. An added advantage of Logistic Regression is its independence from hyperparameter tuning, unlike methods like KNN and RF. Because of these advantages, we have chosen Logistic Regression as the algorithm for our final model.

Our final model for the test period uses the Logistic Regression algorithm with features D1B, D2B, D3B, JPY, GBP, and CNY. In Figure 9 is shown a graph that illustrates our investment if we chose to buy and sell the index, based on predictions created by the Logistic Regression on the validation set compared to the S&P 500 index.

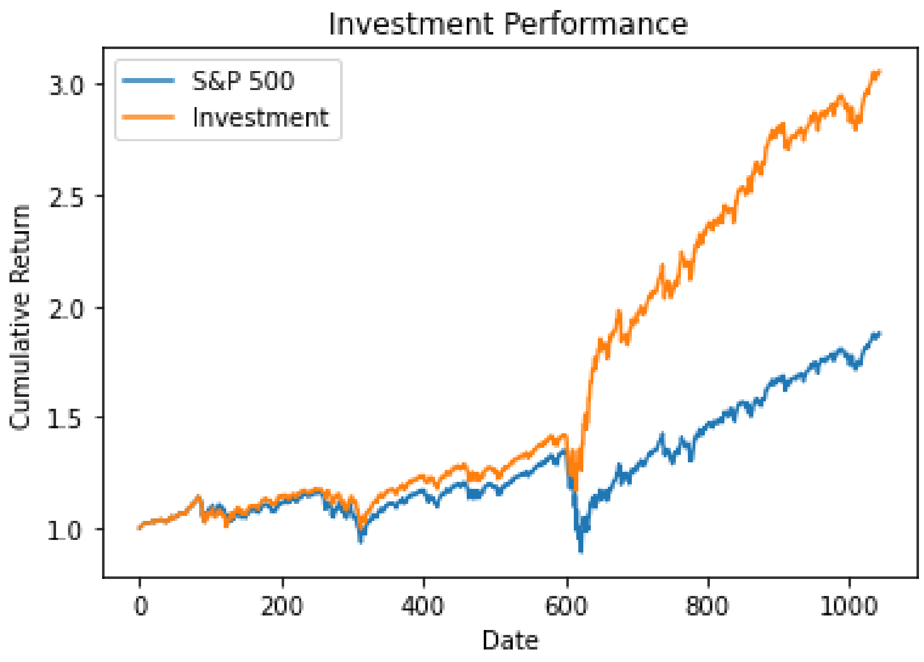

Figure 9: Evolution of our investment if we invest according to predictions generated by our model following the rules outlined in Subsection 2.2.1, compared to the S&P 500 index on the validation set (26.9.2017 - 22.11.2021). Another way to describe the graph is that it compares the NORD value over time to the S&P 500 index. The model consists of features D1B, D2B, D3B, JPY, GBP and CNY and Logistic Regression as an algorithm, with a threshold value of 0.5 used.

The model notably outperformed the S&P 500 index. This significant difference is due to our model's substantial gains during the 2020 COVID stock market crash. Such scenario does not have to happen in our test data.

2.7 Model performance evaluation on a test set

We will refer to the model that utilizes features D1B, D2B, D3B, JPY, CNY, GBP, and Logistic Regression as author's model. In Figure 10 is shown a graph that illustrates our investment if we chose to buy and sell the index, based on predictions created by author's model on the test set compared to the S&P 500 index. The test set spans from 23.11.2021 to 24.1.2024.

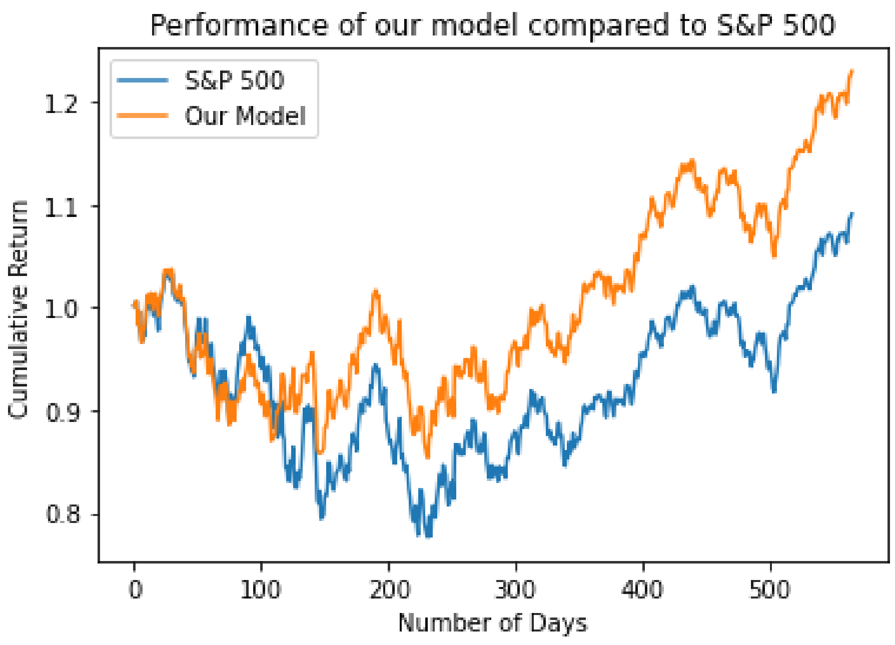

Figure 10: Performance of author's model on the test set. The blue line represents the performance of the S&P 500 index, while the orange line represents the NORD value over time if we invest based on predictions generated by the author's model. In this figure, the author's model performance is denoted by the orange line. The model uses features D1B, D2B, D3B, JPY, CNY, GBP and Logistic Regression as an algorithm.

The NODR value of the author's model on test set was 1.23, indicating that the author's model outperformed the S&P 500 index, thus demonstrating competence in predicting daily movements. author's model uses features selected not only through methodologies A and B but also by applying common sense. Features selected for the author's model differ from those selected by methodologies A and B. Furthermore, we evaluated the performance of models, whose features were selected by methodologies A and B. Figure 11 illustrates the performance of model with features selected by methodology B, which slightly outperformed the S&P 500 index, with a NODR value of 1.11. Methodology B selected features D1B, D4B, D5B, JPY, CNY. Figure 12 displays the performance of the model with features selected by methodology A, which also outperformed the index, with a NODR value of 1.14. Methodology A selected features D1B, D4B, D5B, JPY, CNY, GBP, EUR. Additional use of common sense improved the model's performance compared to just relying on methodologies A and B.

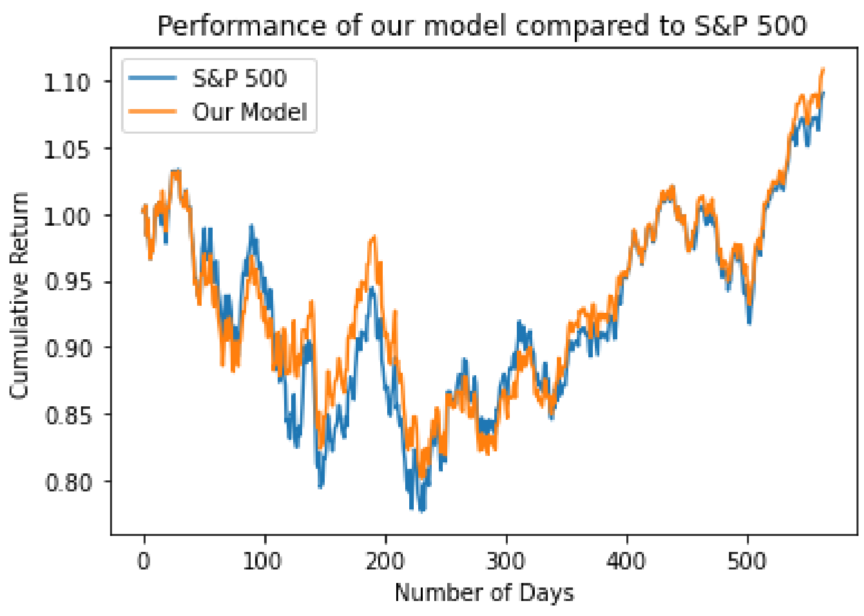

Figure 11: Performance of a model with features selected by methodology B on a test set. The blue line represents the performance of the S&P 500 index, while the orange line illustrates the NORD value over time if we invest based on predictions generated by the model, whose features were selected using methodology B. The model uses features D1B, D4B, D5B, JPY, CNY and Logistic Regression as an algorithm. Threshold value 0.5 is used.

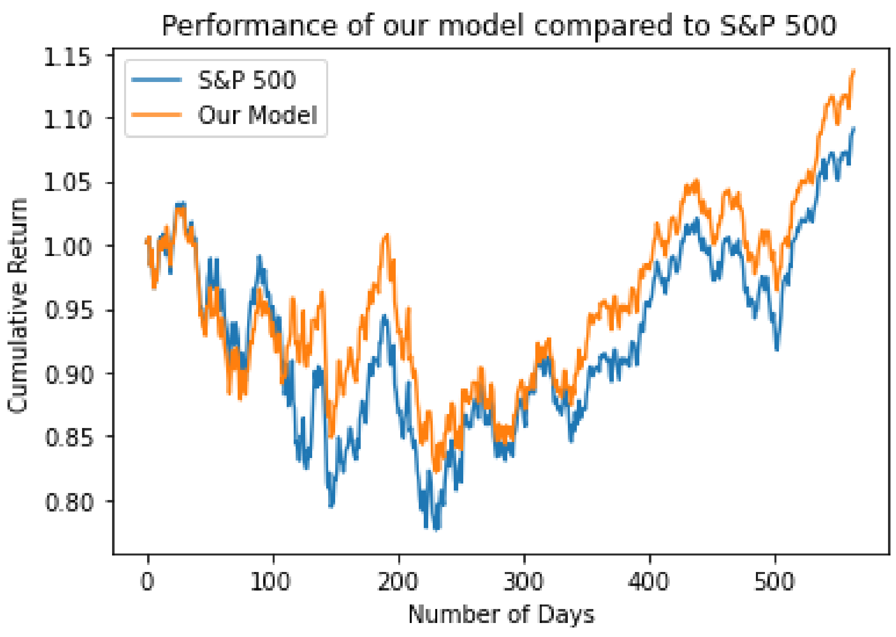

Figure 12: Performance of a model with features selected by methodology A on a test set. The model uses features D1B, D4B, D5B, JPY, CNY, GBP, EUR and Logistic Regression as an algorithm. Threshold value 0.5 is used.

Conclusion

The main goal of this thesis was to build a model that predicts the daily movements of the S&P 500 index. In Section 1, we explained the terminologies used throughout this bachelor thesis. Additionally, we provided explanations for each algorithm utilized in this thesis. The added value of this section lies in its simplified explanations compared to the sources we referenced, which are easier to understand for someone whose knowledge of machine learning is limited. Moreover, we employed Figures 2, 3, created by the author of this thesis, to visually illustrate the algorithms, enhancing the explanations.

In Section 2, we firstly defined model and specified the data sets to be used for training, evaluation, and testing. We also addressed the need to determine a suitable method for evaluating model's performance. Conventional metrics such as accuracy or F1-score were found insufficient, as explained in Subsection 2.2. We introduced our own metric called NODR in Subsection 2.2.1, along with an explanation of its significance and how it overcomes challenges associated with metrics like accuracy and F1-score. We could not find indications that such a metric had been previously used by other researchers.

The primary challenge in this thesis was feature selection, given its significant impact on machine learning model performance. We employed two feature selection methodologies A and B, detailed in Subsections 2.4.1, 2.4.2. Methodology A, created by the author, involves adding features one by one and deeming a feature important if its inclusion increases NODR. Methodology B is a customized version of the Forward step-wise feature selection method, distinguished by its use of NODR for model evaluation. In addition to methodologies A and B, we applied common sense to guide the selection of features for the model.

Features were selected from the following thematic groups: prior day percentage changes, open, high, low and close, volume, Cboe volatility index, currency exchange rates and indices. Additionally, we compared our findings with those of other academic papers that used similar features. Python served as our programming language, and yfinance as our data source. The code for this thesis is provided on page 45.

In Subsection 2.7, we presented a performance evaluation of the author's model, which was selected as our final model after evaluation on the validation set. The author's model uses D1B, D2B, D3B, JPY, CNY, and GBP as features, with Logistic Regression as the algorithm. Our findings demonstrate that author's model outperformed the S&P 500 index on the test set. Additionally, we conducted evaluations on models with features selected by methodologies A and B. Methodologies A and B underperformed compared to author's model, highlighting the significance of incorporating common sense in feature selection. Additionally, this thesis achieved its goal of building a machine learning model that surpasses the performance of the S&P 500 index.

This thesis could be enhanced by exploring new thematic feature groups, such as interest rates or technical indicators, to expand the range of considered features. Furthermore, enhancing this thesis could involve placing greater emphasis on refining the design of the LSTM algorithm.

Appendix A: Code

The code for this thesis is available in various formats, including PDF and Jupyter Notebook. Originally written in Jupyter Notebook, it is also provided in PDF format. Due to the extensive length of the PDF (100 pages), we have chosen not to include the code directly in this thesis. The code is available on GitHub under the username 'matej897' in the repository 'Projects'. The Github repository can be accessed via this link `https://github.com/matej897/Projects`. Additionally, the code is accessible via the following links:

- Link to PDF:

  `https://drive.google.com/file/d/1cgQ84xH-B5NkHmpPUR9VUA3e3Fld2NHQ/view?usp=sharing`.

- Link to Jupyter Notebook:

  `https://drive.google.com/file/d/1BjB_P6-ScyWj9IvN8CTzcghcw3W0AZnX/view?usp=sharing`.

References

[1] Atsalakis, G.S., Valavanis, K.P.: *Surveying Stock Market Forecasting Techniques - Part 2: Soft Computing Methods*, Expert Systems with Applications 36 (2009), 5932-5941.

[2] Cao, Q., Leggio, K.B., Schniederjans, M.J.: *A Comparison Between Fama and French's Model and Artificial Neural Networks in Predicting the Chinese Stock Market*, Computers & Operations Research 32 (2005), 2499-2512.

[3] DeVries, Z., et. al.: *Using a national surgical database to predict complications following posterior lumbar surgery and comparing the area under the curve and F1-score for the assessment of prognostic capability*, The Spine Journal 21 (2021), 1135-1142.

[4] Gareth, J., et. al.: *An introduction to statistical learning with Applications in R*, Springer, New York City, 2021.

[5] Goodfellow, I., Bengio, Y., Courville, A.: *Deep learning*, MIT Press, Cambridge, 2016.

[6] Kim, K.J., Ahn, H.: *Simultaneous Optimization of Artificial Neural Networks for Financial Forecasting*, Applied Intelligence volume 36 (2012), 887–898.

[7] Towards Data Science: *LSTM Networks | A Detailed Explanation*, available at (20.10.2023): `https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9`.

[8] Kia, A. N, Haratizadeh, S., Shouraki, S. B.: *A Hybrid Supervised Semi-supervised Graph-based Model to Predict One-day Ahead Movement of Global Stock Markets and Commodity Prices*, Expert Systems with Applications 105 (2018), 159-173.

[9] Niaki, S.T.A., Hoseinzade, S.: *Forecasting S&P 500 Index Using Artificial Neural Networks and Design of Experiments*, Journal of Industrial Engineering International 9 (2013), 2-9.

[10] Bilmes, J.: *Underfitting and Overfitting in machine learning*, educational texts, University of Washington, Washington, 2020, available at (14.10.2023): `https://people.ece.uw.edu/bilmes/classes/ee511/ee511_spring_2020/overfitting_underfitting.pdf`.

[11] Müller, A. C, Guido, S.: *Introduction to machine learning with Python*, O´Reilly, Sebastopol, 2016.

[12] Rosillo, R., Giner, J., de la Fuente, D.: *The Effectiveness of the Combined Use of VIX and Support Vector Machines on the Prediction of S&P 500*, Neural Computing and Applications 25 (2014), 321-332.

[13] Shen, S., Jiang, H., Zhang, T.: *Stock Market Forecasting Using Machine Learning Algorithms*, Department of Electrical Engineering, Stanford University, Stanford, 2012.

[14] Thawornwong, S., Enke, D.: *The Adaptive Selection of Financial and Economic Variables for Use with Artificial Neural Networks*, Neurocomputing 56 (2004), 205-232.

[15] Trask, A.: *Grokking Deep Learning*, Manning, New York City, 2019.

[16] Zhong, X., Enke, D.: *A Comprehensive Cluster and Classification Mining Procedure for Daily Stock Market Return Forecasting*, Neurocomputing 267 (2017), 152-168.

[17] Zhu, X., Wang, H., Xu, L., Li, H.: *Predicting Stock Index Increments by Neural Networks: The Role of Trading Volume Under Different Horizons*, Expert Systems with Applications 34 (2008), 3043-3054.