Employing Machine Learning Algorithms to build Trading Strategies with higher than Risk-Free Returns

Baris Yalin Uzunlu[®] **and Syed Muzammil Hussain**

Albert-Ludwigs-Universität Freiburg and Albert-Ludwigs-Universität Freiburg

Submission Time: 7 April 2020 | **Acceptance Time:** 11 September 2020

ABSTRACT

This research aims at exploring whether simple trading strategies developed using state-of-the-art Machine Learning (ML) algorithms can guarantee more than the risk-free rate of return or not. For this purpose, the direction of S&P 500 Index returns on every 6th day (*SPYRETDIR6*) and magnitude of S&P 500 Index daily returns (*SPYMAG*) were predicted on a broad selection of independent variables using various ML techniques. Using five consecutive data spans of equal length, GBM was found to provide highest prediction accuracy on *SPYRETDIR6*, consistently. In terms of magnitude prediction of daily returns (*SPYMAG*), Random Forest results indicated that there is a very high correlation between actual/predicted values of SPY. Based on these results, Trading Strategy #1 (using *SPYRETDIR6* predictions) and Trading Strategy #2 (using *SPYMAG* predictions) were developed and tested against a simple Buy & Hold benchmark of the same index. It was found that Trading Strategy #1 provides negative returns on all data spans, while Trading Strategy #2 has positive returns on average when data is separated into consecutive data spans. None of the trading strategies have a positive Sharpe ratio on average, but Trading Strategy #2 is almost as profitable as investing in T-bills using the risk-free rate.

Key words: *Machine Learning, S&P 500, Forecasting, Ensemble Methods, XGBoost*
JEL Classifications: C32, C45, C52, C53, C55

1. INTRODUCTION

There has been growing interest in the application of machine learning techniques in subject areas outside the realm of Artificial Intelligence (Ding et al., 2015). Research these days has become more empirical than ever before because of which the tools of statistics are indispensable to exploit the true characteristics of the data. The field of machine learning has developed some remarkable pattern recognition techniques in recent years, which has revolutionized the researching methods (Michalski and Kodratoff, 1990).

The generalizability of machine learning techniques allows users to apply them to a wide range of subject areas without compromising efficiency. This is precisely the reason why there has been a flood of literature in the area of financial economics, exploring the intricacies of the stock market using machine learning techniques. In this paper, we attempt to develop quantitative trading strategies by way of predicting S&P 500 Index returns' direction on every 6th day (*SPYRETDIR6*) and magnitude on daily basis (*SPYMAG*) using state of the art machine

[®] Baris Yalin Uzunlu, Alumni of M.Sc. Economics, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany (email: byuzunlu@sabanciuniv.edu), Tel: +90 507 057 51 18.
Syed Muzammil Hussain, Student in M.Sc. Economics, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany (email: syed.muzammil-hussain@gfk.com), Tel: +49 178 692 50 77.
The authors gratefully acknowledge the helpful and substantive comments of referees on concepts related to this paper.

learning techniques. Since there exists an inherent tradeoff between accuracy and interpretability, therefore, we shall only be resorting to those machine learning techniques that allow a certain level of interpretability (Gradient Boosting Machines, Logistic Regression, Random Forest, and XGBoost) and would refrain from complete black boxes like artificial neural networks.

The conventional theory dictates that stock markets react only to new information and old information is insufficient to beat financial markets. Although, the volume of algorithmic trading has reached its ever highest, however, naïve usage of machine learning techniques is far from sufficient to beat the market consistently even for short periods. The efficient market hypothesis has discussed the impossibility of beating the market using historical information long ago. Nevertheless, machine learning enthusiasts are biased towards using ML techniques in the hope to detect a consistent market-beating pattern, without developing any sound methodological framework.

Therefore, the problem of beating the market needs to be addressed using unconventional methods and from another dimension. The following three conditions need to be fulfilled before we can start predicting the outcomes in the stock market on a regular basis:

1. A well-researched methodological framework with strong theoretical underpinnings of financial market theory is required.

2. There should be a global model instead of a market-specific model, with consideration of interactions among the markets.

3. “Close to complete” historical information is needed in order to significantly beat the market consistently.

Nevertheless, to fulfill each of these conditions is a challenge of its own, even for seasoned financial researchers. Therefore, we intend to develop 2 trading strategies based on the prediction of direction and magnitude of SPY returns using machine learning algorithms. The profit/loss of ML-based trading strategies will then be compared with the risk-free return to analyze how good/bad some of the most powerful ML algorithm-based trading strategies perform. The idea is to see whether supervised ML algorithms when fed with theoretically justified input variables, can produce good enough predictions that can guarantee returns above the risk-free rate.

The remaining of our paper is comprised of seven further sections. The second section focuses on the review of the literature pertinent to our research topic. In the third section, the methodology adopted for this research is explained along with the chosen variables and performance metrics. The fourth section discusses the details of data preprocessing steps. The fifth and sixth sections provide the details of the results of *SPYRETDIR6* and *SPYMAG* prediction, respectively. The seventh section introduces two trading strategies based on the *SPYRETDIR6* and *SPYMAG* prediction results and their performance assessment compared to the benchmark strategy. The last section provides concluding remarks and highlights the important aspects of this study.

2. REVIEW OF LITERATURE

Considering the growing appetite of investors and researchers beating the market, it is not surprising that a vast number of researches exist in the literature. Especially due to the ongoing improvement in the Machine Learning area, classification techniques and AI models have become even more precise than ever to predict the complex non-linear market behavior. The usage of Machine Learning techniques in finance mainly focuses on the prediction of stock market indices and individual stocks.

2.1. Usage of ML Techniques for Stock Market Index Predictions

Diler (2003) has implemented 2-layered ANN (backpropagation with momentum algorithm) using 7 technical indicators, to predict the direction of daily return on the ISE-100 Index. His results show that the direction of daily returns on the ISE-100 Index is predictable with 60.81% accuracy on the Test set. Similarly, Egeli et al. (2003) have performed two different ANN methods (Multi-Layer Perceptron and the Generalized Feed Forward) to predict Istanbul Stock Exchange (ISE) market index value, while using Moving Average (MA) methods for 5 and 10-days as a benchmark. They have found out that both ANN methods outperform MA methods in terms of Mean Relative Percentage Error, and that the Generalized Feed Forward method slightly outperforms Multi-Layer Perceptron in terms of *R*-squared. Within the same year, Kim (2003) has used Support Vector Machines (SVMs), Back-Propagation Neural Networks (BPNNs), and Case-Based Reasoning (CBR) considering 12 technical variables as inputs, to predict the direction of change in the Daily Korea composite stock price index (KOSPI). He has concluded that SVM’s represent a solid alternative for financial time series forecasting and that it outperforms BPNNs and CBR in terms of hit ratio (57.83%, 54.73%, and 51.98% respectively, for holdout data). Rodriguez and Rodriguez (2004) took a leap forward and implied seven classification techniques to determine whether the direction of daily closing index levels of three emerging market indices (IPC – Mexico, KLSE Composite – Malaysia, and Bovespa – Brazil) is predictable. 8 technical indicators (1-day lagged) were used as inputs and Area under the ROC Curve (AUC) was used as the performance metric. They have reported that for all three indices, daily closing price movements are to be predicted better than random. They have also pointed out that Random Forest outperformed other classification techniques, as it is the only model that is able to predict the daily closing price movements in all three markets better than random. Kumar and Thenmozhi (2006) have used Support Vector Machines and Random Forest methods along with 12 technical variables as inputs, to predict the direction of movement of the SP CNX NIFTY Index. Then they compared the prediction results with other traditional techniques such as Discriminant Analysis, Logit Model, and Neural Network. They have found out that Support Vector Machines has the highest prediction accuracy (68.44%) among 5 classification methods being used, which slightly outperforms Random Forest (67.40%). Soon after, Yudong and Lenan (2009) brought a new perspective and applied Improved Bacterial Chemotaxis Optimization (IBCO) into Back-Propagation Artificial Neural Network (BPNN) approach. They have used 10 technical variables to predict the closing price of the S&P 500 Index value for 1-day in advance (called as short-time prediction) and 15-days in advance (called as long-time prediction). Their results indicate that the IBCO-BPNN approach provides better prediction accuracy than traditional BPNN and is computationally less complex. Kara et al. (2011) have used Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to predict the direction of movement in the Daily Istanbul Stock Exchange National 100 Index, using 10 technical variables. After applying parameter setting experiments, they have found out that the average performance of the best ANN model significantly outperforms the selected SVM model in terms of ISE 100 prediction (75.74% and 71.52%, respectively). Imandoust and Bolandraftar (2014) have used three classification methods, namely Random Forest, Decision Tree, and Naive Bayesian Classifier, in order to predict the direction of movement in the Daily Tehran Stock Exchange Index. They have found out that while using 10 technical and 3 fundamental variables as inputs; Decision Tree method slightly outperforms Random Forest and Naive Bayesian Classifier, in terms of hit rate (respectively 80.08%, 78.39%, and 72.15%). Finally, Labiad et al. (2016) have trained three Machine Learning techniques (Random Forest, Gradient Boosted Trees, and Support Vector Machines) to predict a very short period (10 minutes ahead) of the Moroccan Stock Exchange, using 23 technical indicators. They have concluded that Random Forest and Gradient Boosted Trees outperform Support Vector Machines in terms of testing accuracy (95%, 94%, and 90% respectively).

2.2. Usage of ML Techniques for Individual Stock Price Predictions

In the early 1990s, Yoon and Swales (1991) have compared the 4-layered Neural Network (NN) model with Multiple Discriminant Analysis methods (MDA) to evaluate the stock price performance prediction of a total 98 companies, using 9 qualitative independent and 2 qualitative dependent variables. Variables were extracted from the president’s letter to stockholders in the annual report of each company. They have proved that for both training and testing data, the NN method outperforms the MDA method (91% over 74%, and 77.5% over 65%, respectively). Atsalakis and Valavanis (2009) have published a compiled research and reviewed over 100 scientific articles that were written in terms of stock market prediction, and particularly focused on Neural Network - Neuro-Fuzzy Models. They have classified each reviewed article in terms of the input data used, forecasting methodology, model comparisons, and performance evaluation measurements. After the classification and detailed analyses of all articles, they have concluded that soft-computing models (Neural Networks and Neuro-Fuzzy Models) outperform traditional models in terms of market prediction, and provide higher prediction accuracy, in most of the cases. Ballings et al. (2015) have benchmarked three ensemble classification methods (Random Forest, AdaBoost, and Kernel Factory) against four single classifier models (Logistic Regression, Neural Networks, K-Nearest Neighbors, and Support Vector Machines) in order to predict whether the stock prices of selected 5767 European companies go up by a predetermined amount. They have used various balance sheet and P&L statement variables as well as some general economic features as inputs. They also wanted to examine if ensemble classification methods can outperform single classifier models in terms of stock price direction prediction. Results conclude that Random Forest is the top predictor (with median AUC value of 0.9037) among all seven models, and significantly outperforms all other classifiers except SVM. Finally, Gündüz et al. (2017) have used Logistic Regression and Gradient Boosting Machine in order to predict daily closing returns of the three most traded stocks in ISE (GARAN, THYAO, and ISCTR). They have also applied feature selection methods to obtain the optimal number of features. Various technical and fundamental variables were used as inputs. Results show that applying feature selection methods increase the prediction performance for selected stocks. In terms of prediction accuracy, after applying the feature selection process, GBM resulted in 59.90% for GARAN and 55.80% for THYAO, while LR resulted in 58.10% for ISCTR.

2.3. Usage of ML Techniques Combined with News Articles for Individual Stock/Index Price Predictions

Wütrich et al. (1998) have used several ML techniques (such as KNN and Neural Network algorithms) to daily price movements of 5 main stock indices (Dow Jones Industrial Average, Nikkei 225, Financial Times 100 Index, Hang Seng Index, and Singapore Straits Index), using news articles collected from the leading newspapers. They have come up with an average accuracy of 43.6%, for all 5 stock indices, in terms of predicting their daily price movements. Zhai et al. (2007) have combined news articles as well as 7 technical indicators and implemented Support Vector Machines using them, to predict stock price movements of BHP Billiton Ltd. in the Australian Stock Exchange. They also have run a market simulation in order to check whether the system is profitable. Their results indicate that both prediction accuracy and profitability of the system increase with using technical indicators and news data, as inputs (Prediction accuracy of 70.10% while combining news data and technical indicators, while prediction accuracy of 58.80% without using news data).

3. METHODOLOGY AND DATA DESCRIPTION

Three of the most celebrated ML techniques i.e. Gradient Boosting Machines, Random Forest, and Extreme Gradient Boosting are used to forecast the direction of SPY returns. Logistic Regression is used as the base algorithm to compare how well the most celebrated techniques perform when it comes to the forecasting of financial time series. For the magnitude prediction of SPY returns, Random Forest is used.

A total of 32 independent variables and 2 dependent variables were used. Among 32 independent variables, 12 of them are fundamental variables and 20 of them are technical indicators.

3.1. Dependent Variables (*SPYRETDIR6* and *SPYMAG*)

SPYRETDIR6: The direction of the return of the S&P 500 ETF (SPY) on every 6th day.

Daily adjusted closing prices of SPY were extracted from Yahoo Finance (YAHOO,.2019b). It is known as a good proxy for the S&P 500 Index and is used because it is a traded financial instrument. It is calculated using the following formula:

$$\frac{Adjusted\ Closing\ Price\ of\ SPY_t - Adjusted\ Closing\ Price\ of\ SPY_{\ t-6}}{Adjusted\ Closing\ Price\ of\ SPY_{\ t-6}}$$

SPYMAG: The magnitude of the daily return of the S&P 500 ETF (SPY). It is calculated using the following formula:

$$\frac{Adjusted\ Closing\ Price\ of\ SPY_t - Adjusted\ Closing\ Price\ of\ SPY_{\ t-1}}{Adjusted\ Closing\ Price\ of\ SPY_{\ t-1}}$$

3.2. Independent Variables

Comprised of fundamental and technical indicators. The full list of independent variables (fundamental – technical) as well as their formulas are presented in the Appendix. The selection of independent variables was made mainly based on the literature. The final set of 32 independent variables was formed after removing various variables due to some reasons such as high multicollinearity, non-numeric outcomes, too many missing values, or low correlation with the target variable. Also in order to avoid look-forward bias, all observations of independent variables are lagged for 1-day.

3.3. Performance Metrics

Even though overall accuracy (Hit rate) was used as the main performance metric for *SPYRETDIR6* prediction, 3 other metrics were also calculated from confusion matrices. Definitions and calculations of them are as follows:

Precision: It is a measure that provides the information about how much of the positive returns on SPY detected by the algorithm, were actually positive in the Test Set (Sunasra, 2017). The formula for Precision calculation is;

$$Precision = \frac{True\ Positives}{True\ Positives\ +\ False\ Positives}$$

Recall (Sensitivity): It measures how well the algorithm correctly predicted the positive returns on SPY among the actual positives in the Test Set (Sunasra, 2017). The formula for Recall (Sensitivity) calculation is;

$$Recall\ (Sensitivity) = \frac{True\ Positives}{True\ Positives\ +\ False\ Negatives}$$

F1 Score: It represents the harmonic mean of Precision and Recall (Sunasra, 2017). The formula for F1 Score calculation is;

$$F1\,Score = 2 * \frac{Precision\ *\ Recall}{Precision\ +\ Recall}$$

Accuracy (Hit Rate): Overall accuracy of the model. The formula for Accuracy (Hit Rate) calculation is:

$$Hit\ Rate = \frac{True\ Positives\ +\ True\ Negatives}{True\ Positives\ +\ True\ Negatives\ +\ False\ Positives\ +\ False\ Negatives}$$

Regarding *SPYMAG* prediction, the correlation coefficient is used as the performance metric. The formula is given as:

$$\frac{\sum(SPY_{ACTUAL} - \overline{SPY}_{ACTUAL}) * (SPY_{PREDICTED} - \overline{SPY}_{PREDICTED})}{\sqrt{\sum(SPY_{ACTUAL} - \overline{SPY}_{ACTUAL})^2 * \sum(SPY_{PREDICTED} - \overline{SPY}_{PREDICTED})^2}}$$

Applications of all ML techniques were made using the R program while the collection and adjustment of selected variables' data were made using Microsoft Excel.

4. DATA PREPROCESSING

4.1. Data Spans

The entire dataset was divided into five equal-sized data spans. The purpose is to train the model on the earliest dataset and check its temporal consistency. Each data span consists of 561 observations (trading days).

Exact dates of the data spans are given as:

1st Data Span:	Nov 16, 2016 – Feb 11, 2019
2nd Data Span:	Dec 16, 2016 – Mar 13, 2019
3rd Data Span:	Jan 17, 2017 – Apr 09, 2019
4th Data Span:	Feb 16, 2017 – May 10, 2019
5th Data Span:	Mar 16, 2017 – Jun 07, 2019

Roughly 2 years of data were used; as due to the dynamic and evolving nature of stock markets, recent historical data tends to be more informative in terms of prediction. For each Machine Learning algorithm used in this research:

1- Initially, each model with a grid search for hyperparameter values was trained on the entire dataset.

2- After finding the best model with optimum hyperparameter values for the 1st data span, the same hyperparameters were used in other data spans to check the prediction accuracy of *SPYRETDIR6* on the Test set.

As mentioned by Atsalakis and Valavanis (2009), missing observations were replaced by the most recent previous observation. Also, data is normalized between [0,1]. This is done to avoid models assigning incorrectly more weight to variables that have higher unit value.

4.2. Training/Test Split

A train-test split of 70-30 was used. However; due to the time-series nature of our dataset, temporal order is needed to be preserved in the split. Therefore, instead of performing a random split; for all data spans, first 70% of the total observations were used as Training Set and the last 30% of the total observations were used as Test Set.

5. RESULTS & DISCUSSION

For each of the 4 ML algorithms trained in this research, we present here the results obtained using the entire dataset. Spanwise results for each algorithm can be found in the Appendix section.

5.1. ML Algorithm 1 – Gradient Boosting Machine

5.1.1. Steps for Calculating GBM Results

GBM results consist of 4 main steps:

1- Initially, models with different hyperparameter values were compared to each other using no data spans and the best one was selected in terms of prediction accuracy on *SPYRETDIR6.*

2- Using the same hyperparameter values, the selected model was run for all data spans.

3- Results are compared in terms of their prediction accuracy and other performance metrics.

4- The variable importance of the selected model for all data spans was calculated.

5.1.2. Procedure of Running GBM in R

1- Using the “gbm” function, the GBM algorithm is performed on the Train set, regressing *SPYRETDIR6* on all 32 independent variables after tuning hyperparameters and performing 10-fold cross-validation.

2- Then, using GBM results, *SPYRETDIR6* is predicted on the Test set with planting the optimal number of trees specified by hyperparameter tuning.

3- After prediction, column names are labeled and results are presented in data.frame format.

4- As the last step, the confusion matrix is created to calculate the accuracy and other performance metrics.

5.1.3. Definition of Hyperparameters Used

1- Cv.folds: number of cross-validations to perform.

2- Shrinkage: Learning rate. Smaller values tend to give better prediction results, however, will require more iterations to find the minimum of the loss function and will be computationally expensive (Ridgeway, 2007).

3- N.minobsinnode: Minimum number of samples in the terminal nodes of each tree.

4- N.trees: Total number of trees to grow. To avoid overfitting, finding the optimum number of trees to minimize the loss function is crucial (“Gradient Boosting”, 2018).

5.1.4. Results

As using Automated Grid Search to tune hyperparameters is computationally too expensive; we have conducted a Manual Grid Search to tune 2 hyperparameters (shrinkage and n.trees). As suggested by Ridgeway (2007), the strategy is to use as small shrinkage values as possible and then tuning the number of boosting trees. Therefore, 4 different possible values of shrinkage (0.1, 0.01, 0.001, and 0.0001) were applied growing different numbers of trees (100,500,1000,2500 and 5000) which makes a total of 20 different combinations. Then they were fed into gbm() function to come up with the highest accuracy on the Test Set, without using data spans. The rest of the hyperparameters in the grid function are set as default values. Results show that growing5000 classification trees using multinomial distribution, applying 10-fold cross-validation, and setting values of 0.001 as shrinkage (learning rate) as well as 5 as n.minobsinnode provides the highest accuracy for *SPYRETDIR6* on the Test Set, which is 83.52% (Table 5.1).

However, this result is obtained using no data span. In order to check the consistency of total prediction accuracy of 83.52%, approximately 1-month lag is used (repeated 5 times) and prediction accuracy is calculated for each data span, using the same hyperparameter values (Ceteris Paribus). Results indicate that the arithmetic mean of prediction accuracy for 5 consecutive periods is 91.07% with a standard deviation of 0.65%. This means that results are far better than naive random and very consistent for all data spans, due to the notably low standard deviation. Table 5.2 compares the accuracy (Hit Rate) results for all data spans, while Table 5.3 provides arithmetic mean and standard deviation values of them.

distribution	cv.folds	shrinkage	n.minobsinnode	n.trees	RESULT
multinomial	10	0.1	5	100	82.50%
multinomial	10	0.1	5	500	79.07%
multinomial	10	0.1	5	1000	78.52%
multinomial	10	0.1	5	2500	70.93%
multinomial	10	0.1	5	5000	72.13%
multinomial	10	0.01	5	100	81.85%
multinomial	10	0.01	5	500	83.43%
multinomial	10	0.01	5	1000	83.33%
multinomial	10	0.01	5	2500	83.33%
multinomial	10	0.01	5	5000	81.02%
multinomial	10	0.001	5	100	79.81%
multinomial	10	0.001	5	500	80.09%
multinomial	10	0.001	5	1000	81.94%
multinomial	10	0.001	5	2500	83.33%
multinomial	**10**	**0.001**	**5**	**5000**	**83.52%**
multinomial	10	0.0001	5	100	79.81%
multinomial	10	0.0001	5	500	79.81%
multinomial	10	0.0001	5	1000	79.81%
multinomial	10	0.0001	5	2500	79.72%
multinomial	10	0.0001	5	5000	80.19%

Table 5.1 Results of Hyperparameter Tuning Combinations in GBM (No Data Span).

DATA SPAN	# of Obs	distribution	K-Folds	shrinkage	n.minobsinnode	n.trees	ACCURACY
1	561	multinomial	10	0.001	5	5000	91.07%
2	561	multinomial	10	0.001	5	5000	90.48%
3	561	multinomial	10	0.001	5	5000	92.26%
4	561	multinomial	10	0.001	5	5000	91.07%
5	561	multinomial	10	0.001	5	5000	90.48%

Table 5.2 GBM Results – All Data Spans.

DATA SPAN	ACCURACY
1	91.07%
2	90.48%
3	92.26%
4	91.07%
5	90.48%
MEAN	**91.07%**
STD. DEV	**0.65%**

Table 5.3 GBM Results - Arithmetic Mean and Variance of Accuracy for All Data Spans.

DATA SPAN	Precision	Recall (Sensitivity)	F1 Score	Hit Rate
1	92.63%	91.67%	92.15%	91.07%
2	93.94%	90.29%	92.08%	90.48%
3	93.00%	93.94%	93.47%	92.26%
4	92.93%	92.00%	92.46%	91.07%
5	92.31%	90.32%	91.30%	90.48%
MEAN	**92.96%**	**91.64%**	**92.29%**	**91.07%**
STD. DEV	**0.55%**	**1.34%**	**0.70%**	**0.65%**

Table 5.4 GBM Performance Metrics for All Data Spans.

Given in Table 5.4, in terms of performance metrics; on average, 92.96% of the returns on SPY that are predicted by our model as positive were actually positive, which denotes the Precision metric. Also, 91.64% of the actual positives in the Test Set were predicted correctly, which denotes the Recall metric. Regarding the F1 score, we come up with an accuracy of 92.29% for all data spans.

When it comes to the variable importance, “summary.gbm” was used in R. It is interesting to note that among the 10 most important variables; only 3 fundamental variables (DJIA1WR, COIL1WR, and GOLD1DR) and 5 technical indicators (OSCP, MOMENTUM5, STOCHASTICK, CCI, and ADOSCILLATOR) are common for all data spans, regarding the relative influence on the *SPYRETDIR6*. Therefore, there is also evidence that technical indicators in the GBM algorithm tend to have a higher influence on predicting *SPYRETDIR6*.

5.2. ML Algorithm 2 – Random Forest

5.2.1. Steps for Calculating RF Results

Steps for calculating the results of the Random Forest model are exactly the same as GBM.

5.2.2. Procedure of Running RF in R

1- Using the “randomForest” function, the Random Forest algorithm is performed on the Train set, regressing *SPYRETDIR6* on all 32 independent variables after tuning hyperparameters.

2- Then, using Random Forest results, *SPYRETDIR6* is predicted on the Test set.

3- As the last step, the confusion matrix is created to calculate the accuracy and other performance metrics.

5.2.3. Definition of Hyperparameters Used

1- ntree: Number of trees grown in the forest.

2- mtry: Number of predictors used for splitting in every node (STDTC, 2019).

3- min_samples_split: Minimum number of samples that are used to split an internal node (Pedregosa et. al., 2011).

4- min_samples_leaf: Minimum number of samples that each leaf node should have. At any depth, split points are considered if and only if it leaves the number of training samples equal to min_samples_leaf, for each left and right branches (STDTC, 2019).

5- max_depth: Maximum depth of each classification tree grown in the model.

6- importance: If TRUE, the model calculates the feature importance.

7- max_features: Number of features to be considered while searching for the best split. If it is set as “Auto”, then for classification problems, it equals to the square root of total features (STDTC, 2019).

5.2.4. Results

Similar to GBM; due to the computational complexity, tuneRF() function was used to tune the optimal value of mtry and Manuel Grid Search was applied to compare 5 different combinations of hyperparameter values without using any data spans. Results indicate that the highest prediction accuracy on *SPYRETDIR6* is 83.89%, with hyperparameter values of:
mtry = 7, ntree = 100, max_features = ”Auto”, max_depth = 10, min_samples_split = 10, min_samples_leaf = 10 (Table 5.5)

mtry	ntree	max_features	max_depth	min_samples_split	min_samples_leaf	RESULT
7	**100**	**Auto**	**10**	**10**	**10**	**83.89%**
7	500	Auto	10	10	10	83.33%
7	1000	Auto	10	10	10	83.15%
7	2500	Auto	10	10	10	83.80%
7	5000	Auto	10	10	10	83.33%
7	100	Auto	10	10	10	83.89%

Table 5.5 Results of Hyperparameter Tuning Combinations in RF (No Data Span).

Similar to GBM, the prediction accuracy of *SPYRETDIR6* in RF is quite high without using data spans. In order to check the consistency of this result, the same procedure with GBM was followed.

DATA SPAN	ACCURACY (HIT RATE)
1	90.48%
2	89.29%
3	93.45%
4	89.88%
5	89.88%
MEAN	**90.60%**
STD. DEV	**1.48%**

Table 5.6 RF Results – All Data Spans.

As shown in Table 5.6, using the same hyperparameters that gives that best prediction accuracy on *SPYRETDIR6* without using data spans make results very slightly vary for different data spans with a mean of 90.60% and standard deviation of 1.48%. Therefore, results are statistically significant and consistent due to the low standard deviation. It is also notable that our prediction accuracy is slightly higher when data spans are introduced to the model. Although this provides evidence against the theory that markets are living entities and hyperparameters of algorithms have to be adjusted in accordance with the changing market structure, it is not possible to prove the opposite using solely these results.

DATA SPAN	Precision	Recall (Sensitivity)	F1 Score	Hit Rate
1	90.63%	92.55%	91.58%	90.48%
2	87.38%	94.74%	90.91%	89.29%
3	94.95%	94.00%	94.47%	93.45%
4	91.00%	91.92%	91.46%	89.88%
5	87.10%	94.19%	90.50%	89.88%
MEAN	**90.21%**	**93.48%**	**91.78%**	**90.60%**
STD. DEV	**2.86%**	**1.06%**	**1.40%**	**1.48%**

Table 5.7 RF Performance Metrics for All Data Spans.

Regarding performance metrics given in Table 5.7; on average, 90.21% of the positive returns on SPY detected by our random forest model were actually positive in the Test Set (Precision).

Similarly, our random forest model correctly predicted 93.48% of the actual positives in the Test Set (Recall) and the harmonic mean of them is 91.78% (F1 Score).

In terms of variable importance, “importance” and “varImpPlot” functions were applied. Results show that among the 5 most important variables, only 3 technical indicators (*DJIA1WR*, *LARRYWILLIAMSR*, and *MOMENTUM5*) are common for all data spans. Therefore, there is evidence supporting that technical variables tend to have a higher influence on predicting *SPYRETDIR6*.

5.3. ML Algorithm 3 – Extreme Gradient Boosting

5.3.1. Steps for Calculating XGB Results

Steps for calculating the results of the XGB model are exactly the same with GBM and RF.

5.3.2. Procedure of Running XGB in R

1- Train and Test sets were converted into matrix form, as XGB requires.

2- Hyperparameters are tuned using an Automated Grid Search and 10-fold cross-validation is performed to choose the best iteration.

3- Using the “xgb.train” function, the XGB algorithm is performed on the Train set and the prediction is done on the Test set.

4- As the last step, the confusion matrix is created to calculate the Hit rate and other performance metrics.

5.3.3. Definition of Hyperparameters Used

1- booster = “gbtree”: Type of the booster used. This indicates that tree-based models are used.

2- objective = “binary:logistic”: Denotes the logistic regression for binary classification (XGBOOST, 2020).

3- eta: Learning rate. Level of shrinkage for feature weights to come up with a more conservative boosting process, in order to prevent overfitting (XGBOOST, 2020).

4- gamma: Denotes the minimum level of loss reduction in order to perform another partition on a leaf node of the tree. Larger values result in a more conservative algorithm (XGBOOST, 2020).

5- max_depth: Denotes the maximum depth of a tree. Larger values result in more complex models and increase the risk of overfitting. Must be optimized along with the eta value.

6- min_child_weight: Denotes the minimum sum of instance weight that a child requires. If the sum of instance weights appears to be less than this value in a tree partitioning step, then further partitioning is stopped. Larger values result in a more conservative algorithm (XGBOOST, 2020).

7- subsample: Ratio of subsampling of the training instances. Used to prevent overfitting (XGBOOST, 2020).

8- colsample_bytree: Denotes the ratio of subsampling while building each tree in the algorithm (XGBOOST, 2020).

9- nrounds: Number of boosting rounds.

5.3.4. Results

Unlike GBM and RF models, an Automated Grid Search was applied in order to tune 7 hyperparameters (gamma was held constant due to the computational complexity). As presented in Table 5.8, results indicate that highest prediction accuracy on the Test Set for using no data spans is 83.43% and with hyperparameter values of:
eta = 0.1, max_depth = 10, min_child_weight = 5, colsample_bytree = 0.7, subsample = 0.5, gamma = 0 and nrounds = 100

eta	gamma	max_depth	min_child_weight	subsample	colsample_bytree	nrounds	ACCURACY
0.1	0	10	5	0.5	0.7	100	83.43%

Table 5.8 Grid Search Results for Hyperparameter Tuning Combinations in XGB (No Data Span).

To check the consistency of 83.43% accuracy, the same model was applied using different data spans, as previously done in GBM, LR, and RF. Results presented in Table 5.9 have an arithmetic mean of 90.71% and a standard deviation of 0.89%. Therefore, we can conclude that using XGB can predict *SPYRETDIR6* for 5 different data spans with a mean value of 90.71% accuracy, which is slightly better than the mean value of RF and slightly worse than GBM results (90.60% and 91.07%, respectively). Very low standard deviation indicates that the results are consistent over time. Again, similar to the GBM and RF models, prediction accuracy slightly increases when data spans are introduced in the model.

DATA SPAN	ACCURACY (HIT RATE)
1	90.48%
2	91.07%
3	89.88%
4	89.88%
5	92.26%
MEAN	**90.71%**
STD. DEV	**0.89%**

Table 5.9 XGB Results – Arithmetic Mean and Variance of Accuracy for Different Data Spans.

DATA SPAN	Precision	Recall (Sensitivity)	F1 Score	Hit Rate
1	94.79%	89.22%	91.92%	90.48%
2	94.17%	91.51%	92.82%	91.07%
3	93.94%	89.42%	91.63%	89.88%
4	93.00%	90.29%	91.63%	89.88%
5	92.47%	93.48%	92.97%	92.26%
MEAN	**93.68%**	**90.78%**	**92.19%**	**90.71%**
STD. DEV	**0.83%**	**1.57%**	**0.59%**	**0.89%**

Table 5.10 XGB Performance Metrics for All Data Spans.

Regarding performance metrics given in Table 5.10, on average, 93.68% of the positive returns on SPY detected by our XGB model were true positives in the Test Set (Precision). Similarly, our XGB model correctly predicted 90.78% of the positives in the Test Set (Recall) and the harmonic mean of them is 92.19% (F1 Score).

In terms of variable importance, “xgb.importance” and “xgb.plot.importance” functions were applied. It was found that 4 out of 10 important variables provided by XGB results, were coincident with the important variables provided earlier by GBM and RF. Among those, 3 of them are technical indicators (*MOMENTUM5*, *STOCHASTICK*, and *RSI9*), while 1 of them is a fundamental variable (*DJIA1WR*). Using the randomly chosen number of variables for different data spans, there is evidence supporting that XGB makes use of technical indicators rather than fundamental variables in order to predict *SPYRETDIR6*.

5.4. ML Algorithm 4 – Logistic Regression

5.4.1. Steps for Calculating LR Results

Logistic regression results consist of 3 main steps:

1- According to the variable importance functions of the ensemble methods, 9 common independent variables were included in the first model with no data spans. Then, due to singularities and high multicollinearity (VIF>5), 3 of them were removed and the final model was trained on the selected 7 independent variables (OSCP, RSI9, GOLD1DR, DJIA1WR, MOMENTUM5, COIL1WR, and ADOSCILLATOR).

2- For every data span, Logistic Regression was run to predict *SPYRETDIR6*, using all selected independent variables.

3- Results are compared to each other, in terms of prediction accuracy and other performance metrics.

5.4.2. Procedure of Running LR in R

1- Using the “glm” function, we have performed Logistic Regression on the Train set, regressing *SPYRETDIR6* on selected 7 independent variables. We have also denoted family = “binomial” in the function, to point out that *SPYRETDIR6* is a binary categorical variable which takes either 1 (if the daily return is positive) or 0 (if the daily return is negative).

2- Then, using LR results, we have predicted *SPYRETDIR6* on the Test set.

3- To interpret the predicted probabilities, we have created a vector of 0’s, which has the same length as the Test set. Then, we have specified that if the predicted probabilities exceed 0.5, the prediction of stock market return should be denoted as 1, and 0 otherwise.

4- As the last step, we have created a confusion matrix to calculate the accuracy and other performance metrics.

5.4.3. Results

As presented in Table 5.11, results indicate that using no data spans Logistic Regression comes up with a prediction accuracy of 83.52%. However, introducing data spans improves the predictive power of the algorithm up to that of ensemble methods, results in an accuracy of 90.60% on average. Considering the arithmetic mean of prediction accuracies on all data spans, it is important to point out that Logistic Regression performs as good as RF (90.60%) and performs slightly worse than GBM and XGB (GBM: 91.07% and XGB: 90.71%). When it comes to the consistency of these results, LR does a very good job having a standard deviation of 0.69% on average, proves that they are very consistent over time.

DATA SPAN	ACCURACY (HIT RATE)	NO DATA SPAN
1	91.07%	83.52%
2	89.88%	
3	90.48%	
4	89.88%	
5	91.67%	
MEAN	**90.60%**	
STD. DEV	**0.69%**	

Table 5.11 LR Results – Arithmetic Mean and Variance of Accuracy for No Data Spans / Different Data Spans.

Regarding the mean of the performance metrics given in Table 5.12, 93.50% of the returns on SPY that were predicted by our model as positive, were true positives (Precision). Similarly, 90.72% of the true positives in the Test Set were predicted correctly (Recall) and the Harmonic mean of Precision and Recall is 92.09% (F1 Score). Finally, the overall accuracy is 90.60% (Hit Rate).

DATA SPAN	Precision	Recall (Sensitivity)	F1 Score	Hit Rate
1	94.79%	90.10%	92.39%	91.07%
2	93.20%	90.57%	91.87%	89.88%
3	93.94%	90.29%	92.08%	90.48%
4	92.00%	91.09%	91.54%	89.88%
5	93.55%	91.58%	92.55%	91.67%
MEAN	**93.50%**	**90.72%**	**92.09%**	**90.60%**
STD. DEV	**0.92%**	**0.54%**	**0.36%**	**0.69%**

Table 5.12 LR Performance Metrics for All Data Spans.

5.5. Comparison of 4 ML Algorithms for All Data Spans

DATA SPAN	GBM	RF	LR	XGB
1	91.07%	90.48%	91.07%	90.48%
2	90.48%	89.29%	89.88%	91.07%
3	92.26%	93.45%	90.48%	89.88%
4	91.07%	89.88%	89.88%	89.88%
5	90.48%	89.88%	91.67%	92.26%
MEAN	**91.07%**	**90.60%**	**90.60%**	**90.71%**
SD	**0.65%**	**1.48%**	**0.69%**	**0.89%**

Table 5.13 Comparison of 4 ML Algorithms for All Data Spans.

As given in Table 5.13; regardless of which algorithm is applied to predict *SPYRETDIR6*, we come up with a mean accuracy of 90.75% (average of the averages of all algorithms) and with an average standard deviation of 0.93%. Also, for different data spans different algorithms work better, but their results are nearly identical to each other. These results point out that there is neither one best algorithm to predict the market returns, nor one best model. Additionally, the results for all algorithms show improvement when data spans are used indicating the evolving nature of SPY returns series. *SPYRETDIR6* can consistently (SD values in a range of 0.65% - 1.48%) be predicted with historical data using ML techniques with significant accuracy (Mean values in a range of 90.60% - 91.07%).

6. MAGNITUDE PREDICTION OF SPY USING RANDOM FOREST

As magnitude prediction of SPY is a regression problem, we have applied Random Forest to the same dataset using the same methodology of predicting *SPYRETDIR6* after making 2 changes:

1- Instead of using *SPYRETDIR6*; we have used the daily adjusted closing prices of SPY as the dependent variable (*SPYMAG*), as we are interested in predicting the magnitude of prices instead of direction.

2- Due to the computational complexity of applying Grid search for hyperparameter tuning in Random Forest, we have only tuned mtry using no data spans and used the same hyperparameter values for all data spans. Final hyperparameter values are as follows:

mtry = 10, ntree = 100, max_features = ”Auto”, max_depth = 10, min_samples_split = 10, min_samples_leaf = 10, importance = TRUE

DATA SPAN	CUMULATIVE NET PROFIT/LOSS WHEN “BUY” ($)	CUMULATIVE NET PROFIT/LOSS WHEN “SELL” ($)
NO DATA SPAN	281.61	-618.48
1	59.20	-319.51
2	6.58	-396.04
3	137.07	-289.51
4	-59.23	-433.88
5	150.20	-522.70
MEAN[1]	**58.76**	**-392.33**
STD. DEV[2]	**88.23**	**93.02**

Table 6.14 Net Cumulative Profit of Trading Strategy #1 (*SPYRETDIR6*) – Investing Only When It Signals “BUY” or “SELL” – All Data Spans.

Notes: 1. “No Data Span” results are excluded while calculating mean.
2. “No Data Span” results are excluded while calculating standard deviation.

DATA SPAN	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)	SHARPE RATIO
1	10000	-75.15	-0.75	2.55	-308.99
2	10000	152.59	1.53	2.57	-97.85
3	10000	347.55	3.48	2.57	83.21
4	10000	104.87	1.05	2.57	-139.06
5	10000	531.23	5.31	2.53	243.83
MEAN		**212.22**	**2.12**	**2.56**	**-43.77**
STD. DEV		**233.31**	**2.33**	**0.02**	**212.83**

Table 6.15 Results of Buy & Hold Strategy (*SPYRETDIR6*) – All Data Spans.

As presented in Figures 6.1 - 6.6 and Tables 6.14 - 6.17, actual / predicted values of SPY are highly correlated with each other, with an average correlation coefficient of 0.89 for 5 data spans. It is also interesting to note that the correlation coefficient declines to 0.15 using no data spans. Therefore, analyzing these results point out that our Random Forest model can predict

SPYMAG for the short-term with high accuracy, however it is useless for long-term predictions. Section 7.3 provides a guideline for a trading strategy to check the results.

DATA SPAN	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)	SHARPE RATIO
1	10000	-260.31	-2.60	2.55	-482.53
2	10000	-389.46	-3.89	2.57	-605.42
3	10000	-152.43	-1.52	2.57	-378.15
4	10000	-493.11	-4.93	2.57	-687.31
5	10000	-372.50	-3.73	2.53	-547.99
MEAN		**-333.56**	**-3.34**	**2.56**	**-540.28**
STD. DEV		**130.64**	**1.31**	**0.02**	**117.84**

Table 6.16 Results of Trading Strategy #1 (*SPYRETDIR6*) – All Data Spans.

DATA SPAN	CORRELATION COEFFICIENT
1	0.80
2	0.88
3	0.93
4	0.94
5	0.92
MEAN	**0.89**

Table 6.17 Correlation Coefficients of SPY (Actual vs. Predicted) – All Data Spans.

Figure 6.1 Actual vs. Predicted Values of SPY – No Data Spans.

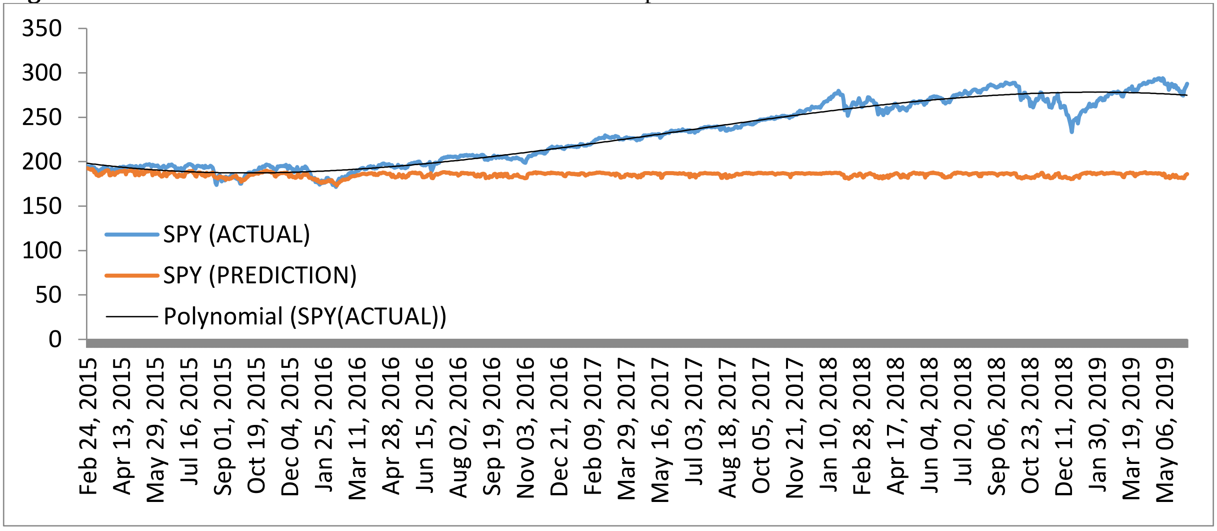


Figure 6.2 Actual vs. Predicted Values of SPY – 1st Data Span.

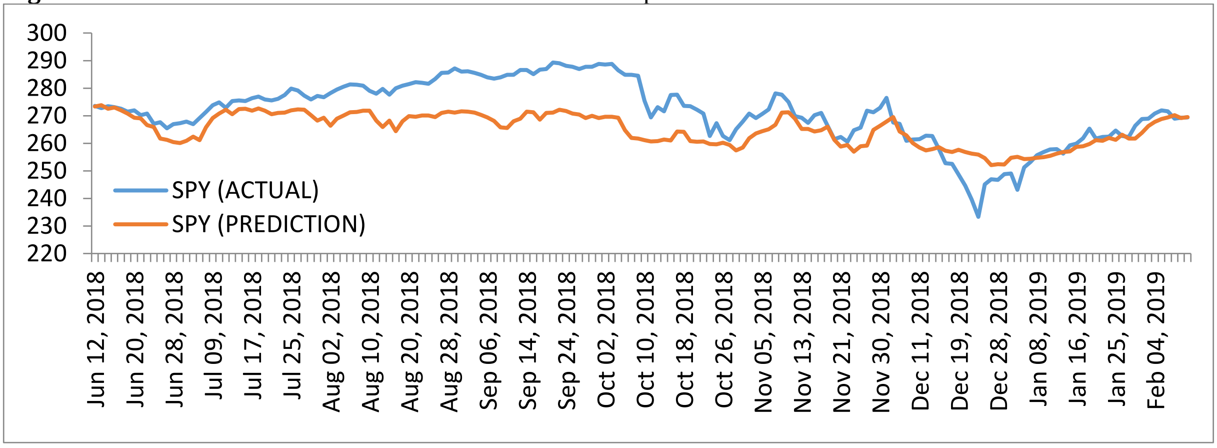


Figure 6.3 Actual vs. Predicted Values of SPY – 2nd Data Span.

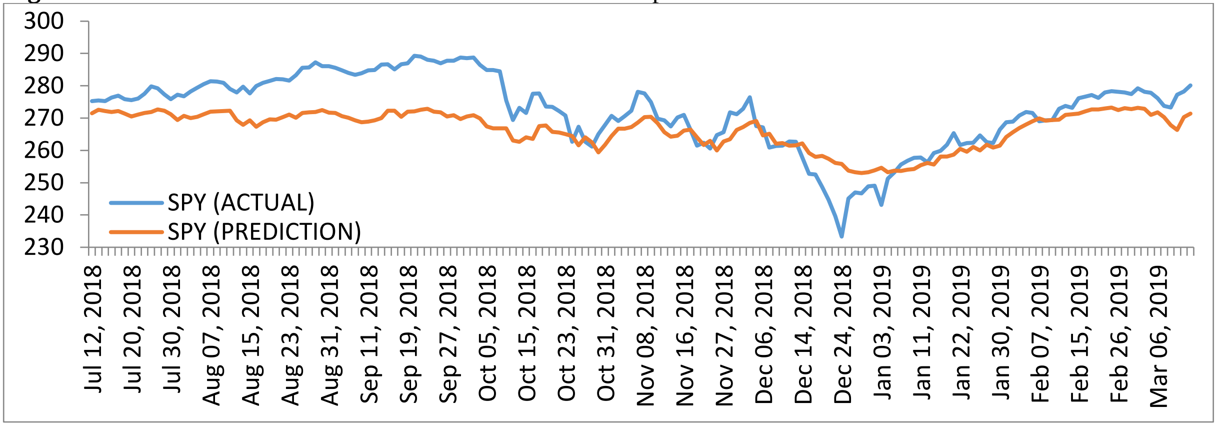


Figure 6.4 Actual vs. Predicted Values of SPY – 3rd Data Span.

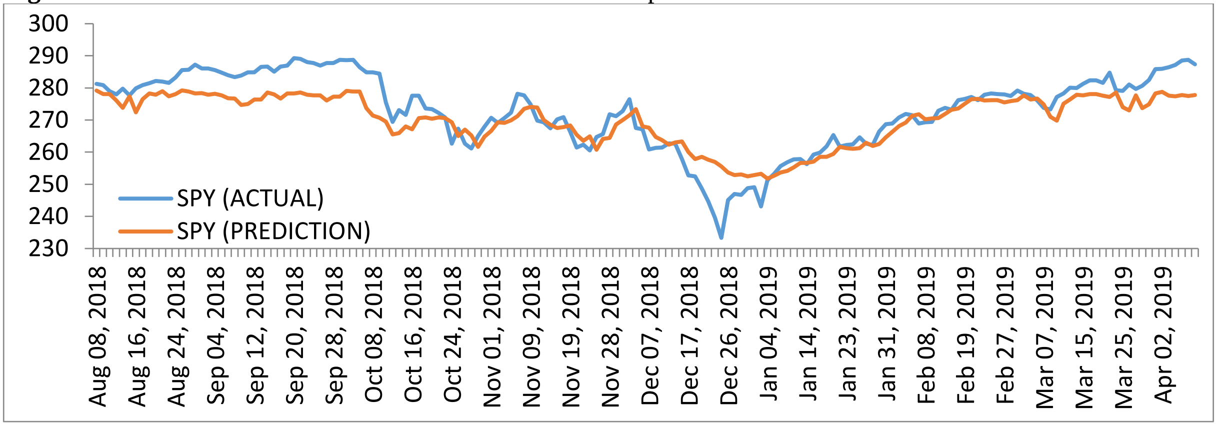


Figure 6.5 Actual vs. Predicted Values of SPY – 4th Data Span.

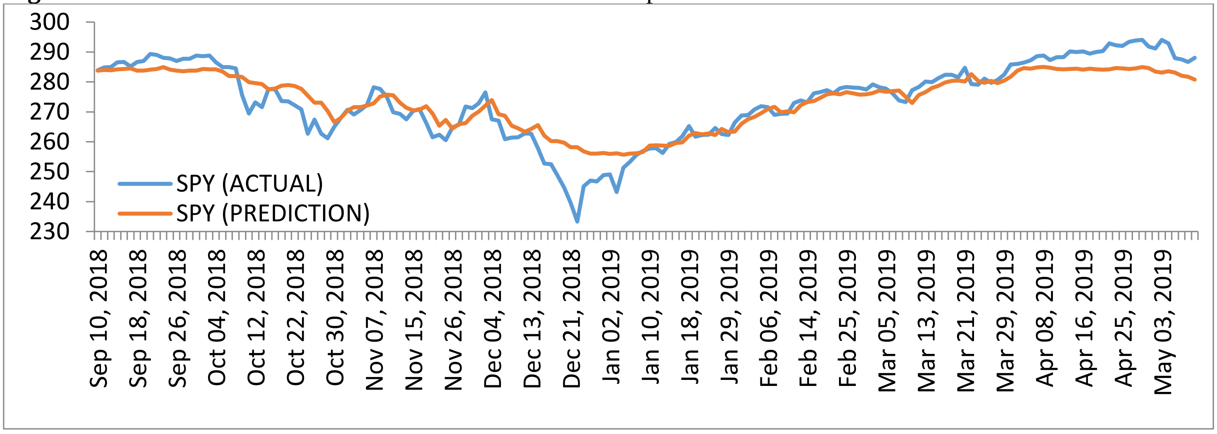
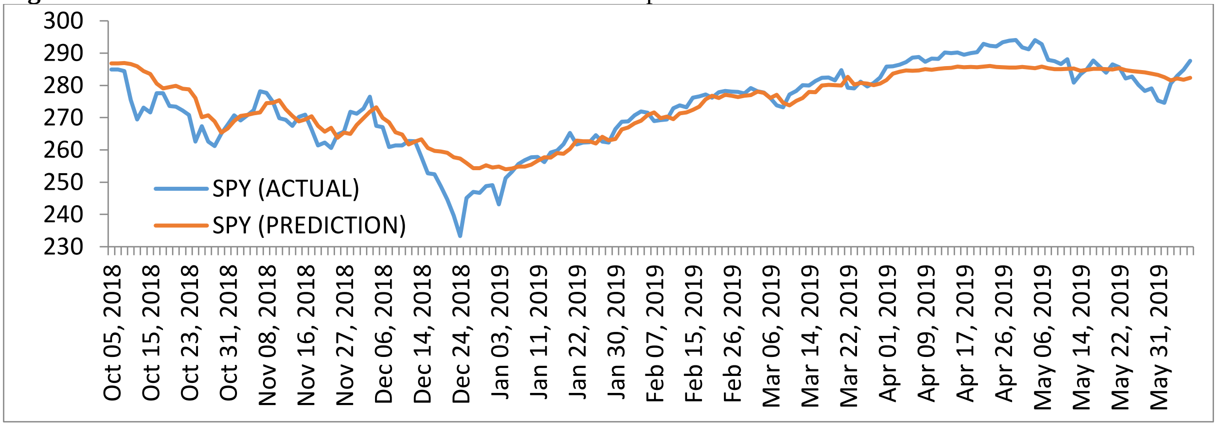


Figure 6.6 Actual vs. Predicted Values of SPY – 5th Data Span.



7. ML ALGORITHM DRIVEN TRADING STRATEGIES

7.1. Buy & Hold Strategy

As a benchmark, we are using a simple Buy & Hold strategy to compare its performance with the ML-driven trading strategies developed in this study. The Buy & Hold strategy can be simply explained as follows:

1- A day before the first day of the beginning of the first data span, open a long position using the entire initial investment at the adjusted closing price of the stock/index.

2- Hold your position throughout the data span and only on the last day of that data span, close your position using the adjusted closing price announced on that day.

7.1.2 Performance of the Buy & Hold Strategy on SPYMAG and SPYRETDIR6

SPYMAG

Without using any data spans; applying the Buy & Hold strategy with an initial investment of 10,000 USD for 1080 trading days (roughly 4.2 years) results in a cumulative net profit of 4,772.49 USD at the maturity. As the average 1-Year T-bill yield in 2015 with a maturity of 3 and 5 years is 1.28%, it signals that investing in T-bills using the risk-free rate is less profitable than investing in SPY using the Buy & Hold strategy.

However, if we divide our Test set into 5 subsets to catch the interest of short-term investors as well as to check the consistency of our results, we come up with a cumulative net profit of 96.45 USD and a yield of 0.96% at the maturity, on average. As provided in Table 7.18, it is important to point out that Buy & Hold strategy provides positive returns separately for all data spans except the first. However, T-bill yields are greater than our returns for all data spans, which results in negative Sharpe ratios. Thereupon, given the notably high standard deviation (145.73), investing in SPY using Buy & Hold strategy is less profitable and more inconsistent than investing in T-bills using the risk-free rate.

DATA SPAN	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)	SHARPE RATIO
1	10000	-152.77	-1.53	2.55	-381.74
2	10000	175.46	1.75	2.57	-76.44
3	10000	214.38	2.14	2.57	-39.67
4	10000	148.30	1.48	2.57	-99.25
5	10000	96.88	0.97	2.53	-136.74
MEAN		**96.45**	**0.96**	**2.56**	**-146.77**
STD. DEV		**145.73**	**1.46**	**0.02**	**136.00**

Table 7.18 Results of Buy & Hold Strategy (*SPYMAG*) – All Data Spans.

SPYRETDIR6

Using no data spans, the Buy & Hold strategy results in a cumulative net profit of 4821.21 USD and a yield of 48.21%. As the average 1-Year T-bill yield is 1.28%, the Buy & Hold strategy has a very high Sharpe ratio and therefore is very profitable compared to investing in T-bills using the risk-free rate.

To catch the attention of short-term investors as well as checking the consistency of these results, 5 data spans are introduced. Then, the average net cumulative profit becomes 212.22 USD with a standard deviation of 233.31. However, as the average T-bill yield happens to be 2.56%, the Sharpe ratio becomes negative. Therefore, investing in T-bills using the risk-free rate is more profitable than applying the Buy & Hold strategy. In fact; Table 6.15 shows that this strategy provides positive returns for all data spans separately, except for the first one. But the results are inconsistent due to the very high standard deviation.

7.2. Trading Strategy #1

The following steps provide a guideline on how to implement this research, based solely on the prediction of *SPYRETDIR6* direction, into an algorithmic trading strategy:

1- Train the model using the methodology provided in this research on an index or individual stock of your choice.

2- Start generating (t+6)th trading day prediction for every day.

3- On (t+6)th day, if the opening price of the stock is lower (higher) than the closing of (t)th day and the model predicted positive (negative) direction, then open a long (short) position to be closed before the end of that trading day.

Using no data spans, applying this trading strategy to the Test set results in a cumulative net loss of 336.87 USD for 10,000 USD initial investment. Due to the fact that we did not use the magnitude prediction of the adjusted closing prices of SPY, our trading algorithm allows us to trade only 128 days.

In order to check the consistency of results as well as how the trading algorithm works for shorter periods, we have used 5 consecutive data spans. Results point out that on average, we come up with a cumulative net loss of 333.56 USD with an initial investment of 10,000 USD (Table 6.16). Similarly, without predicting the magnitude of the adjusted closing prices of SPY, we are restricted to trade 14 days out of 162, on average.

It is important to note that as shown in Table 6.14, Trading Strategy #1 results in a cumulative net profit of 58.76 USD if the investment is made solely when the strategy provides "BUY" signals. However, the same does not occur with "SELL" signals. As the actual price movements of SPY is depicted in Figure 6.1, it follows a random walk with a slight upward trend (which is captured by using a 3rd-degree polynomial trend). Therefore, as the prices are expected to rise in the long term, it is reasonable that investing only in "BUY" signals work.

7.3. Trading Strategy #2

The following steps provide a guideline on how to implement this research on magnitude prediction of SPY into an algorithmic trading strategy:

1- Train the model using the methodology provided in this research on an index or individual stock of your choice.

2- Start generating (t+1)st trading day magnitude prediction for every day.

3- One day before the first trading day, open a long position of the stock or index of your preference using all the capital and the adjusted closing price of the same day.

4- On (t)th day, if the adjusted opening price of the stock on (t)th day is lower (higher) than the predicted closing of (t)th day, then open a long (short) position using the adjusted opening price of (t)th day to be closed before the end of that trading day.

Without using any data spans, if we apply trading strategy #2 instead, we would be making a cumulative net loss of 667.67 USD. As the average T-bill yield with a maturity of 1 year is 1.28%, investing in the bond market using the risk-free rate is more profitable than investing in SPY using our strategy.

On the other hand, applying trading strategy #2 to all data spans provides us a cumulative net profit of 218.35 USD and a yield of 2.18% at the maturity, on average (Table 7.19). As the average T-bill yield with a maturity of 1 year is 2.56%; even though our trading strategy provides positive returns on average, investing in the bond market using the risk-free rate is slightly more profitable.

DATA SPAN	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)	SHARPE RATIO
1	10000	1611.65	16.12	2.55	12.72
2	10000	289.64	2.90	2.57	0.30
3	10000	-56.14	-0.56	2.57	-2.89
4	10000	187.18	1.87	2.57	-0.64
5	10000	-940.59	-9.41	2.53	-10.46
MEAN		**218.35**	**2.18**	**2.56**	**-0.19**
STD. DEV		**917.31**	**9.17**	**0.02**	**8.36**

Table 7.19 Results of Trading Strategy #2 – All Data Spans.

7.4. Comparison of Buy & Hold vs. Trading Strategy #1

As presented in Table 7.20, the Buy & Hold strategy significantly outperforms Trading strategy #1, when no data spans are used (with a total yield of 48.21% and -3.37%, respectively). Results provide clear evidence that investing in SPY after solely predicting its direction provides negative returns and therefore is not profitable.

STRATEGY	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)
BUY & HOLD	10000	4821.21	48.21	1.28
TRADING STRATEGY #1	10000	-336.87	-3.37	1.28

Table 7.20 Comparison of Buy & Hold (*SPYRETDIR6*) vs. Trading Strategy #1 – No Data Span.

STRATEGY	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)
BUY & HOLD	10000	212.22	2.12	2.56
TRADING STRATEGY #1	10000	-333.56	-3.34	2.56

Table 7.21 Comparison of Buy & Hold (*SPYRETDIR6*) vs. Trading Strategy #1 – All Data Spans, Average n.

Introducing 5 data spans does not change the results much. This time using Buy & Hold strategy, we come up with a net cumulative profit of 212.22 USD on average, while investing using Trading strategy #1 still results in a total cumulative loss of 333.56 USD (Table 7.21).

Another important finding is that if the Buy & Hold strategy is applied without any data spans, due to the very high Sharpe ratio, it is almost 38 times more profitable than investing in T-bills using the risk-free rate. However, this is not the case when data spans are included. Then, investing in T-bills using the risk-free rate becomes slightly more profitable (0.83 times) than applying the Buy & Hold strategy. However, no matter if data spans are introduced or not, it is empirically proven that investing in T-bills using the risk-free rate is always more profitable than applying Trading Strategy #1 (Table 7.20 and 7.21).

7.5. Comparison of Buy & Hold vs. Trading Strategy #2

As presented in Table 7.22, the Buy & Hold strategy significantly outperforms Trading Strategy #2 when no data spans are used (with a total yield of 47.72% and -6.68%, respectively). This is mainly because our model is naïve when it comes to predicting the magnitude of SPY for the long term. Figure 6.1 shows how the actual and predicted prices of SPY change along with time. As the model is unable to capture the changing market dynamics for the long term, either the hyperparameters should be tuned more conservatively, or a more complex trading strategy should be proposed.

STRATEGY	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)
BUY & HOLD	10000	4772.49	47.72	1.28
TRADING STRATEGY #2	10000	-667.67	-6.68	1.28

Table 7.22 Comparison of Buy & Hold (*SPYMAG*) vs. Trading Strategy #2 – No Data Span.

Interestingly, results drastically change when data spans are introduced to the model. After applying 5 different data spans to the model, it is empirically proved that Trading Strategy #2 outperforms the Buy & Hold strategy with a total yield of 2.18% and 0.96%, respectively (Table 7.23). These results are also supported by Figure 6.2-6.6 and Table 6.17, in which eyeballing suggests that the predicted vs. Actual prices of SPY are procyclical. In a more statistical sense, Table 6.17 shows that the arithmetic means of the correlation coefficient of predicted vs. actual prices of SPY for 5 different data spans is 0.89.

STRATEGY	INITIAL INVESTMENT ($)	ACTUAL GAIN/LOSS ($)	YIELD (%)	T-BILL YIELD (1 YEAR, %)
BUY & HOLD	10000	96.45	0.96	2.56
TRADING STRATEGY #2	10000	218.35	2.18	2.56

Table 7.23 Comparison of Buy & Hold (*SPYMAG*) vs. Trading Strategy #2 – All Data Spans, Average.

However, there is another "safe" option for investors: Investing in T-bills using the risk-free rate. In order to conclude whether an investor can actually better off investing in the market using either Buy & Hold or Trading Strategy #2, we have broadened our analysis by calculating Sharpe ratios for each of them.

In contrast to that, introducing different data spans pointed out surprising results. First of all, even though the cumulative net profits on average for both strategies are positive, they are lower than the average T-bill yields. Therefore, the results point out that investing in SPY using Trading Strategy #2 is less profitable than investing in T-bills using the risk-free rate. Also, even though separately considering some data spans (e.g. 1st and 2nd ones) using Trading

Strategy #2 provide positive Sharpe ratios and therefore being more profitable than investing in T-bills; the consistency of this profitability is quite low, due to the high standard deviation.

CONCLUSION

This study aimed at developing stock market trading strategies using powerful machine learning algorithms using only the most basic theoretically validated input variables. The idea was to assess whether ML algorithms have become sophisticated enough over time to guarantee returns higher than the risk-free rate of return. For this purpose, state-of-the-art ML algorithms were used to predict the direction of the S&P 500 Index along with its magnitude, using only a few basic and technical indicators.

The results indicated that ML algorithm-driven trading strategies are inconsistent and have either low or negative Sharpe ratios. The strategies become quite risky as the traditional ML algorithms are static whereas the nature of financial markets is dynamic. Therefore, they do not respond to the fluctuations in the market. On average, the ML algorithm-driven strategies are profitable, but the profits do not match with the returns provided by risk-free returns. This implies that even with the most celebrated techniques of machine learning, profits higher than risk-free return cannot be generated consistently without human intervention.

APPENDIX

A.1 Variables Selection (Full Set)

A.1.1. Fundamental Variables

1- **USDEUR:** USD/EURO Foreign Exchange Rate (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
2- **USDGBP:** USD/GBP Foreign Exchange Rate (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
3- **CNYUSD:** CNY/USD Foreign Exchange Rate (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
4- **GOLD:** Gold Fixing Price in London Bullion Market. Based in USD, per Ounce (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
5- **GOLD1DR:** Daily return of GOLD. Calculated in Excel.
6- **GOLD1WR:** Weekly (5-trading days) return of GOLD. Calculated in Excel.
7- **COIL:** Crude Oil Prices. West Texas Intermediate (WTI) - Cushing, Oklahoma. USD per Barrel (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
8- **COIL1DR:** Daily return of COIL. Calculated in Excel.
9- **COIL1WR:** Weekly (5-trading days) return of COIL. Calculated in Excel.
10- **MLIBOR:** 1-Month LIBOR (%) (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
11- **DJIA:** Dow Jones Industrial Average (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
12- **DJIADR:** Daily return of DJIA. Calculated in Excel.
13- **DJIA1WR:** Weekly (5-trading days) return of DJIA. Calculated in Excel.
14- **WS5000:** Wilshire 5000 Total Market Full Cap Index (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).
15- **WS5000DR:** Daily return of WS5000. Calculated in Excel.

16- **NYSE:** Adjusted Closing Prices of New York Stock Exchange Composite. Data was extracted from Yahoo Finance (YAHOO, 2019a).
17- **VIX:** CBOE Volatility Index (Daily). Data were extracted from the Chicago Board Options Exchange (CBOE, 2020) and Investing (2019).
18- **VIXDR:** Daily return of VIX. Calculated in Excel.
19- **NSDQ100:** Nasdaq 100 Index (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019) and Investing (2019).
20- **EPUIUS:** Economic Policy Uncertainty Index for U.S. (Daily). Data were extracted from the Federal Reserve Bank of St. Louis (FRED, 2019).

A.1.2. Technical Variables

1- **RSI14:** Relative Strength Index for the last 14 trading days. It measures how much and fast price movements change. Ranges between 0-100. Values above 70 give overbought signals, while values below 30 give oversold signals (Fidelity, n.d.). Calculated in R using the "TTR" package.
2- **RSI9:** Relative Strength Index for the last 9 trading days. Calculated in R using the "TTR" package.
3- **SMA5:** Simple Moving Average for the last 5 trading days. Calculated in R using the "TTR" package.
4- **SMA10:** Simple Moving Average for the last 10 trading days. Calculated in R using the "TTR" package.
5- **SMA20:** Simple Moving Average for the last 20 trading days. Calculated in R using the "TTR" package.
6- **EMA10:** Exponential Moving Average for the last 10 trading days. Calculated in R using the "TTR" package.
7- **EMA20:** Exponential Moving Average for the last 20 trading days. Calculated in R using the "TTR" package.
8- **EMA30:** Exponential Moving Average for the last 30 trading days. Calculated in R using the "TTR" package.
9- **STOCHASTICK:** Compares the adjusted closing price of SPY relative to its price range over 14 trading days (Kim, 2003)**.** Calculated in Excel. Formula is:

$$\frac{Adjusted\ Closing\ Price_t - Adjusted\ Lowest\ Low_{14}}{Adjusted\ Highest\ High_{14} - Adjusted\ Lowest\ Low_{14}} * 100$$

10- **STOCHASTICD:**3-days moving average of STOCHASTICK (Kim, 2003)**.** Calculated in Excel.
11- **ADOSCILLATOR:** Accumulation/distribution oscillator. It is used for detecting price changes (Kim, 2003). Calculated in Excel. Formula is:

$$\frac{Adjusted\ High\ Price_t - Adjusted\ Closing\ Price_{t-1}}{Adjusted\ High\ Price_t - Adjusted\ Low\ Price_t}$$

12- **ROC:** Rate of change in prices. Measures the difference between the Current Adjusted Closing Price and Yesterday's Adjusted Closing Price (Kim, 2003). Calculated in Excel. Formula is:

$$\frac{Adjusted\ Closing\ Price_t}{Adjusted\ Closing\ Price_{t-1}} * 100$$

13- **ROC3:** Rate of change in prices. Current Adjusted Closing Price vs. Adjusted Closing Price of 3 trading days ago. Calculated in Excel
14- **MOMENTUM:** Measures how much the Adjusted Closing Price has changed in 1 trading day (Kim, 2003). Calculated in Excel. Formula is:

$$Adjusted\ Closing\ Price_t - Adjusted\ Closing\ Price_{t-1}$$

15- **MOMENTUM 5:** Measures how much the Adjusted Closing Price has changed in 5 trading days. Calculated in Excel.

16- **MOMENTUM15:** Measures how much the Adjusted Closing Price has changed in 15 trading days. Calculated in Excel.

17- **MOMENTUM30:** Measures how much the Adjusted Closing Price has changed in 30 trading days. Calculated in Excel.

18- **OSCP:** Price oscillator. Measures the difference between SMA10 and SMA5 (Kim, 2003). Calculated in Excel. Formula is:

$$1 - \frac{SMA10}{SMA5}$$

19- **CCI:** Commodity Channel Index. Measures how much the Adjusted Closing Price of a security varies from its statistical mean (Kim, 2003). Calculated in R using the “TTR” package.

20- **LARRYWILLIAMSR:** Larry Williams’s R%. Used to detect overbought/oversold levels of a security (Kim, 2003). Calculated in Excel. Formula is:

$$\frac{Adjusted\ High\ Price_{14} - \ Adjusted\ Closing\ Price_{t}}{Adjusted\ High\ Price_{14} - \ Adjusted\ Low\ Price_{14}} * 100$$

REFERENCES

Atsalakis, G.S. and K.P. Valavanis, (2009). Surveying stock market forecasting techniques–Part II: Soft computing methods. *Expert Systems with Applications*, 36 (3), 5932-5941.

Ballings, M., D. Van den Poel, N. Hespeels and R. Gryp (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, 42 (20), 7046-7056.

CBOE. (2020). VIX Index [Data File]. Retrieved 09 22, 2019, from http://www.cboe.com/chardata/GetDownloadSymbolData/?Symbol=VIX.

Diler, A.I. (2003). Forecasting the Direction of the ISE National-100 Index By Neural Networks Backpropagation Algorithm. *Istanbul Stock Exchange Review*, 7 (25-26), 65-82.

Ding, S., H. Zhao, Y. Zhang, X. Xu and R. Nie (2015). Extreme learning machine: algorithm, theory and applications. *Artificial Intelligence Review*, 44 (1), 103-115.

Egeli, B., M. Ozturan and B. Badur (2003). Stock Market Prediction Using Artificial Neural Networks. *Decision Support Systems*, 22, 171-185.

Fidelity. (n.d.). Relative Strength Index (RSI). Retrieved September 26, 2019, from https://www.fidelity.com/learning-center/trading-investing/technical-analysis/technical-indicator-guide/RSI.

FRED. (2019). Retrieved 09 27, 2019, from Federal Reserve Bank of St.Louis Economic Research: https://fred.stlouisfed.org/.

Gradient Boosting Machines. (2018). Retrieved 09 22, 2019, from https://uc-r.github.io/2018/06/14/gbm-regression/.

Gündüz, H., Z. Çataltepe and Y. Yaslan (2017). Stock daily return prediction using expanded features and feature selection. *Turkish Journal of Electrical Engineering & Computer Sciences*, 25 (6), 4829-4840.

Imandoust, S.B. and M. Bolandraftar (2014). Forecasting the direction of stock market index movement using three data mining techniques: the case of Tehran Stock Exchange. *International Journal of Engineering Research and Applications*, 4 (6), 106-117.

Investing. (2019). Retrieved 09 22, 2019, from https://www.investing.com/.

Kara, Y., M.A. Boyacioglu and Ö.K. Baykan (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38 (5), 5311-5319.

Kim, K.J. (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, 55 (1-2), 307-319.

Kumar, M. and M. Thenmozhi (2006). Forecasting Stock Index Movement: A Comparision of Support Vector Machines and Random Forest. *Indian Institute of Capital Markets 9th Capital Markets Conference Paper*.

Labiad, B., A. Berrado and L. Benabbou (2016). Machine learning techniques for short term stock movements classification for moroccan stock exchange. In 2016 *11th International Conference on Intelligent Systems: Theories and Applications (SITA)* (pp. 1-6). IEEE.

Michalski, R.S. and Y. Kodratoff (1990). Research in machine learning: Recent progress, classification of methods, and future directions. In *Machine learning* (pp. 3-30). Morgan Kaufmann.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel and J. Vanderplas (2011). Scikit-learn: Machine learning in Python. *The journal of machine learning research*, 12, 2825-2830.

Ridgeway, G. (2007). Generalized Boosted Models: A guide to the gbm package. *Update*, 1 (1), 2007.

Rodriguez, P.N. and A. Rodriguez (2004). Predicting stock market indices movements. *WIT Transactions on Modelling and Simulation, 38.*

Sunasra, M. (2017). Performance metrics for classification problems in machine learning. *Medium*. Retrieved 09 22, 2019, from https://medium.com/thalus-ai/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b.

Wüthrich, B., D. Permunetilleke, S. Leung, W. Lam, V. Cho and J. Zhang (1998). Daily prediction of major stock indices from textual www data. *Hkie transactions*, 5 (3), 151-156.

XGBOOST. (2020). XGBoost Parameters. Retrieved 09 22, 2019, from https://xgboost.readthedocs.io/en/latest/parameter.html.

YAHOO. (2019a). Retrieved September 27, 2019, from https://finance.yahoo.com/.

YAHOO. (2019b). Retrieved September 20, 2019, from https://finance.yahoo.com/quote/SPY/history?p=SPY.

Yoon, Y. and G. Swales (1991). Predicting stock price performance: A neural network approach. In *Proceedings of the twenty-fourth annual Hawaii international conference on system sciences* (Vol. 4, pp. 156-162). IEEE.

Zhai, Y., A. Hsu, and S.K. Halgamuge (2007). Combining news and technical indicators in daily stock price trends prediction. In *International symposium on neural networks* (pp. 1087-1096). Springer, Berlin, Heidelberg.

Zhang, Y. and L. Wu, (2009). Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network. *Expert Systems with Applications*, 36 (5), 8849-8854.