Predicting the Daily Directionality of Open Price of SSE Composite Index

Project for Udacity Machine Learning Engineer Nanodegree

I. Definition

Overview

Statistical models associated with machine learning algorithms are playing increasingly important roles in financial market analyses. In this study, the XGBoost classifier is used to predict daily directionality of the Open price of SSE composite index. An important benefit of XGBoost models is that there are several hyperparameters that can be tuned for improving accuracy. This study evaluates the performance of the XGBoost classifier on forecasting the daily directionality of the open price of SSE.

Problem Statement

The purpose of this project is to apply the XGBoost classifier into the China's stock market to determine if the XGboost classifier can be used to predict the directionality of Open Price of SSE Composite index (up or down). In this project, the label will be defined as "1" if the predicted Open price goes up relative to the Open price in the last day.

Metrics

There are basically two metrics used to evaluate the performance of the model. One of them are Accuracy Score, define by Accuracy Score = (True Positives + True Negatives) / Total Population. Another one is Net Profit generated by trading simulation. The reason for choosing this metric is that sometimes although the accuracy score is high, the trading outcome doesn’t suit people’s expectation. Therefore, the trading simulation is employed. Because China’s stock market doesn’t allow shorting position and require all investors to hold the stock at least one trading days, long position will be created if the predicted open price of SSE goes up.

II. Analysis

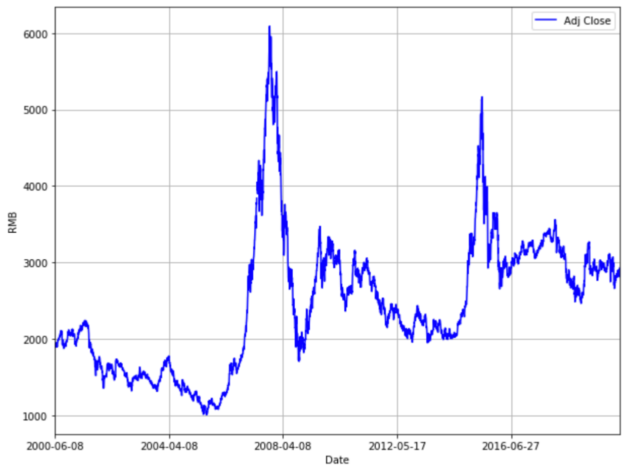
Data Exploration

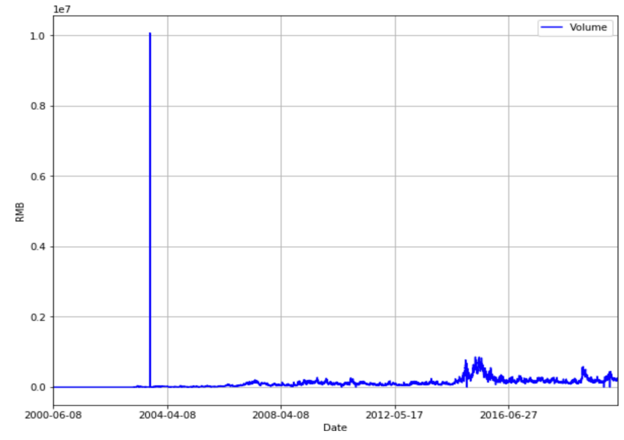
Exploratory data analysis (EDA), proposed by John Tukey, is a very important first step in data analysis (Tukey 1977). EDA can be used to determine the relationships among exploratory variables and correlation between explanatory variables and outcome variables. Exploratory can be classified in two ways, graphical or non-graphical method, univariate or multivariate method. The graphical method refers to the way that the summary statistics is calculated, and the non-graphical method refers to the way that the data is summarized in a diagrammatic way. Univariate methods refer to focusing on only one variable (data column) at a time, and multivariate methods explore two or more variables at a time to determine relationships. In this study, graphical and multivariate methods are used for feature selections (Seltman 2008).

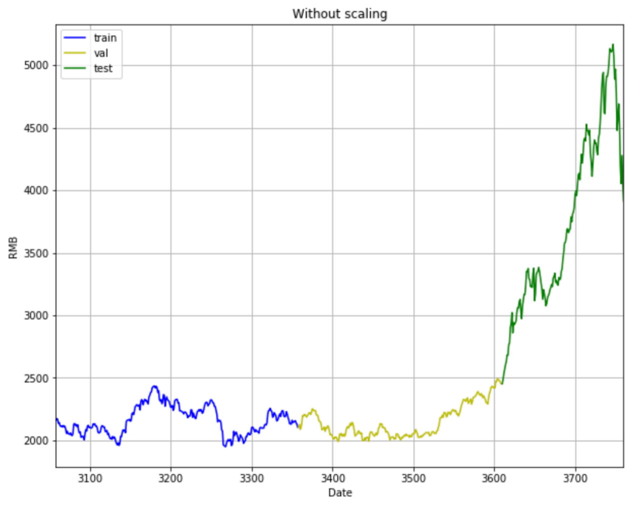
The data included in this study consist of daily directionality of open price of SSE as the output, along with 18 technical indicators as input features. The whole dataset consists of 700 trading days. All data used for this project comes from Yahoo Finance: The data is the historical data from 2000-06-08 to 2020-6-05. And the raw data is included in the file "000001.SS_new.csv". And it contains 6 indicators discussed below.

Date: calendar date for any given data row
Open: opening value (recorded at 9:30) for the SSE
High: highest value on any given day for the SSE
Low: lowest value on any given day for the SSE
Close: closing value (recorded at 16:00) for the SSE
Volume: number of shares of SSE components traded
The raw data consists of 4959 trading days.

Figure1: Data Exploration Visualization: SSE Composite Index

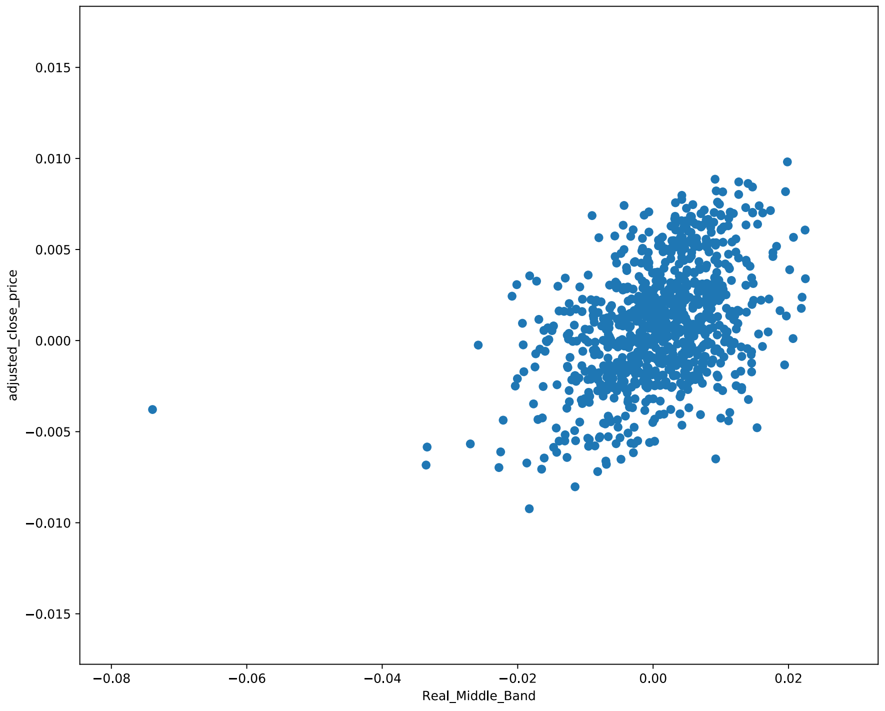






The following correlation figure shows that the relationship between the explanatory variable real middle band in Bollinger Band and the daily adjusted price of SSE is slightly positive, which means that the daily adjusted price increase as the real middle band goes up.

(Figure: correlation figure)



Algorithms and Techniques

XGBoost that implements the gradient boosting decision tree algorithm was first introduced by Chen and Guestrin (2016) from University of Washington. XGBoost, designed to be used with large and complex datasets, is an advanced version of the gradient boosting method. It is a decision-tree-based ensemble machine learning algorithm that combines multiple decision trees to a strong learner to enhance the performance and accuracy of machine learning model. Decision-tree-based algorithms usually outperforms other algorithms such as artificial neural network when it deals with tabular data. XGBoost can achieve very high accuracy but only requires less training time than other machine learning algorithms such as Gradient Boosting and Random Boosting (Morde 2019).

In this study, XGBoost Classifier is used for classfication, so decisions trees are employed.

The basic steps of training the dataset is to define the model and learn the functions. In the step of learning functions, we will define the objective and optimize it. The derivation of the mathematical formula was published by Chen and Guestrin (2016).

At first, we define the model and assume we have K trees for a given data set with n examples and m features.

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \ , f_k \epsilon \mathcal{F}, x_i \in \mathbb{R}^m \qquad (1)$$

where $\mathcal{F}$ is the space of functions containing all classfication trees,
x_i is in the set of features, and $\hat{y}_i$ is the prediction of the i-th instance.

And, then, define the objective by

$$\mathcal{L}(\phi) = \sum_i l\,(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \text{ , where } \Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2 \qquad (2)$$

l: a differentiable convex loss function, which is the difference between the predicted value $\hat{y}_i$ and the target y_i
Ω : a regularization function, which penalizes the complexity of the model

Our main goal is to find an optimal prediction that minimizes the objective function.

Benchmark

The year 2014-11-20 to year 2015-07-02 serves as the test set for the models. And SSE closed higher for about 58% trading days. Therefore, it is good to predict market will increase, which labels as 1, and it would generate the accuracy of 58%. These values will be the benchmark of this study.

III. Methodology

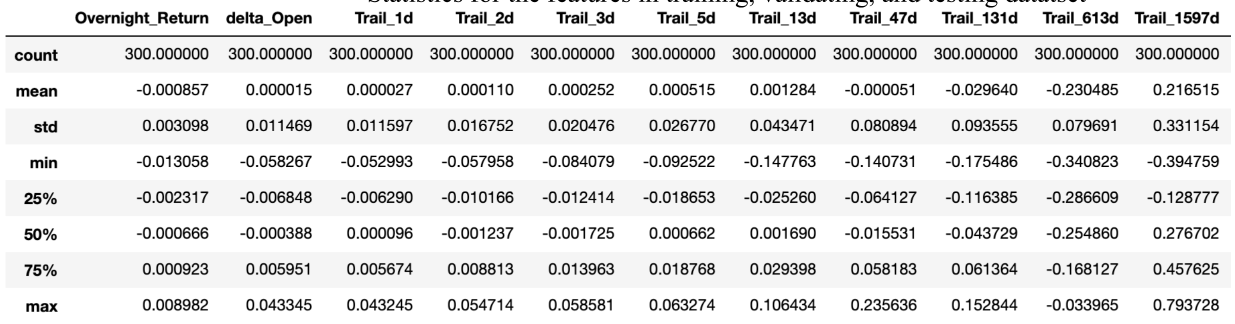
Data Preprocessing

The data included in this study consist of daily directionality of open price of SSE as the output, along with 18 technical indicators as input features. The whole dataset consists of 700 trading days. The features and technical indicators include overnight return, delta open, trailing return, trailing volume, Change from high to low, Open relative to Close and etc. Most of the features are "relative values", and the reason for choosing the relative value is that absolute value may not be very productive in predicting the directionality of the Open price.

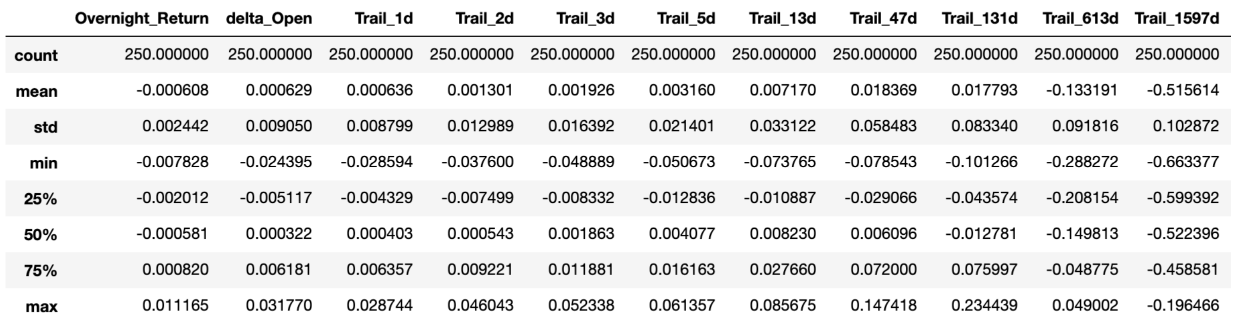
(Table 1: The 18 features of technical indicators)

Name	Description
Overnight_Return	Percentage of increase from adjusted closing price at day n-1 to open price to day n
delta_Open	Percentage increase of open price from day n-1 to day n
Trail_1d	Adjusted Closing price in n-1 relative to n-2
Trail_2d	Adjusted Closing price in n-1 relative to n-3
Trail_3d	Adjusted Closing price in n-1 relative to n-4
Trail_5d	Adjusted Closing price in n-1 relative to n-6
Trail_13d	Adjusted Closing price in n-1 relative to n-14
Trail_47d	Adjusted Closing price in n-1 relative to n-47
Trail_131d	Adjusted Closing price in n-1 relative to n-131
Trail_613d	Adjusted Closing price in n-1 relative to n-613
Trail_1597d	Adjusted Closing price in n-1 relative to n-1597
Trail_1d_Vol1	Trading volume in day n-1 relative to 5 days moving average of trading volume
Trail_1d_Vol2	Trading volume in day n-1 relative to 18 days moving average of trading volume
Trail_1d_Vol3	Trading volume in day n-1 relative to 56 days moving average of trading volume
Trail_5d_Rel_Vol1	5 days moving average of trading volume relative to 18 days moving average of trading volume
Trail_5d_Rel_Vol2	18 days moving average of trading volume relative to 56 days moving average of trading volume
High_Low	Change from High to low
HL_OC	how much the SSE move from open to close relative to low to high

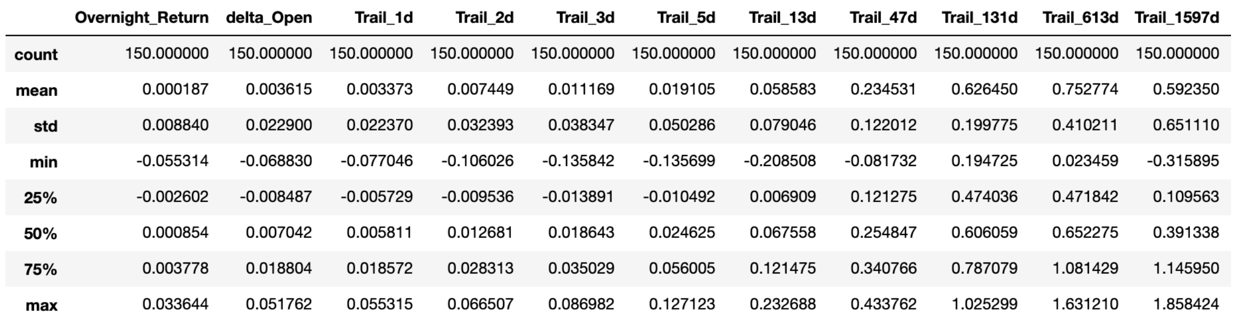
Statistics for the features in training, validating, and testing datatset



	Overnight_Return	delta_Open	Trail_1d	Trail_2d	Trail_3d	Trail_5d	Trail_13d	Trail_47d	Trail_131d	Trail_613d	Trail_1597d
count	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000	300.000000
mean	-0.000857	0.000015	0.000027	0.000110	0.000252	0.000515	0.001284	-0.000051	-0.029640	-0.230485	0.216515
std	0.003098	0.011469	0.011597	0.016752	0.020476	0.026770	0.043471	0.080894	0.093555	0.079691	0.331154
min	-0.013058	-0.058267	-0.052993	-0.057958	-0.084079	-0.092522	-0.147763	-0.140731	-0.175486	-0.340823	-0.394759
25%	-0.002317	-0.006848	-0.006290	-0.010166	-0.012414	-0.018653	-0.025260	-0.064127	-0.116385	-0.286609	-0.128777
50%	-0.000666	-0.000388	0.000096	-0.001237	-0.001725	0.000662	0.001690	-0.015531	-0.043729	-0.254860	0.276702
75%	0.000923	0.005951	0.005674	0.008813	0.013963	0.018768	0.029398	0.058183	0.061364	-0.168127	0.457625
max	0.008982	0.043345	0.043245	0.054714	0.058581	0.063274	0.106434	0.235636	0.152844	-0.033965	0.793728



	Overnight_Return	delta_Open	Trail_1d	Trail_2d	Trail_3d	Trail_5d	Trail_13d	Trail_47d	Trail_131d	Trail_613d	Trail_1597d
count	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000	250.000000
mean	-0.000608	0.000629	0.000636	0.001301	0.001926	0.003160	0.007170	0.018369	0.017793	-0.133191	-0.515614
std	0.002442	0.009050	0.008799	0.012989	0.016392	0.021401	0.033122	0.058483	0.083340	0.091816	0.102872
min	-0.007828	-0.024395	-0.028594	-0.037600	-0.048889	-0.050673	-0.073765	-0.078543	-0.101266	-0.288272	-0.663377
25%	-0.002012	-0.005117	-0.004329	-0.007499	-0.008332	-0.012836	-0.010887	-0.029066	-0.043574	-0.208154	-0.599392
50%	-0.000581	0.000322	0.000403	0.000543	0.001863	0.004077	0.008230	0.006096	-0.012781	-0.149813	-0.522396
75%	0.000820	0.006181	0.006357	0.009221	0.011881	0.016163	0.027660	0.072000	0.075997	-0.048775	-0.458581
max	0.011165	0.031770	0.028744	0.046043	0.052338	0.061357	0.085675	0.147418	0.234439	0.049002	-0.196466



	Overnight_Return	delta_Open	Trail_1d	Trail_2d	Trail_3d	Trail_5d	Trail_13d	Trail_47d	Trail_131d	Trail_613d	Trail_1597d
count	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000
mean	0.000187	0.003615	0.003373	0.007449	0.011169	0.019105	0.058583	0.234531	0.626450	0.752774	0.592350
std	0.008840	0.022900	0.022370	0.032393	0.038347	0.050286	0.079046	0.122012	0.199775	0.410211	0.651110
min	-0.055314	-0.068830	-0.077046	-0.106026	-0.135842	-0.135699	-0.208508	-0.081732	0.194725	0.023459	-0.315895
25%	-0.002602	-0.008487	-0.005729	-0.009536	-0.013891	-0.010492	0.006909	0.121275	0.474036	0.471842	0.109563
50%	0.000854	0.007042	0.005811	0.012681	0.018643	0.024625	0.067558	0.254847	0.606059	0.652275	0.391338
75%	0.003778	0.018804	0.018572	0.028313	0.035029	0.056005	0.121475	0.340766	0.787079	1.081429	1.145950
max	0.033644	0.051762	0.055315	0.066507	0.086982	0.127123	0.232688	0.433762	1.025299	1.631210	1.858424

Implementation

Supervised machine learning algorithms XGBoost classifier was used in the project. In this study, SSE is chosen for testing the model and backtesting the trading strategies. The reason for choosing SSE for preliminary testing is that price of SSE is more volatile in the testing period, experiencing a big drop after an upward movement, so the performance of the trading simulation during the test day is fair. The main objective of this study is to develop the classfication model (predicting the directionality value of the stock) that works in the real trading market. The validation set is used to tun the hyperparameter, and then the testing set is used to test the predicting accuracy. Trading simulation of the models are discussed in the later sections.

8. Refinement

After the XGBoost model is trained by the training set, the model is used to perform hyperparameter tuning in the validation set. In this process, the training features are fed into the model, and then the after-fitting model is used to predict the daily directionality of open price of SSE in the validation time period, from 2014-11-20 to 2015-07-02 (150 trading days).

In performing hyperparameter tuning, two hyperparameters are adjusted at the same time. One of the benefits of the XGBoost classifieris that there are lots of hyperparameters offered to make adjustment. The following Table contains all the hyperparameters of XGBoost that will be adjusted in the model.

Hyperparameter	**Definition**
n_estimators	Number of boosted trees to fit
max_depth	Maximum tree depth for base learners
learning_rate	Boosting learning rate
min_child_weight	Minimum sum of instance weight(hessian) needed in a child
subsample	Subsample ratio of the training instance
colsample_bytree	Subsample ratio of columns when constructing each tree
colsample_bylevel	Subsample ratio of columns for each split, in each level
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree

There are four pairs of hyperparameters that will be tuned. They are n_estimators and max_depth, XGBoost learning rate and min_child_weight, subsample and gamma, and colsample_bytree and colsample_bylevel. The study uses the same hyperparameter technique that data scientist Ng (2019) used in his study.

XGBoost n_estimators and max_depth

N_estimators is the number of the boosted trees used in the model. The goal of each tree is to minimize the error of the previous tree. When adding more trees in the series, each tree will focus on the error from the previous one to make boosting more efficient. However, it is still a trade-off game. Although more trees added sequentially will make the model perform better, an overfitting problem will arise if too many trees are used. Overfitting is when the model is too closely fit to the training set and the value of loss function (usually refers to Mean Square Error) is low, but the performance of the model on the test set will be bad (Yildirim 2020).

Max_depth, maximum number of nodes from the root to the farthest leaf of the tree, indicates the depth of the tree. The model will be more complex and will capture more information of the data as the depth increases. However, increased depth will cause an overfitting problem because splits will be less relevant when adding more nodes (Spark 2019).

XGBoost learning rate and min_child_weight

The hyperparameter learning rate means that the effect of each new tree will be less. This is such an important technique to deal with the overfitting problem that the model overfits the training set. The learning rate will slow down the learning in the model by applying a weight for the correction by additional trees. The model will be more robust in preventing the overfitting problem when the learning rate is lower (Leoni 2020).

The hyperparameter min_child_weight corresponds to the minimum sum of instance weight (hessian) needed in a child. In other words, it is the minimum number of instances required to create a new node in the tree. Generally speaking, if all samples have a weight of 1, min_child_weight is just the number of samples. The benefit of the smaller min_child_weight is to allow for more complex trees by allowing the algorithm to create the children corresponding to fewer samples. However, extremely smaller min_child_weight will give rise to the overfitting problem (Spark 2019).

XGBoost Subsample and Gamma

XGBoost subsample is denoted as the fraction of observations to be randomly sampled for each tree (XZZ 2020). For example, when the subsample equals 0.7, the algorithm will sample 70% training data prior to growing trees. The algorithm will then become more conservative and the overfitting problem will be controlled when the value of the subsample becomes smaller.

Gamma is the minimum loss reduction required to make a further partition on the leaf node of the tree. The value of gamma ranges from zero to infinity. The algorithm will become more conservative as the gamma becomes larger.

XGBoost colsample_bytree and colsample_bylevel

Colsample_bytree denotes the subsample ratio of columns when constructing each tree (XGBoost Document 2020). In other words, it is the fraction of the randomly selected training samples used to train each tree.

Colsample_bylevel denotes the subsample ratio of columns for each level (XGBoost Document 2020). In other words, it is the fraction of randomly selected features in each node that will be used to train each tree.

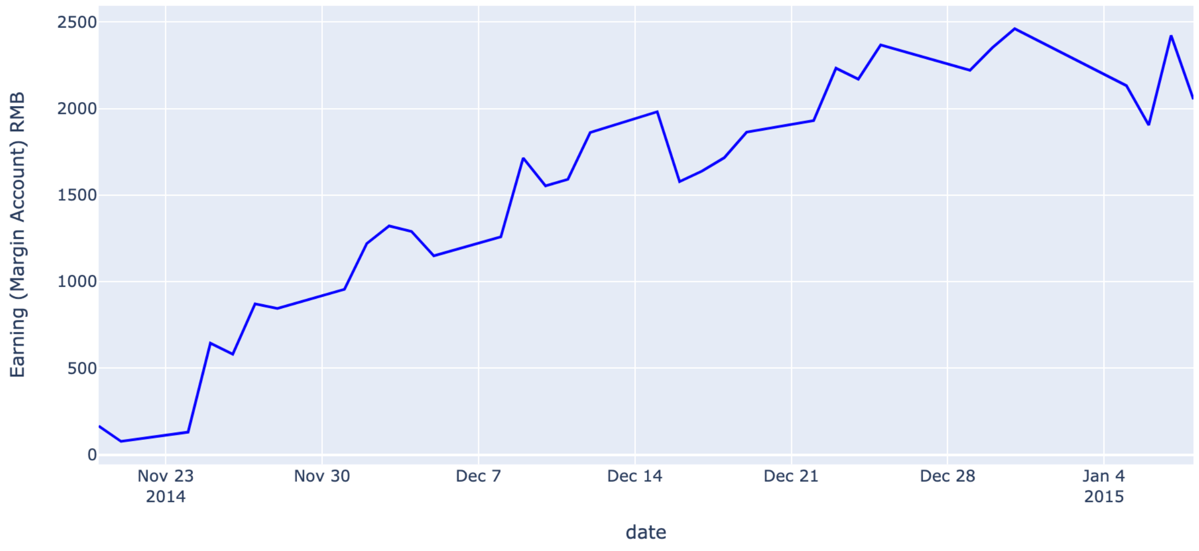
(Hyperparameters tuning)

	hyperparameters	**original**	**after_tuning**
0	n_estimators	100	10
1	max_depth	3	4
2	learning_rate	0.1	0.3
3	min_child_weight	1	20
4	subsample	1	0.9
5	colsample_bytree	1	1
6	colsample_bylevel	1	0.9
7	gamma	0	0.6
	Accuracy Score		63.64%

Result

The XGBoost Classifier ends up generating the accuracy score of 63.64%. To directly perceive the performance of the XGBoost-based trading strategy, trading simulation is used to test the performance of the strategy between 2014-11-20 to 2015-07-02 (150 trading days). In a practical trading environment, backtesting is used by the investor to evaluate and optimize trading strategies. If the strategies had performed well in the past, the strategies will probably perform well in the future, and vice versa. The risky assumption is that past performance of the strategies can also predict future performance.

In this study, the backtesting technique evaluates the performance of XGBoost-based Trading strategy on trading the SSE stock from 2014-10-21 to 2015-12-31 (150 trading days) (testing set). In the trading simulation, the long position will be created as the predicted open price of SSE goes up, and the SSE stock will not be short without existing in the long position. Moreover, all the money in the account will be used when creating the position, and all the positions will be cleaned when the predicted price takes the unfavorable direction. For example, in this study, $10,000 will be used to trade. All $10,000 will be used to buy SSE composite index when the predicted open price of SSE goes up. And the return for the trading strategy is 20.53 percent, which is excellent compared with the benchmark.



11. Conclusion

A far-reaching data analytics procedure employing a decision-tree-based ensemble machine learning algorithm, the XGBoost classifier, has been developed to predict the directionality of the open price of SSE. To visualize how well the XGBoost classifier performs on forecasting daily open price direction of the SSE, the researcher visualizes the performance of the XGBoost classifier by backtesting the trading strategy. The data analytics procedure starts with preprocessing the data including scaling the data and concluding with forecasting and backtesing results. Moreover, the accuracy of the XGboost classifieron forecasting SSE is highest when n_estimators = 10, max_depth = 4, learning_rate = 0.3, min_child_weight = 20, subsample = 0.9, colsample_bytree = 1, colsample_bylevel =0.9, and gamma = 0.6.

References

Arora, Aman. “Why Random Forests Can't Predict Trends and How to Overcome This Problem?” *Medium*, Data Driven Investor, 29 Dec. 2018, medium.com/datadriveninvestor/why-wont-time-series-data-and-random-forests-work-very-well-together-3c9f7b271631.

Atsalakis, George & Valavanis, Kimon. (2009). Surveying stock market forecasting techniques - Part II: Soft computing methods. Expert Syst. Appl.. 36. 5932-5941. 10.1016/j.eswa.2008.07.006.

Brownlee, Jason. “Feature Importance and Feature Selection with XGBoost in Python.” *Machine Learning Mastery*, 11 Dec. 2019, machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/.

Chen, Tianqi, and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System.” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, 2016, doi:10.1145/2939672.2939785.

Hargrave, Marshall. “How to Use the Sharpe Ratio to Analyze Portfolio Risk and Return.” *Investopedia*, 29 Jan. 2020, www.investopedia.com/terms/s/sharperatio.asp.

Leoni, Florencia. “From Zero to Hero in XGBoost Tuning.” *Medium*, Towards Data Science, 14 Jan. 2020, towardsdatascience.com/from-zero-to-hero-in-xgboost-tuning-e48b59bfaf58.

Maverick, J.B. “Learn What a Good Sharpe Ratio Is.” *Investopedia*, 29 Jan. 2020, www.investopedia.com/ask/answers/010815/what-good-sharpe-ratio.asp.

Morde, Vishal. “XGBoost Algorithm: Long May She Reign!” *Medium*, Towards Data Science, 8 Apr. 2019, towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d.

Ng, Yibin. “Machine Learning Techniques Applied to Stock Price Prediction.” *Medium*, Towards Data Science, 3 Oct. 2019, towardsdatascience.com/machine-learning-techniques-applied-to-stock-price-prediction-6c1994da8001.

Raghavan. “Scatter Matrix, Covariance and Correlation Explained.” *Medium*, Medium, 16 Aug. 2018, medium.com/@raghavan99o/scatter-matrix-covariance-and-correlation-explained-14921741ca56.

“Exploratory Data Analysis.” Experimental Design and Analysis, by Howard J. Seltman, Carnegie Mellon University , 2018, pp. 61–63.

Spark, Cambridge. “Hyperparameter Tuning in XGBoost.” *Medium*, Cambridge Spark, 23 Dec. 2019, blog.cambridgespark.com/hyperparameter-tuning-in-xgboost-4ff9100a3b2f.

Tukey, John W. Exploratory Data Analysis. Reading, Mass: Addison-Wesley Pub. Co, 1977.

Thawornwong, Suraphan, and David Enke. “The Adaptive Selection of Financial and Economic Variables for Use with Artificial Neural Networks.” *Neurocomputing*, vol. 56, 2004, pp. 205–232, doi:10.1016/j.neucom.2003.05.001.