**Navigating Market Volatility: An In-Depth Evaluation of Machine Learning Models in Stock Price Prediction Dynamics**

Brian Hsu, Jason Ji

Emory University

Atlanta Georgia USA

[Brian.hsu@emory.edu](mailto:Brian.hsu@emory.edu)

[Jason.ji@emory.edu](mailto:Jason.ji@emory.edu)

**Abstract**

Stock price prediction poses a formidable challenge in the financial domain due to its complexity and susceptibility to numerous external factors. The volatile nature of financial markets and the multitude of variables influencing stock prices make accurate prediction a crucial yet intricate problem. Individual investors, financial analysts, and institutions continue to seek improved tools and quantitative models to anticipate stock movements, facilitating more informed decision-making. Consequently, developing and implementing effective stock price prediction models have become essential in the status quo. Motivated by machine learning lectures and online sources, this report investigates the intricate challenge of exploring machine learning techniques to enhance predictive accuracy and contribute to the ongoing discourse in financial forecasting.

In addressing the complexities of stock price prediction, our project centers on analyzing and implementing a diverse set of machine learning algorithms. Linear regression, random forest, and ensemble methods such as bagging, AdaBoost, and stacking regressor are employed to capture the relationships within the historical stock data. Leveraging a dataset sourced from Yahoo Finance through the yfinance library with data ranging from December 12, 1980, to December 31, 2022, we include a comprehensive set of 108 attributes. These attributes encompass daily highs, lows, openings, closings, volumes, moving averages, indexes (e.g., S&P500), and competitors' and suppliers' stock statistics. Incorporating competitor stock performance as an additional feature in our approach distinguishes us from existing models. The ultimate goal is to predict the next day adjusted price, offering a more nuanced understanding of stock dynamics. Through this approach, we aim to contribute novel insights and methodologies to stock price prediction to address the importance of considering a broader spectrum of influential features in financial modeling.

**Introduction**

Stock price prediction is a critical and intricate task within the financial market, as complex economic, political, and environmental factors influence the stock market dynamics. Markets' volatility and complexity necessitate implementing machine learning models to capture the nuanced interplay between different variables. We want to explore a stock price prediction using models we learned in class to predict stock prices and compare their accuracy against LSTM results in existing works.

Our approach addresses this problem by embracing diverse machine learning algorithms with 108 attributes to capture different facets of the intricate relationships within historical stock data and company financials. As we recognize the interconnected nature of stocks within the market ecosystem, our approach to incorporating competitor, supplier, and index performances as predictive features sets us apart from other research, introducing more comprehensive analysis than existing research. This novel approach aims to provide a more holistic and realistic representation of the stock market, offering valuable insights for investors and decision-makers to navigate the complexities of financial markets.

**Data Description & Data Preprocessing**

The dataset, sourced from Yahoo Finance using the yfinance library, spans from December 12, 1980, to December 31, 2022. We set our target company to be Apple as it has an initial public offering (IPO) date of December 12, 1980, with a wide range of competitors and suppliers for us to construct our data features. The dataset comprises 10,603 samples with 108 attributes, encompassing daily high, low, open, close, volume, moving average, index (e.g., S&P500), and competitors/suppliers' stock statistics. The dataset also includes crucial attributes like competitor stock performance to enhance its comprehensiveness. The next-day adjusted price was chosen as the target variable, accounting for dividends and splits.

Predicted Class

We adjusted the price by shifting it one day forward to ensure accurate predictions of the next day's price for comparisons, analyzing correlation matrices, and error measurements.

Null values/Missing Value

Because we need to find competitors’ stock statistics, we had to take the difference in Initial Public Offering (IPO) time into account. For dates before a competitor’s IPO, the stock price will be Null. Null value were addressed in two circumstances:

1. NaN values prior to the latest competitor’s IPO date (2011):For NaN prior to the IPO date, the company was not traded in the secondary market, so we delete the data samples prior to 2011.
2. NaN values after latest competitor;s IPO date (2011): For NaN after the IPO date, we replace the value with mean.

Pearson Correlation

We utilized Pearson correlation to identify highly correlated features with correlation to target features higher than 0.5 (Figure 1).

A red and blue plaid with white lines

Description automatically generated

Figure 1: Correlation Matrix Heatmap of 108 Features

Remove Highly Correlated Features:

We aim to reduce redundancy in highly correlated features. Therefore, collinearity was mitigated by using correlation matrix heatmap. Highly correlated values are removed.

Normalization

We normalized the dataset by implementing StandardScaler() from sklearn.

Train/Test Split

We split the dataset into training and testing sets using train\_test\_split() from sklearn .model\_selection. We set the training set size to 0.7 of the original dataset size and the testing set size to 0.3 of the original datasets.

**Methods**

In this section, we outline the selected machine learning algorithms employed in our stock price prediction model, offering a comprehensive overview of each algorithm's rationale, description, best hyperparameter obtained, and applicability to the problem at hand.

We implemented hyperparameter tuning to select the best parameters for each machine-learning model. The hyperparameter tuning is performed using the RandomizedSearchCV function from sklearn.model\_selection with the negative mean absolute error scoring metric, indicating that the optimization minimizes the negative mean absolute error (MAE).

1. Linear Regression

* Description: Linear regression is a foundational model for predicting stock prices. It assumes a linear relationship between the independent variables (features) and the dependent variable (stock price). The model calculates the coefficients for each feature to minimize the sum of squared differences between predicted and actual values.
* Rationale**:** The simplicity and interpretability of linear regression make it an essential baseline model. It clearly explains how individual features impact stock prices and serves as a benchmark for more complex algorithms.

1. Random Forest
   * Description**:** Random Forest is an ensemble learning method that constructs multiple decision trees during training. Each tree is trained on a subset of the data and a random subset of features. The final prediction is an average or voting of predictions from individual trees.
   * Rationale**:** Random Forest excels at handling complex relationships in data and reducing overfitting. By combining multiple decision trees, it captures non-linear patterns in stock price movements, offering improved accuracy and generalization.
   * BestParameter**:** n = 150; min\_samples\_split =2; min\_leaf\_samples = 1; max\_features = log2; max\_depth = 10
2. Ensemble Methods (Bagging, AdaBoost, and Stacking Regressor)
   * Bagging (Bootstrap Aggregating):
     1. Description:Bagging creates an ensemble by training multiple models independently on bootstrapped subsets of the data. The final prediction is an average or voting of individual model predictions.
     2. Rationale**:** By creating diverse subsets of the training data, Bagging reduces the prediction variance, mitigating the risk of overfitting. This diversity ensures that each model captures different patterns in the data, leading to a more robust and accurate ensemble prediction.
     3. Best Parameter**:** n = 40
   * AdaBoost (Adaptive Boosting):
     1. Description: AdaBoost sequentially improves the performance of weak learners by assigning more weight to misclassified instances in each iteration.
     2. Rationale:AdaBoost's strength lies in its ability to adapt to the weaknesses of individual models. By emphasizing misclassified instances in each iteration, AdaBoost hones in on the complex patterns that may be overlooked by unique models, thus improving overall prediction accuracy.
     3. Best Parameter: n =150; learning rate = 1
   * Stacking Regressor:
     1. Description:Ensemble methods enhance predictive performance by leveraging the diversity of multiple models. Bagging reduces variance, AdaBoost focuses on sequential improvement, and Stacking Regressor combines the strengths of diverse models, collectively contributing to improved accuracy and robustness.
     2. Rationale:Stacking Regressor capitalizes on the diverse strengths of individual models by learning how to combine their predictions best. This approach allows the ensemble to benefit from the complementary aspects of each model, resulting in a more nuanced and accurate prediction.
     3. Best Parameter:alpha = 1

**Experiments & Results**

A graph of blue rectangular bars

Description automatically generated with medium confidence

Figure 2: Mean Absolute Error (MAE) Comparison

A graph of blue squares

Description automatically generated

Figure 3: Mean Square Error (MSE) Comparison

A graph of blue rectangular bars

Description automatically generated

Figure 4: Root Mean Square Error Comparison

A graph showing the price of a stock price

Description automatically generated

Figure 5: Actual Stock Prices vs Model Prediction

A black screen with white text

Description automatically generated

Table 1: MAE, MSE, and RMSE for 6 Different Machine Learning Models

The analysis reveals that the Random Forest, Bagging, and Stacking regressors have achieved comparable performance, each yielding an RMSE (Root Mean Square Error) around 1.58. This consistency is also observed in their MSE (Mean Squared Error) and MAE (Mean Absolute Error) results, although the Random Forest exhibits a notably superior performance in these metrics. This outcome underscores the strength of the Random Forest approach in managing complex data relationships and mitigating overfitting.

Conversely, the AdaBoost regressor has underperformed in comparison. Upon examination, it appears that its shortcomings may be attributable to residual outliers and noise within the preprocessed data. These elements seem to have led to overfitting in the AdaBoost model, as it overly adjusts for errors during the boosting process. This insight into AdaBoost's limitations within this specific data context is crucial for guiding future model selection and data preprocessing strategies.

Overall, the predictions of different models for the next day closing price are fairly accurate, as shown by Figure 5 where being compared to the real price.

**Discussion**

After obtaining the model results, we want to compare it to the existing work so far. Our research has found that the most commonly used models are Linear Regression, Random Forest, and Long Short Term Memory (LSTM). Below is a table that summarizes the existing model’s performance and compares with our models.

|  |  |  |  |
| --- | --- | --- | --- |
| Paper | Model | Existing Work Performance (RMSE) | Our Performance (RMSE) |
| “Stock price prediction using machine learning  on least-squares linear regression basis” | Linear Regression | 0.512 | 1.7 |
| “Stock Closing Price Prediction using Machine  Learning Techniques.” | Random Forest | 1.53 | 1.57 |

Linear Regression:

The findings indicate that the linear regression model exhibits suboptimal performance when applied to our dataset, which is characterized by a greater number of attributes compared to the datasets used in prior studies. Typically, linear regression, owing to its simplicity, is more effective with datasets that have fewer attributes, where the relationships between variables are less complex.

Random Forest:

The current implementation of the Random Forest Regressor model yields an RMSE (Root Mean Squared Error) of 1.574399, which is closely aligned with the benchmark RMSE of 1.53 reported in existing research. This near parity in results demonstrates that our model is capable of achieving a level of accuracy comparable to established studies.

LSTM:

The existing work (paper named “Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model”) demonstrates the use of LSTM and Bi-Directional LSTM for deep learning-based stock price prediction, with the lowest RMSE value of 0.0004.

In conclusion, it is apparent that while our models exhibit a commendable level of accuracy, there remains a notable disparity when benchmarked against existing studies, particularly those utilizing LSTM models. This underscores the exceptional capabilities of LSTM in forecasting time-series data like stock prices. It would be worthwhile to explore the differences in dataset complexity, feature engineering, and model tuning between the current work and the existing literature to understand the reasons behind these discrepancies and to identify opportunities for improvement.

Top of Form

Bottom of Form

**Contributions**

Brian Hsu:

Responsible for constructing and gathering the stock data information for the raw dataset by extracting historical data from Yahoo Finance. His tasks involve extracting historical data from Yahoo Finance, including stock prices, trading volumes, and other relevant financial metrics. Brian is also responsible for data preprocessing, which involves cleaning the raw data, handling missing values, and transforming the dataset to make it suitable for machine learning applications. To conduct hyperparameter tunning to select the best parameters for each model.

Jason Ji

Responsible for preparation of the presentation, conducting extensive research on existing works, and evaluating results in discovering hidden reasons for best and worst-performing algorithms. Jason is also responsible for implementing all the machine learning models and visualizations to better portray the performance of each machine learning model.

**Code/Dataset**

The following link leads to the Google Drive folder containing all code and datasets that were used throughout the project:

<https://drive.google.com/drive/folders/1bsnqWInCYs9BS6Tj3gBfmrEnzdLnSvcQ?usp=sharing>