Predicting Twitter Stock Price with Machine Learning

Jingbo Yang
Brown University
DSI

Link: https://github.com/Psy3962/data1030_projects.git

Introduction

The economic value of machine learning has been extensively explored, with large companies leveraging its capabilities for predicting preferences, generating suggestions, and classifying customers. However, the application of machine learning techniques to generate income for individuals remained an elusive concept. This paper aims to explore the potential of machine learning for individual benefit by focusing on stock price prediction.

This project specifically investigates the prediction of Twitter's daily high price using various machine learning techniques, posing a typical regression problem. While data regarding Twitter's stock price is available on Investing.com, the author obtained the dataset from Kaggle. This dataset comprises date and various price measurements for each date (including close price, open price, high price, and low price).

Several studies have examined the effectiveness of machine learning techniques for predicting stock prices. Islam et al. (2020) compared the performance of different models, including Elastic Net, SVM, Random Forest, and XGBoost, for predicting stock prices in Bangladesh. Their findings identified XGBoost as the most accurate model, followed by Random Forest and SVM. Baviskar et al. (2019) proposed a hybrid model combining SVM and Random Forest for predicting stock market trends, achieving an accuracy of 75%, exceeding the performance of individual models. Additionally, Boonlert et al. (2022) investigated the efficacy of Random Forest and XGBoost for predicting stock movements in Thailand, incorporating technical indicators and Google Trend searches. Both models proved effective, with XGBoost exhibiting slightly superior performance in terms of annualized return. These studies highlight the potential of machine learning for stock price prediction and suggest that specific models can offer significant accuracy improvements over traditional approaches.

EDA

The Twitter stock price dataset consists primarily of numerical data representing prices, with date information also present. The target variable, high price, has a mean of $35.34 and a standard deviation of $13.82.

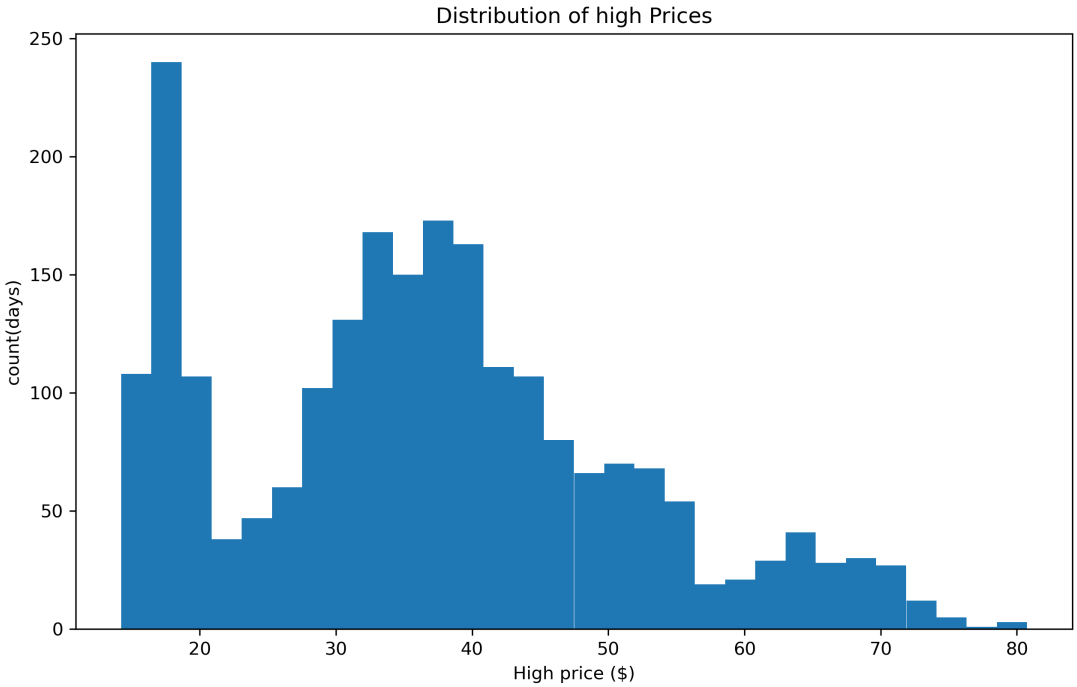


Fig.1: Overall distribution of high price. This figure shows the distribution of high price, where most values are concentrated between $30 and $40. This indicates that the high price falls within this range most frequently.

To further demonstrate whether the each point of a time series data showing high price are positively correlated, negatively correlated, or independent of each other, an autocorrelation plot was presented:

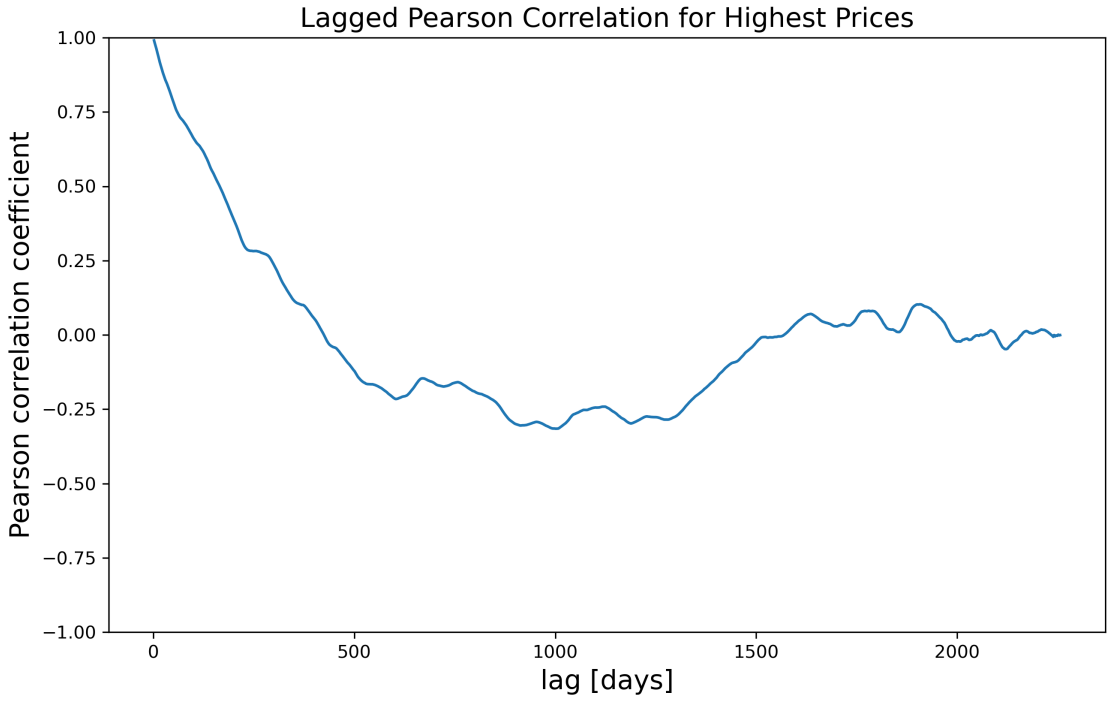


Fig.2: Lagged Pearson correlation for high prices of the day. This figure presents an autocorrelation plot for the high price values. It reveals a decrease in correlation to negative values around a lag of 500 days, followed by an increase back to zero. The oscillating nature of the correlation suggests that multiple factors influence the relationship between different points in the time series data of the stock price.

Last, a line graph is presented to show the changes of high price as progression with time:

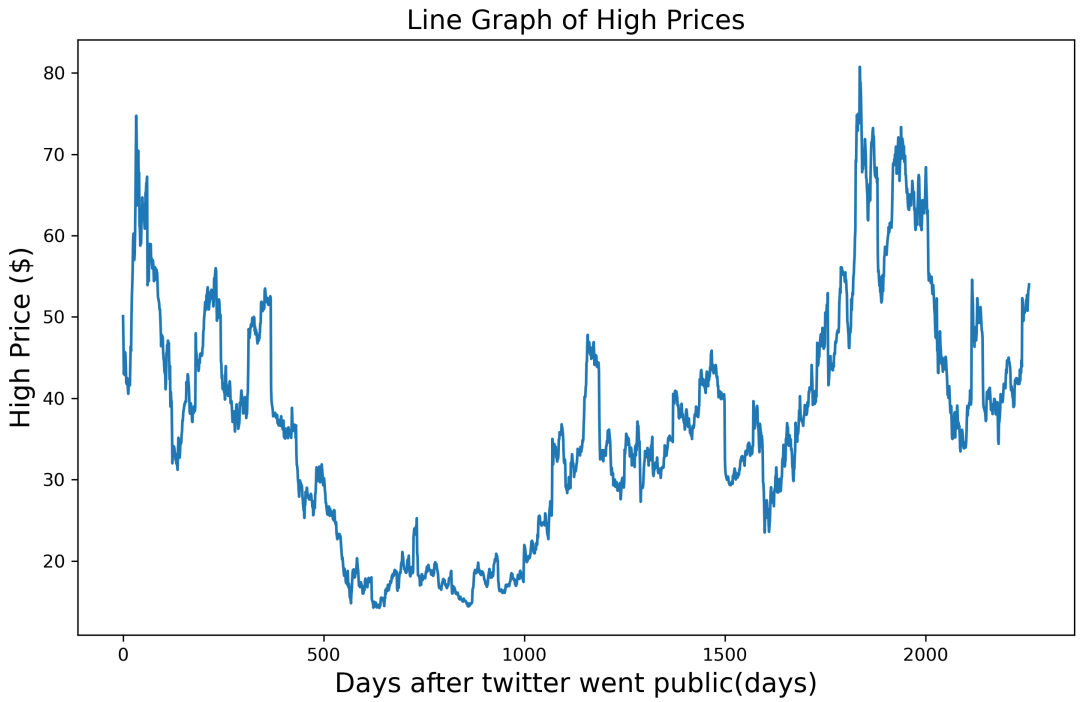


Fig.3: Line graph of high prices. This figure depicts the progression of high prices over time. The graph shows that Twitter's highest daily price reached its lowest point between 500 and 1000 days after the company went public, followed by a peak around 1500 to 2000 days.

Methods:

1. Feature Engineering:

1.1 Date Conversion:

The first step involved converting the date string into more informative features. This was achieved by extracting the day, month, and year as separate columns. Additionally, a new

column indicating the day of the week (Monday to Sunday) was created using numerical encoding (0-6). This transformation aimed to prepare the date data for further analysis by making it easier to work with and extracting relevant temporal features useful for time series forecasting.

1.2 Introducing Lagging:

This process focused on creating new features based on existing ones by analyzing their past values, potentially valuable for time series forecasting. The author iterated through all price-related variables (high price, low price, etc.) and a set of time lags (1, 3, 7, etc.). For each variable and lag combination, a new feature was created in the data frame by shifting the original variable values back by the specified lag.

2. Splitting:

2.1 Train-Test Split:

Due to the time-series nature of the data, maintaining the temporal order while splitting was crucial. To avoid data leakage, the author used `train_test_split` with `shuffling = False` and `test_size = 0.8`. This approach ensured that the last 20% of the data served as the test set, while the first 80% formed the combined training and validation set. This guaranteed that the test set contained no data earlier than the training and validation sets.

2.2 Cross-Validation:

Within the cross-validation section of the `GridSearchCV`, the author used `timeseriessplit` with `n_splits = 5` on the training and validation set. This specialized function in scikit-learn ensured that the validation/test set remained chronologically later than the training set in each fold of the cross-validation process.

3. Preprocessing and Pipeline:

3.1 Feature Types:

Feature engineering resulted in two types of features: categorical and numerical.

3.2 Preprocessing Pipeline:

A pipeline was constructed to apply appropriate transformations to each feature type. Categorical features were processed using a one-hot encoder followed by standard scaling. Numerical features, on the other hand, were only subjected to standard scaling.

4. Machine Learning Models and Evaluation Metrics:

Four different machine learning models were applied, of which three were non-linear and one was linear. Additionally, two models required controlling for randomness. The table below summarizes the models, the hyperparameters that were tuned, and the need for randomness control.

Model	Is linear?	Hyperparemeter	Uncertainties?
XGBOOST	NO	"reg_alpha": [1e-2,1e-1, 1e0, 1e1,1e2], "reg_lambda": [1e-2,1e-1, 1e0, 1e1,1e2], "max_depth": [1, 3, 10, 30, 50]	Yes, check for different seeds
Random Forest Regressor	NO	n_estimators': [100, 200, 300,400,1000], 'max_depth': [None, 10, 20, 30,50,100]	Yes, checked for random_state
Support Vector Regression	NO	C': [0.1, 1, 10, 100,150, 200,500], 'gamma': [0.000001, 0.01,0.02 ,0.1, 1, 10, 100]	NO
Elastic Net	YES	alpha': [0,0.01,0.03,0.05,0.1,0.5], 'l1_ratio': [0.11, 0.25, 0.5, 0.75, 0.9]	NO

Table 1: This table shows the four models and their characteristics. Among the 4 models, only Elastic Net is linear and only XGBOOST and Random Forest Regressor needs to be controlled for random state.
Due to the introduction of lagging, some features have missing values in some row. XGBOOST was applied to the dataset with missing values, and all other models was applied to the dataset that dropped all columns contain missing values.

5. Cross-Validation and Evaluation:

For all models except XGBoost, the author utilized `GridSearchCV` for cross-validation. As mentioned earlier, the `cv` parameter was set to `timeseriessplit` with `n_splits=5` to ensure chronological order in each fold. Due to the typical regression nature of the problem, the primary evaluation metric was the mean squared error. After running `GridSearchCV`, the best model was identified and its test score was obtained.

For Random Forest Regressor, the author iterated through five random states to create five different models. This allowed for the calculation of the mean and standard deviation of the test scores across these models.

XGBoost, due to its longer runtime, was subjected to `RandomSearchCV` instead of a full grid search. Five random states were tuned within `RandomSearchCV`, resulting in 25 total scores (5 for each random state in `RandomSearchCV` and 5 for each seed within XGBoost). Finally, the 25 test scores were averaged and their standard deviation was computed.

Results

Due to the time series nature of the data, the baseline model for this project simply predicts the high price of today as the same as the high price of yesterday. Since there is only one baseline score, calculating the standard deviation for each model's performance relative to the baseline is not possible. Therefore, only the mean scores will be discussed.

Overall, only the Elastic Net model was able to outperform the baseline accuracy. The following figure summarizes the results:

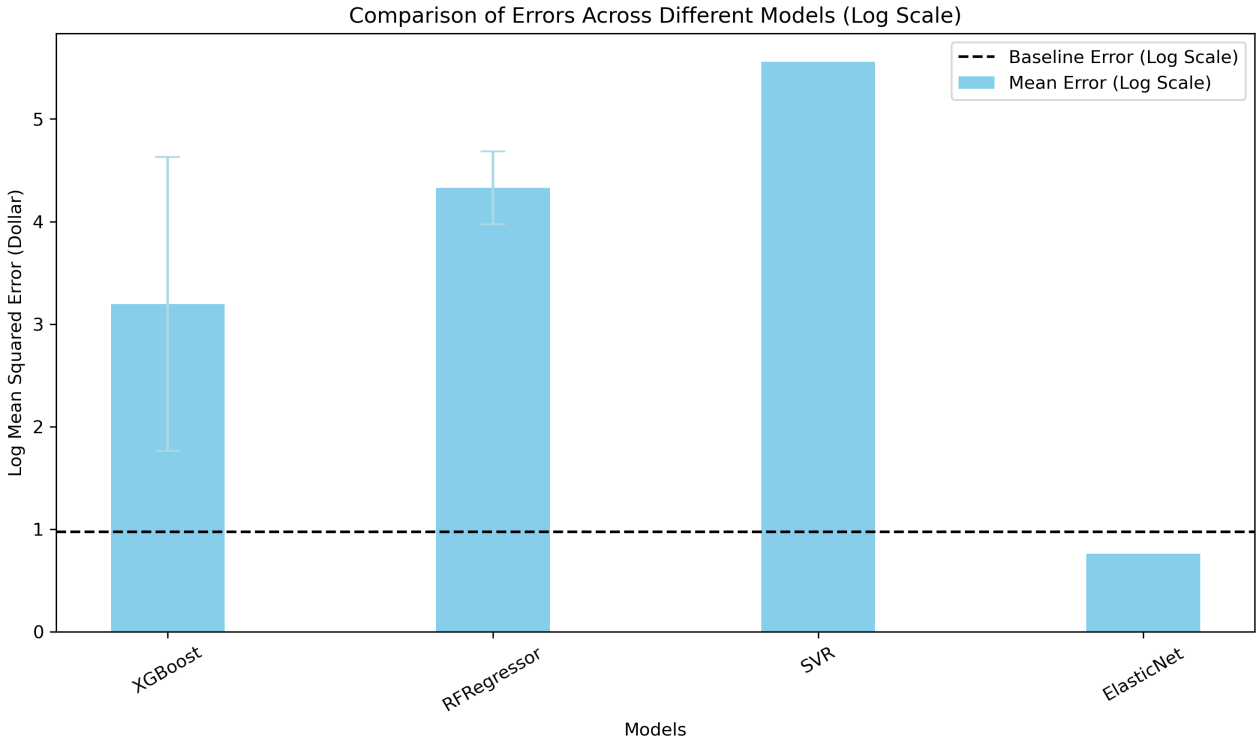
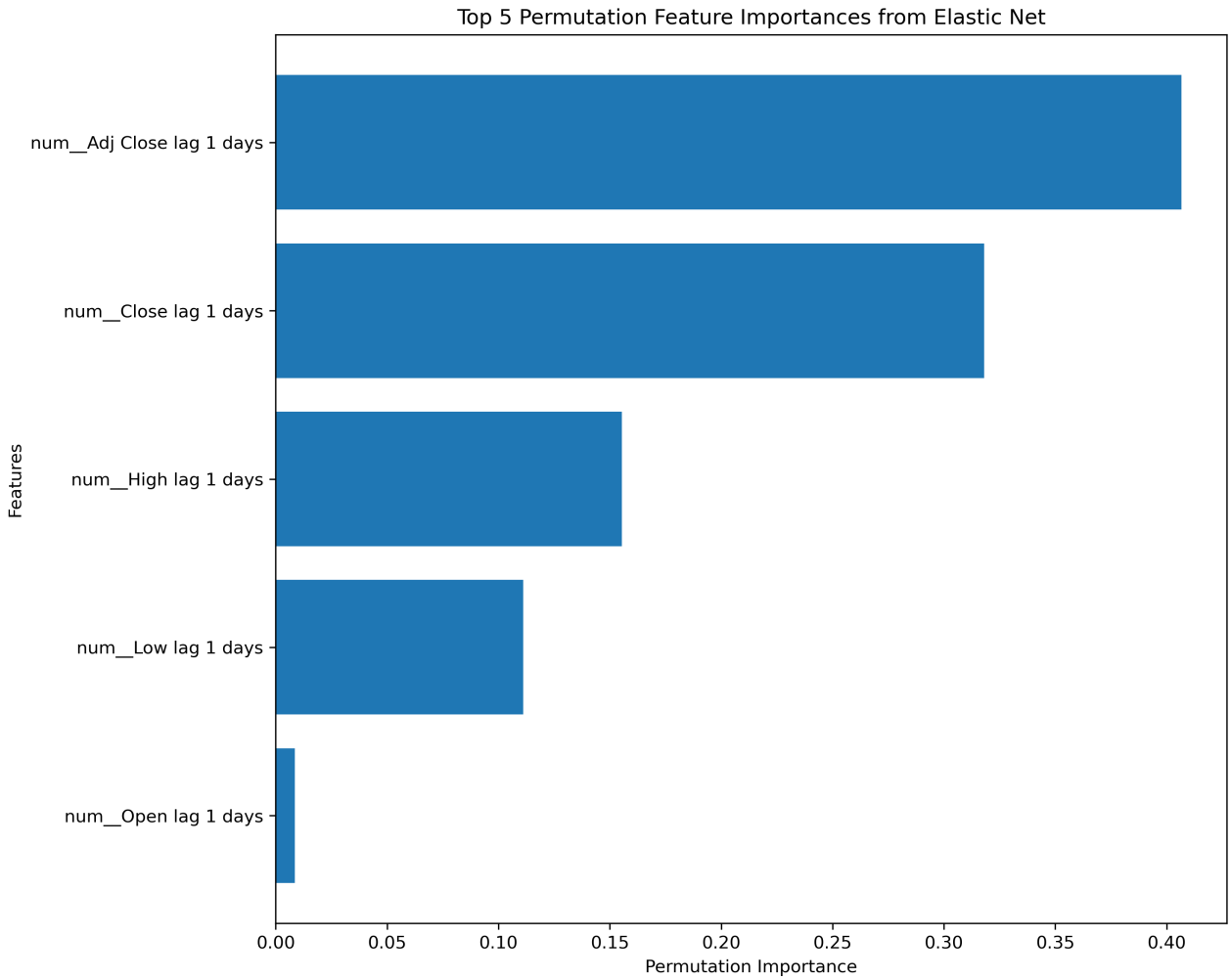


Fig.4: Result of various models in log. It shows that only Elastic Net has a MSE which is smaller than the baseline. The error bar(also in log scale) is only displayed for XGBOOST and Random forest regressor which required control for randomness.

The performance of the models yielded rather surprising results. Contrary to previous research where non-linear models like XGBoost often achieve the best performance, Elastic Net, a linear model, surprisingly captured the relationship between lagged features and high price most accurately in this case. This suggests that the relationship between high price and other lagged features might be predominantly linear for Twitter's stock price.

To further explore the performance of Elastic Net, the author created global and local feature importance using both coefficient and permutation importance. This will provide deeper insights into which features were most influential in the model's predictions.


Top 5 Permutation Feature Importances from Elastic Net
num__Adj Close lag 1 days
num__Close lag 1 days
num__High lag 1 days
num__Low lag 1 days
num__Open lag 1 days
Features
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Permutation Importance

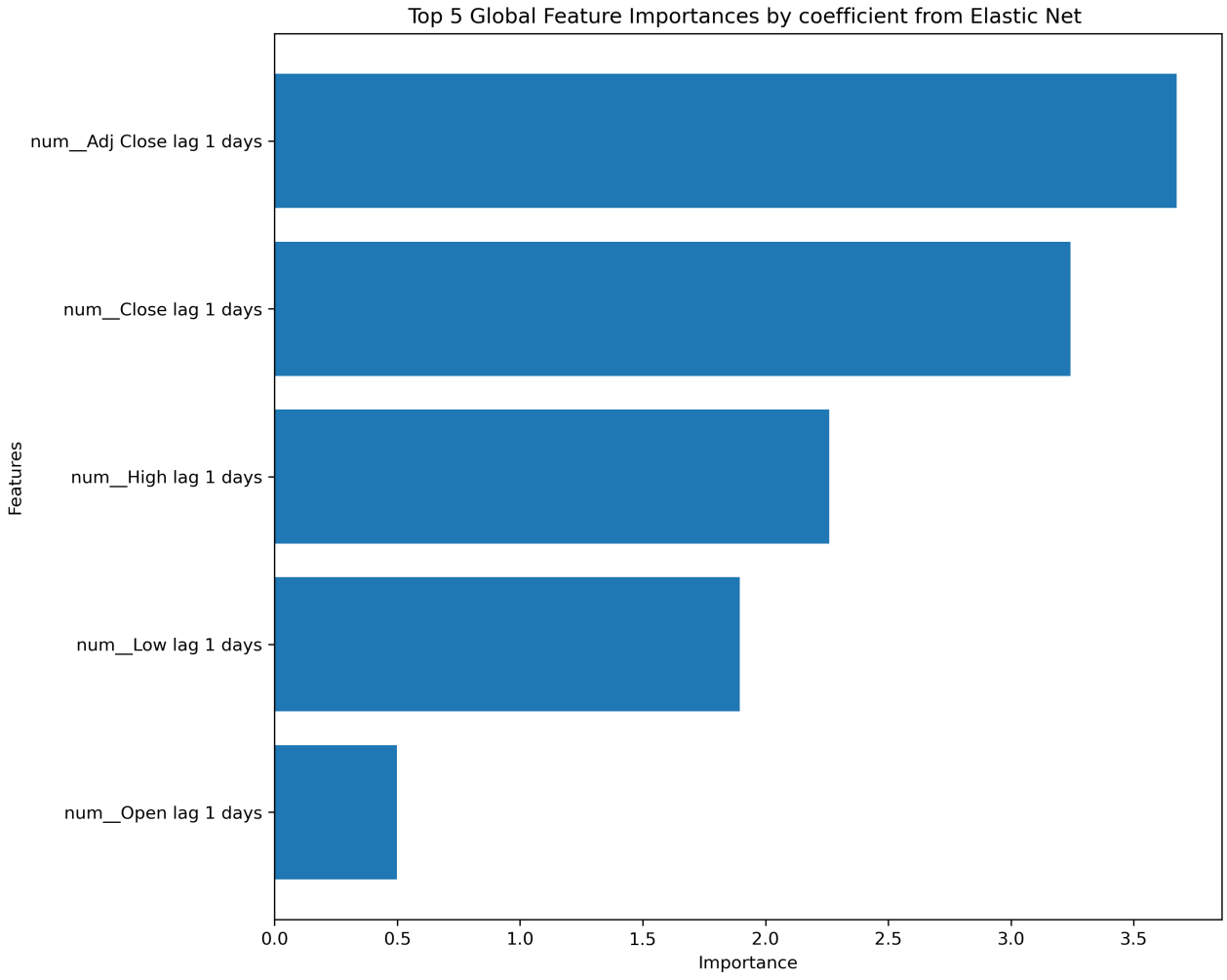
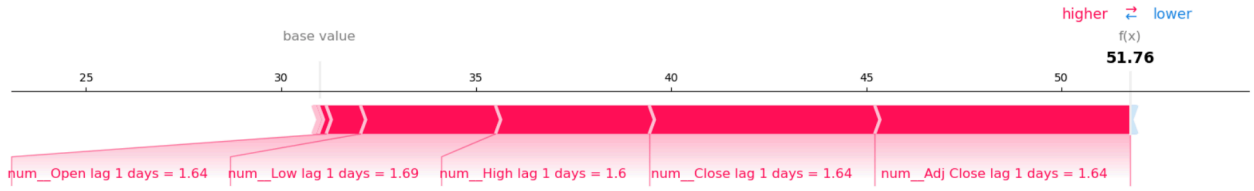

Top 5 Global Feature Importances by coefficient from Elastic Net
num__Adj Close lag 1 days
num__Close lag 1 days
num__High lag 1 days
num__Low lag 1 days
num__Open lag 1 days
Features
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Importance

Fig.5 and 6: Figures 5 and 6 present the global feature importances for the Elastic Net model, calculated using both coefficient and permutation importance. While the values differ between the two methods, they agree on the same set of features most relevant to predicting high price.

Local feature importance for point 1



Local feature importance for point 500

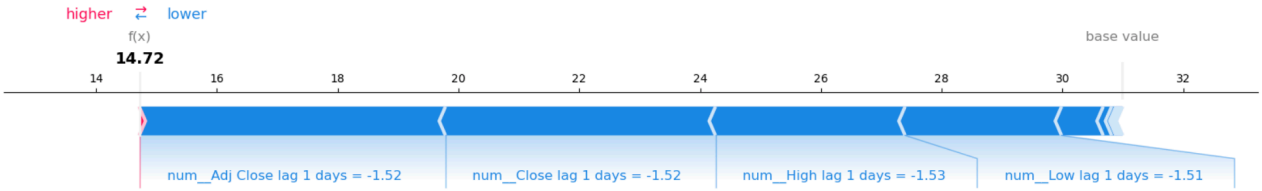


Fig.7 and 8: Figures 7 and 8 present the lobal feature importances for the Elastic Net model, calculated using shap. They agree on the same set of features most relevant to making the prediction above/below baseline

The analysis revealed that the Elastic Net model outperformed other models in predicting Twitter's high stock price. This unexpected result suggests a predominantly linear relationship between the high price and the lagged features, unlike previous studies where non-linear models dominated.

Further analysis using both coefficient and permutation importance confirmed that the most influential features for predicting high price were the adjusted close price, close price, high price, low price, and open price, all lagged by one day. This finding aligns with intuition, as yesterday's price is likely to be highly correlated with today's high price.

Interestingly, despite their varying performance, different models agreed on the same set of important features for predicting high price. This suggests a level of consistency in the data and the underlying relationships between features.

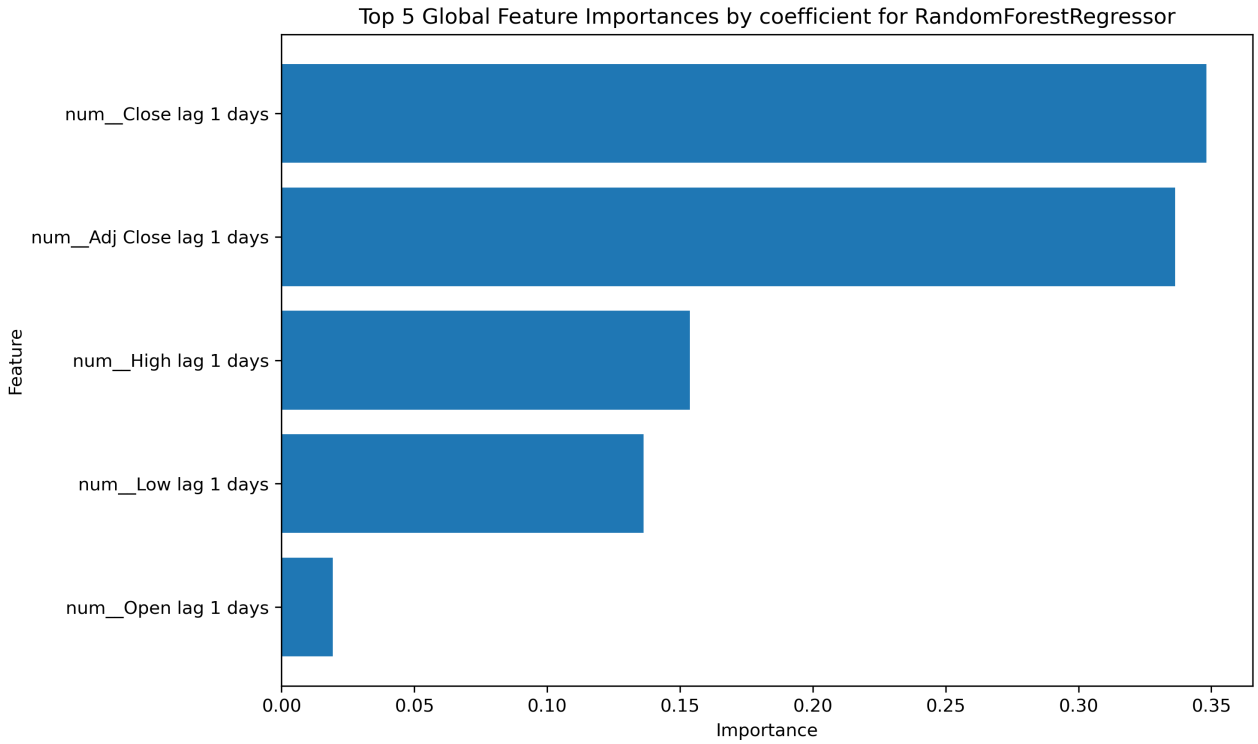


Fig.9: Global feature importances (top 5) by coefficient for random forest regressor. The graph shows that even the precise ranking are different but both random forest regressor and elastic net agree on the same set of features that contribute the most toward predicting highest price.

Outlook

The surprising findings of this project, where a linear model outperformed non-linear models, necessitate further investigation. Several factors could contribute to this result:

1. Limited Scope: The analysis focused solely on predicting high price for a single stock (Twitter) during a specific period. Expanding this scope to include more stocks and longer timeframes could provide a broader picture and reveal new insights into model behavior across different market conditions.

2. Narrow Focus: Predicting only high price using only four models limits the potential for uncovering valuable information. Exploring additional models, such as K-Nearest Neighbors regressor, and predicting other target variables, like lowest price, could yield more comprehensive insights into stock movement.

3. Single Feature Analysis: The current analysis focuses on individual feature importance. Investigating the correlation between features by permuting them in pairs could provide deeper insights into how combinations of factors influence the high price prediction.

Future research should explore these avenues to refine the model and gain a more thorough understanding of the factors influencing Twitter's stock price. This could lead to more accurate predictions and valuable insights for investors and traders seeking to navigate the market successfully.

Reference

Baviskar, S. S., Patil, V. B., & Deshmukh, P. R. (2019). Predicting Stock Market Trends Using Machine Learning Techniques. International Journal of Innovative Technology and Exploring Engineering, 8(8S2). https://www.irjet.net/archives/V5/i10/IRJET-V5I10193.pdf

Boonlert, S., Wongwatcharapaiboon, C., & Suthapanit, K. (2022). Stock Movement Prediction Using Machine Learning Based on Technical Indicators and Google Trend Searches in Thailand. Mathematics, 10(1), 5. https://www.mdpi.com/2227-7072/11/1/5

Islam, M. R., Uddin, M. S., Rahman, M. M., & Hasan, K. M. (2020). Stock Price Prediction Using Machine Learning Techniques. Journal of Advanced Research in Applied Artificial Intelligence, 7(4), 47–60. https://ijcrt.org/papers/IJCRT2210354.pdf