Predicting Stock Market Signals

Tree-based Classifiers Using Technical Indicators

Team 12B

Dennis Wu

Vy Nguyen

Kenjee Koh

Hsiang-Han Huang

Becky Wang

TABLE OF CONTENTS

Project Overview

1. What defines good investing?

Good investing can be defined in many ways. For our project, we will specifically focus on stock, or equity, investing. A common metric for institutional equity investors is outperforming relevant benchmarks, such as specific indices. However, for individual investors, we'll simplify this by using the S&P 500 as a benchmark. We will consider an individual investor's stock investing performance to be good if it outperforms the S&P 500 over a specific time frame.

2. How to keep up with the market?

Keeping up with the market can be challenging, even for professional investors. In fact, a significant portion of active equity fund managers fail to outperform their benchmarks. According to S&P Dow Jones Indices, approximately 85% of active U.S. equity fund managers underperformed the S&P 500 over a 10-year period. Given this statistic, it's clear that individual investors face an even steeper uphill battle.

This is why many investors opt for a passive investing strategy, such as investing in index funds or ETFs. By tracking a specific index like the S&P 500, passive investors can benefit from the long-term growth of the overall market without the need for active management."

3. How can we outperform the market?

Traditionally, a combination of fundamental and technical analysis can be employed for both stock selection and market timing.

a. Stock Selection:

- **Fundamental Analysis:** This involves a deep dive into a company's financial health, business model, and industry trends. By analyzing factors like revenue growth, profitability, debt levels, and competitive advantage, investors can identify undervalued stocks with strong growth potential.

- **Technical Analysis:** This technique uses historical price and volume data to identify patterns and trends. Investors can spot potential individual stocks that are poised for a significant price increase.

b. Market Timing:

- **Fundamental Analysis:** Investors can monitor economic indicators, industry trends, and company-specific news to gauge the overall market sentiment and make informed decisions about buying, selling, or holding specific stocks at a particular time.
- **Technical Analysis:** Using technical indicators and chart patterns, investors can identify overbought and oversold conditions, as well as potential trend reversals. This helps them determine optimal entry and exit points to maximize returns.

However, it's becoming increasingly difficult to consistently outperform the stock market using traditional strategies alone. As a Business Analytics student, we wonder: **Can machine learning be a game-changer in the world of investing?** By leveraging advanced algorithms and vast datasets, can we develop models that can identify profitable trading opportunities and outperform the benchmarks?

4. What are we doing for our project?

For our project, we aim to optimize investment timing using a machine learning model. Instead of traditional buy-and-hold strategies or investing at fixed intervals, our model will identify the best times to buy a specific stock. To simplify data acquisition, we'll focus on technical analysis indicators derived from historical stock price data. The model will classify each trading day into two categories: "Up Opportunity," indicating a favorable buying point, and "No Opportunity" otherwise.

Data

1. Sourcing

For this project, we retrieved **Tesla Inc. (TSLA)** historical stock market data from the **yfinance** library as our primary dataset. Tesla was selected for its market prominence and volatility, offering a rich dataset to evaluate predictive models. The dataset includes daily stock performance metrics like Close and Volume which serve as the foundation for engineering technical indicators used in subsequent modeling stages. The table below summarizes an overview of the original dataset.

Data Source	yfinance Python library
Description	Daily historical data
Key Features	Date/Open/High/Low/Close/Volume/Dividends/Stock Splits
Size	3632 data points spanning more than 14 years
Data Type	Numerical/Temporal

Throughout the analysis, we employed only Python libraries for data manipulation (**pandas** and **numpy**), machine learning model construction and evaluation (**scikit-learn**), and data visualization (**matplotlib** and **seaborn**).

2. Preprocessing - Variable Engineering

After collecting the historical stock data for Tesla, we engineered technical indicator features and the target variable to be used in training our prediction models in subsequent sections.

Target Variable: Signal

Our target variable, Signal, was designed as a binary classification to identify trading opportunities where

- **1 (Up Opportunity):** Indicates a favorable trading opportunity when the future return exceeds a specified threshold.
- **0 (No Opportunity):** Represents cases without significant trading opportunities.

The 4 steps in creating our target variable are as follows:

Step 1: Calculate Future Returns

$$\text{Future Return}_t = \frac{\text{Close}_{t+5} - \text{Close}_t}{\text{Close}_t} \times 100$$

Step 2: Set threshold for "Up Opportunity"

Threshold = Average future return x 1.5

Step 3: Create Signal column

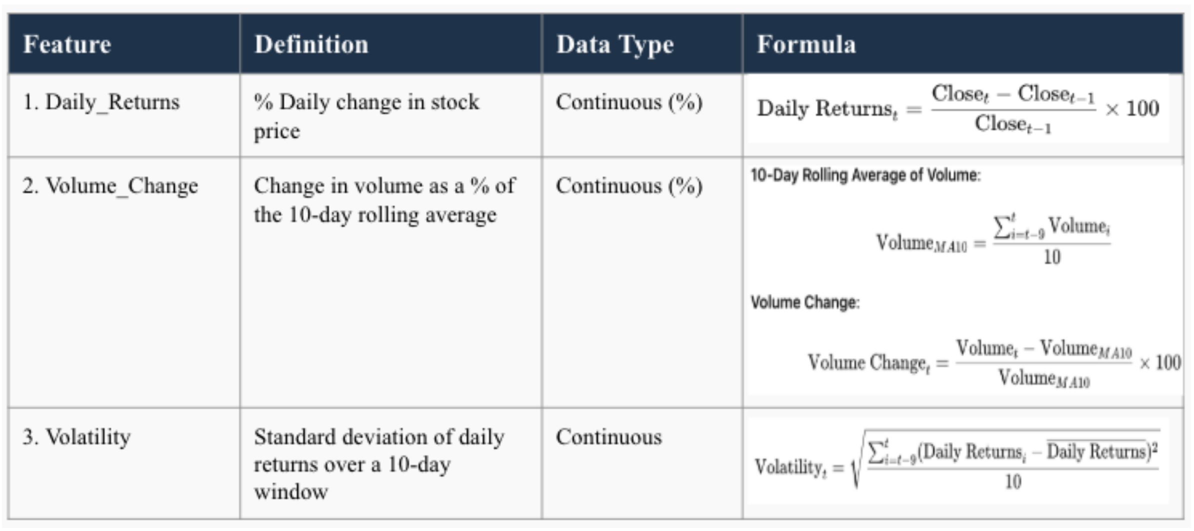
$$\text{Signal}_t = \begin{cases} 1 & \text{if Future Return}_t \geq \text{Threshold}_{u} \\ 0 & \text{otherwise} \end{cases}$$

Step 4: Drop rows where we cannot calculate Future Return due to shifting

With this approach, we aim to simplify the predictive task to a binary classification problem and align with short-term trading strategies by focusing on a 5-day horizon for identifying potential opportunities.

Features

Next, six features were engineered to enhance the model's predictive capability, including 5 continuous and 1 categorical features. We selected these features to incorporate both lagging indicators, such as moving averages and volatility, and momentum-based measures like RSI. Additionally, features like Volume_Change and Day_of_Week were included to provide contextual information, improving the model's ability to account for market behavior. The description of features are summarized in the 2 tables below:

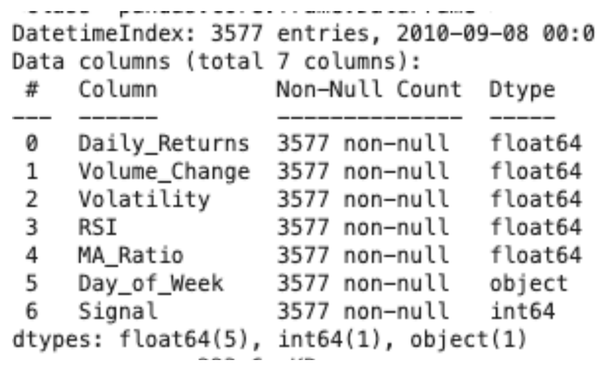


Feature	Definition	Data Type	Formula
1. Daily_Returns	% Daily change in stock price	Continuous (%)	$\text{Daily Returns}_t = \frac{\text{Close}_t - \text{Close}_{t-1}}{\text{Close}_{t-1}} \times 100$
2. Volume_Change	Change in volume as a % of the 10-day rolling average	Continuous (%)	10-Day Rolling Average of Volume: $\text{Volume}_{MA10} = \frac{\sum_{i=t-9}^{t} \text{Volume}_i}{10}$ Volume Change: $\text{Volume Change}_t = \frac{\text{Volume}_t - \text{Volume}_{MA10}}{\text{Volume}_{MA10}} \times 100$
3. Volatility	Standard deviation of daily returns over a 10-day window	Continuous	$\text{Volatility}_t = \sqrt{\frac{\sum_{i=t-9}^{t}(\text{Daily Returns}_i - \overline{\text{Daily Returns}})^2}{10}}$

Feature	Definition	Data Type	Formula
4. MA_Ratio	Ratio of 10-day to 50-day moving averages on price	Continuous	$\text{MA Ratio}_t = \frac{\text{MA}_{\text{Short},t}}{\text{MA}_{\text{Long},t}}$
5. RSI (Relative Strength Index)	Momentum indicator: Measures the magnitude of price gains vs losses over 14-day window	Continuous	**Relative Strength (RS):** $\text{RS}_t = \frac{\text{Gain}_t}{\text{Loss}_t}$ **RSI Formula:** $\text{RSI}_t = 100 - \frac{100}{1 + \text{RS}_t}$
6. Day of Week	Monday,..., Sunday	Categorical	Extracted from the Date index -> Create dummies

Our choice of rolling windows ranges from 5 to 14 days for certain features. These windows are common in practice and short enough to capture meaningful trends in the data while smoothing out daily noise. However, it should be noted that applying rolling windows results in the loss of data points due to the removal of rows with missing values.

3. Creating Final Dataset



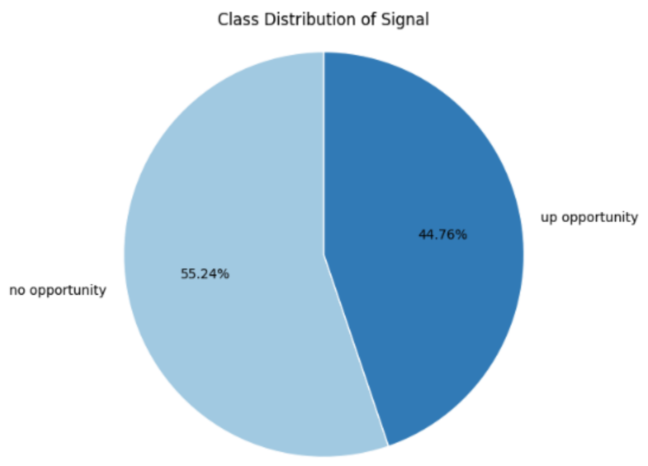
```
DatetimeIndex: 3577 entries, 2010-09-08 00:0
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Daily_Returns  3577 non-null   float64
 1   Volume_Change  3577 non-null   float64
 2   Volatility     3577 non-null   float64
 3   RSI            3577 non-null   float64
 4   MA_Ratio       3577 non-null   float64
 5   Day_of_Week    3577 non-null   object
 6   Signal         3577 non-null   int64
dtypes: float64(5), int64(1), object(1)
```

Building on the engineered features and target variable, the clean final dataset was constructed, including 7 columns. However, the data points were reduced from 3632 to 3577 after preprocessing due to dropping missing value rows.

Exploratory Analysis

With the final dataset prepared, an exploratory analysis was conducted to assess the characteristics of the target variable and the selected features. This step ensures a deeper understanding of the data structure and relationships and provides a rationale for why we will use tree-based classifier models in later sections.

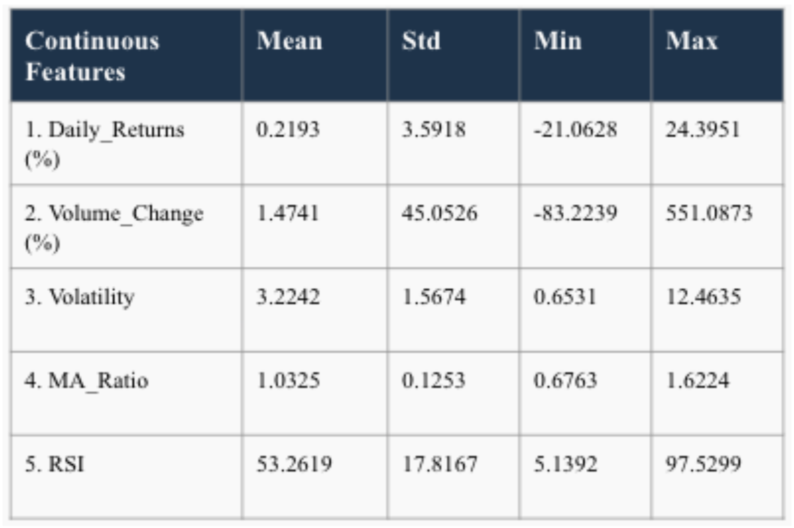
1. Class Distribution



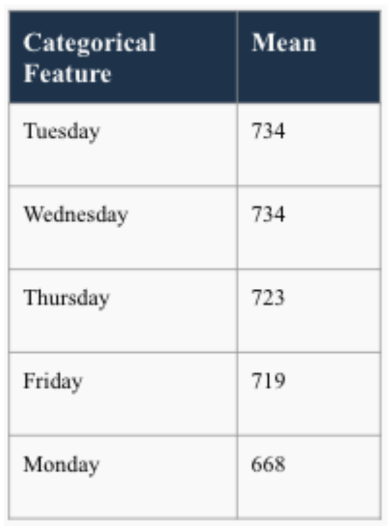
From the pie chart, the class distribution of the target variable is relatively balanced, with 44.76% of instances classified as "up opportunity" and 55.24% as "no opportunity." This balance enhances the robustness of classifier models by reducing the likelihood of class imbalance issues.

2. Feature Statistics

The table below summarizes basic statistics for each feature whereas the boxplots visualize the distribution of them.



Continuous Features	Mean	Std	Min	Max
1. Daily_Returns (%)	0.2193	3.5918	-21.0628	24.3951
2. Volume_Change (%)	1.4741	45.0526	-83.2239	551.0873
3. Volatility	3.2242	1.5674	0.6531	12.4635
4. MA_Ratio	1.0325	0.1253	0.6763	1.6224
5. RSI	53.2619	17.8167	5.1392	97.5299

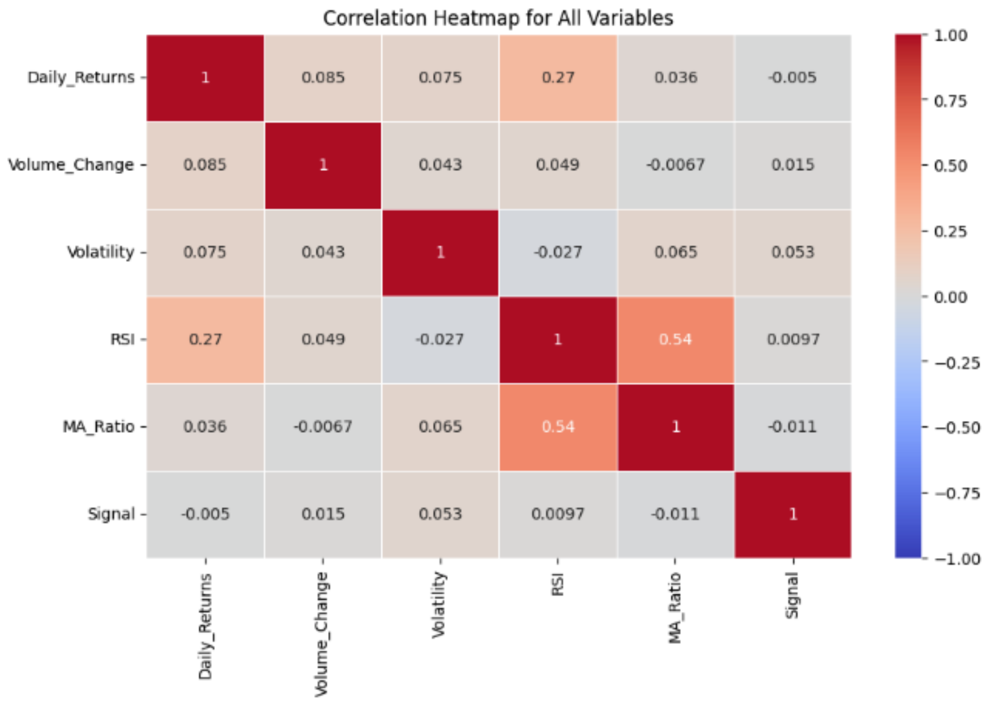


Categorical Feature	Mean
Tuesday	734
Wednesday	734
Thursday	723
Friday	719
Monday	668

It is prevalent that Volume_Change, Volatility, and MA_Ratio are positively skewed, with Volume_Change displaying extreme outliers. Despite these characteristics, no transformations or scaling were performed after this step, as tree-based models are insensitive to the distribution of individual features.

3. Correlation

The correlation heatmap below illustrates a low correlation the target variable and the features, suggesting non-linear relationships. This reaffirms our choice of the suitability of tree-based classifiers as they can effectively capture complex interactions.



Analysis

1. Machine Learning Methods

Given the complex and dynamic nature of financial markets, we selected tree-based ensemble methods for their ability to capture non-linear relationships between our target variable and its features, handle both numerical and categorical features, and manage outliers effectively. The 3 models we employed are as follows:

- **Decision Tree:** repeatedly divides the data into smaller groups based on the features and is easy to understand.
- **Random Forest:** builds many decision trees using different parts of the training data and different sets of features to reduce errors and thus is better at predicting new, unseen data.
- **Gradient Boosting:** builds trees one after another, where each new tree tries to correct

the mistakes made by the previous trees, leading to more accurate predictions.

2. Implementation

Our analysis involved splitting the data, training the models, fine-tuning them, and evaluating their performance. In terms of data splitting, we used two different ways to split the data into training and testing sets, 70% and 30%, respectively:

- **Random Splitting:** We randomly divided the data using a tool called **train_test_split** from the scikit-learn library. This ensured that the training and testing sets were representative of different periods. The test set had a fairly balanced class distribution (593/481).
- **Chronological Splitting:** To better reflect how these models would be used in the real world, we also split the data by time, using older data for training and newer data for testing. This helped us see how well the models could predict future stock behavior. The test set also had a relatively balanced class distribution (617/457).

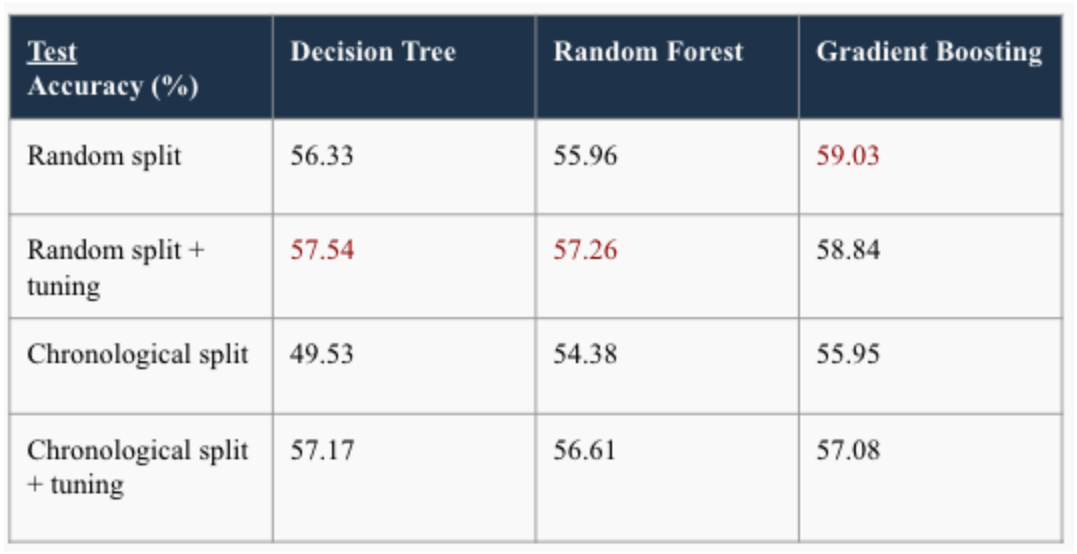
For each train and test set of the two split methods, we initially trained each of the three models, including Decision Tree, Random Forest, and Gradient Boosting, using their default parameters. After evaluating the models using **classification_report** tool, hyperparameter tuning was implemented by leveraging a technique called RandomizedSearchCV to acquire a set of parameters that provides the possible highest performance. To ensure the tuning process was reliable, we utilized a method called **TimeSeriesSplit(n_splits=5)** for cross-validation. This method makes sure that the model is always tested on data that comes after the data it was trained on, similar to real-world stock forecasting. Similarly, we evaluated the new models to compare their performance with prior-tuning counterparts.

As a result, we trained and tested a total of 8 models. Eventually, our final model selection was based on the overall accuracy and stratified accuracy of predictions. We thereafter used the chosen model to evaluate feature importance.

3. Result

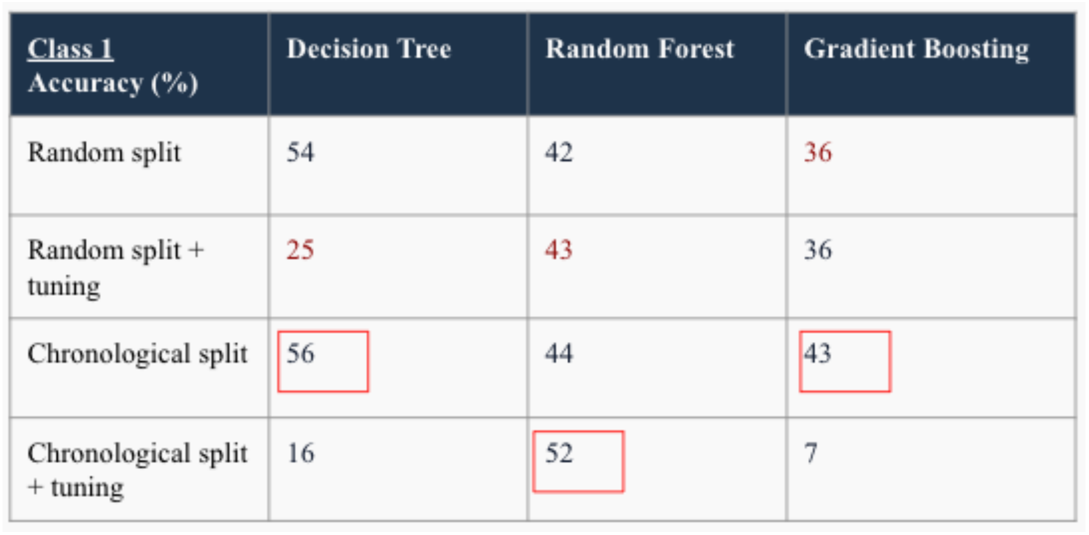
Performance Summary

The following tables summarize the test set accuracy of each model, both with and without hyperparameter tuning, under the two splitting methods:



Test Accuracy (%)	Decision Tree	Random Forest	Gradient Boosting
Random split	56.33	55.96	59.03
Random split + tuning	57.54	57.26	58.84
Chronological split	49.53	54.38	55.95
Chronological split + tuning	57.17	56.61	57.08

As highlighted in the overall accuracy table, Gradient Boosting with random splitting achieved the highest overall accuracy (59.03%) before tuning, followed by Decision Tree and Random Forest. It appears that random splitting generally yielded higher overall accuracy than chronological splitting, possibly indicating that it allows the models to learn a broader range of patterns.



Class 1 Accuracy (%)	Decision Tree	Random Forest	Gradient Boosting
Random split	54	42	36
Random split + tuning	25	43	36
Chronological split	$\boxed{56}$	44	$\boxed{43}$
Chronological split + tuning	16	$\boxed{52}$	7

Nevertheless, the three models with the highest overall accuracy did not achieve the highest stratified accuracy for class 1 (Up Opportunity), as marked in the above table. Specifically, the

Decision Tree with chronological splitting most accurately predicts class 1 (56%) before tuning. In contrast to random splitting, the chronological method appears to be more representative of real-world forecasting of "Up Opportunity" in stock prices. This could be because it offers a more realistic evaluation of the model's ability to predict positive price movements.

Final Model Selection

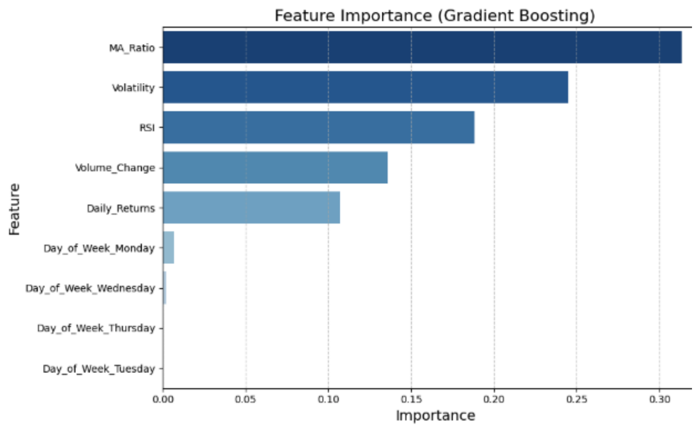
The selection of the most appropriate model involves a trade-off between overall accuracy and class 1 performance. Despite differences in risk tolerance, we believe that the cost of making a wrong investment decision is more damaging than missing on possible returns. Therefore, we selected the Gradient Boosting model with random splitting.

59.03% accuracy	**Precision**	**Recall**	**F-1 score**	**Support**
Class 1	57%	36%	44%	481
Class 0	60%	78%	68%	593

The model exhibited the highest overall accuracy of 59.03% for a strong general predictive capability. While its recall score for class 1 is not the highest, 36%, the model still indicates a reasonable ability to identify positive return opportunities while balancing the need to minimize false positives thanks to a 57% precision.

Feature Importance

In assessing feature importance from the final model, the line graph shows that lagging indicator (MA Ratio), price fluctuation (Volatility), and leading indicator (RSI) are the most influential in short-term price forecasting. Meanwhile, there are no day-specific effects on stock returns.



Limitations & Recommendations

1. All Engineered Variables

Bias in indicator selection and rolling window parameters may limit the model's objectivity and generalizability. Lagging indicators also cause data loss, reducing the dataset size and potentially impacting model performance.

2. Not Robust for Temporal Pattern

Gradient Boosting Classifier does not capture sequential or time-dependent relationships in the data. Advanced time-series models like Long Short-Term Memory (LSTM) networks or other recurrent neural networks could better identify temporal patterns.

3. Not Accounting for More Variables

The model lacks macroeconomic indicators, fundamental metrics (e.g., earnings, P/E ratios), news events, and sector-specific trends, limiting its ability to capture broader market dynamics and real-time or industry-specific influences.

Conclusion

1. Indicator Impacts on Stock Return

Our research indicates that a combination of lagging, leading, and price fluctuation indicators offers valuable insights for short-term stock price forecasting. Specifically, the Moving Average Ratio (MA Ratio), a lagging indicator, helps identify established trends. Volatility, measured by price fluctuations, provides crucial information about market uncertainty and potential risk. Lastly, the Relative Strength Index (RSI), a leading indicator, signals potential overbought or oversold conditions, anticipating future price movements. Interestingly, our analysis also reveals no significant day-specific effects on stock returns, suggesting that consistent application of these technical indicators can yield valuable predictions regardless of the day of the week.

2. Role of the Model in Investment Decisions

It is crucial to remember that our model is not a crystal ball, but rather a tool to assist in investment decision-making. While it can identify potential "Up Opportunities" with a 59% accuracy rate, it should be combined with other forms of analysis. No model can perfectly predict the complexities of the stock market, but even incremental improvements in forecasting accuracy can contribute significantly to more informed and potentially profitable investment strategies.

3. Scenario Analysis

Assumptions:

- Initial Investment: $10,000
- Average Future Return (5 days): 2%
- Threshold for "Up Opportunity": 3% (1.5*Average Future Return)
- Prediction Accuracy: 59%
- Up Opportunity Days: 114 (45% of 252 trading days)
- Capital Allocation Per Trade: Entire portfolio (compounding gains)
- Gain per Correct Trade: +3%
- Loss per Incorrect Trade: -1%

Scenario Calculations:

- Correct Trades: 59%*114=67
- Incorrect Trades: 41%*114=47
- Final Portfolio Value = 10,000*((1.03)^(67))*((0.99)^(47)) = 49,375
- Annual Return Rate = 393%

While achieving 100% accuracy is unrealistic, it's worth noting that in such a scenario, our model could generate an extraordinary annual return of 28,405%, transforming a $10,000 investment into over $2.8 million.

- Final Portfolio Value = 10,000*((1.03)^(114)) = 2,850,526
- Annual Return Rate = 28,405%

4. Future Work

While these theoretical results suggest a significant potential for exceeding the S&P 500's average annual returns of 10%, it's important to acknowledge that our model was trained exclusively on TSLA data. Therefore, its performance on other stocks remains uncertain. To address this and other limitations identified in our analysis, such as the reliance on engineered features and the need for improved temporal pattern recognition, we are actively working to expand the model by incorporating a wider range of variables and data sources. A key next step is to explore alternative machine learning algorithms, such as Long Short-Term Memory (LSTM) networks or Recurrent Neural Networks (RNNs), which are specifically designed to capture temporal patterns in sequential data like stock prices. We believe this will lead to more accurate predictions and further enhance the model's performance.

Bibliography

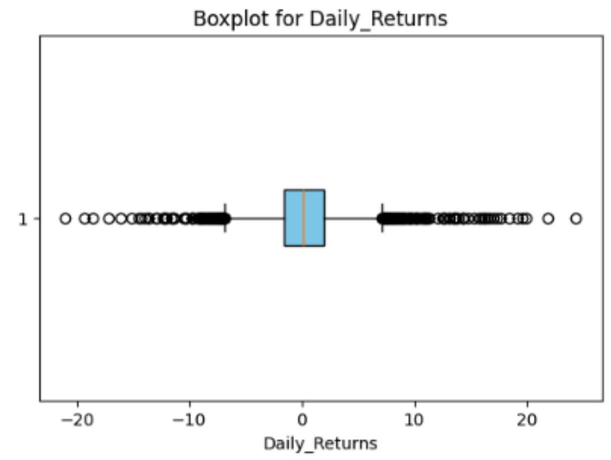
Elton, E. J., Gruber, M. J., Brown, S. J., & Goetzmann, W. N. (2014). *Modern portfolio theory and investment analysis*(9th ed.). Wiley.

S&P Dow Jones Indices. (2021). *SPIVA U.S. Year-End 2021 Scorecard.*

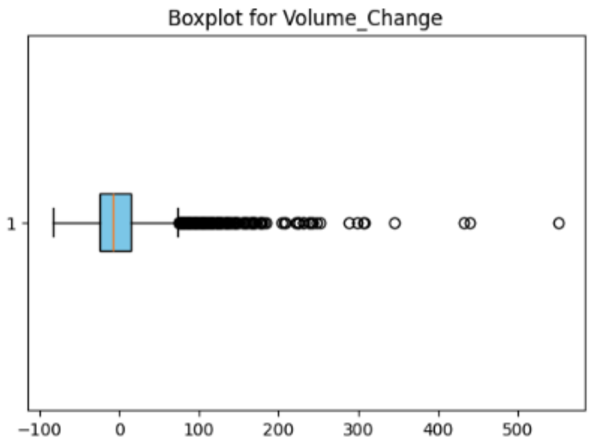
Sezer, O. B., & Ozbayoglu, A. M. (2018, September). Stock price prediction using LSTM, RNN and CNN-sliding window model. In *2018 International Conference on Computer Science and Engineering (UBMK)* (pp. 1-7). IEEE.

Strong, R. A. (1988). Technical analysis and fundamental analysis: Is there really a conflict? *The Journal of Portfolio Management, 14*(3), 43-48.

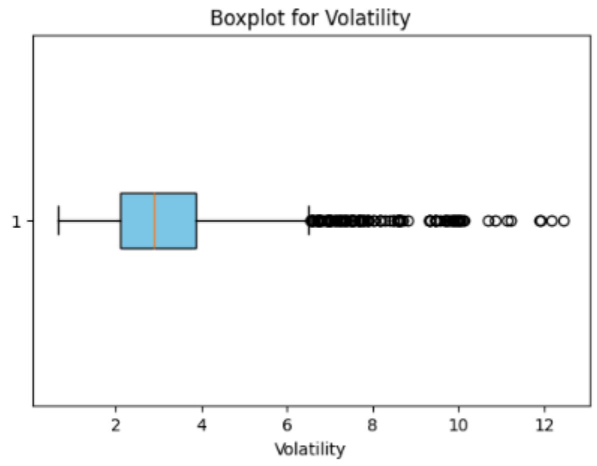
Appendix



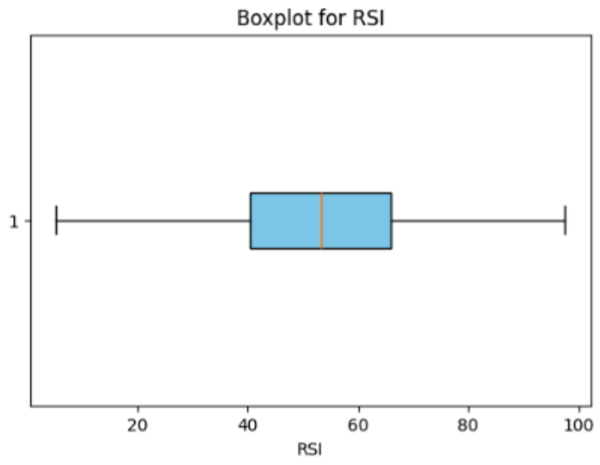
Boxplot for Daily_Returns
1
−20
−10
0
10
20
Daily_Returns



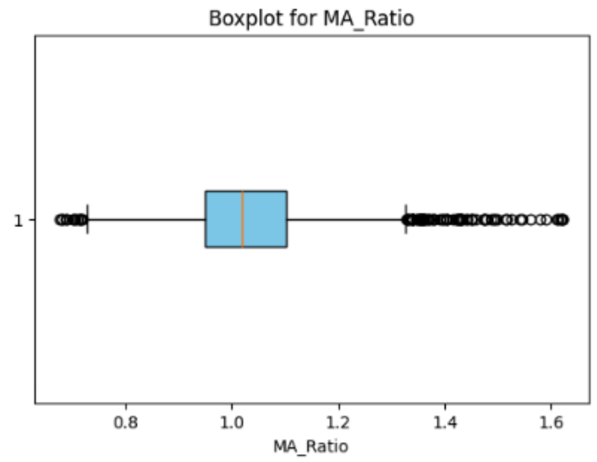
Boxplot for Volume_Change
1
−100
0
100
200
300
400
500



Boxplot for Volatility
1
2
4
6
8
10
12
Volatility



Boxplot for RSI
1
20
40
60
80
100
RSI



Boxplot for MA_Ratio
1
0.8
1.0
1.2
1.4
1.6
MA_Ratio