Stock Sentiment Analysis Using Machine Learning

Report

RAHUL SINGH

Enrollment No: 22117113

This document contains the analysis, report and recommendations related to the project.

Table of Contents

Introduction

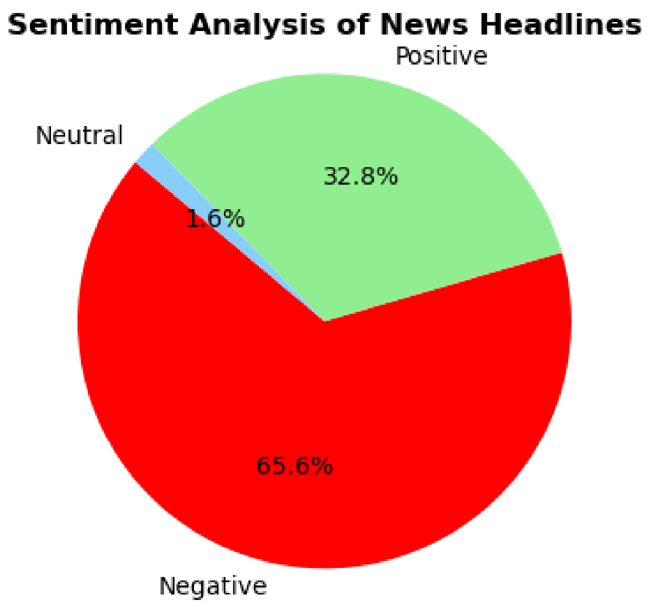
The project aims to develop a sentiment analysis model to predict the movement of stock prices based on textual data from news articles, social media posts, and other sources of financial news and opinions. By analyzing the sentiment expressed in these texts, the model will seek to uncover insights into investor sentiment and market sentiment, which can be valuable indicators for making informed trading decisions.

Dataset

- The dataset consists of News Headlines and Financial Data from 01-01-2023 to 31-12-2023.
- Also there is some extra dataset which can also be used which is from 01-01-2024 to 17-06-2024.
- These datasets are collected by using API's of different sites namely- The New York Times (For News Headlines) and Alpha Vantage (For Financial Data)
- The Datasets are for two stocks- GOOGL (Google Stocks) and MSFT (Microsoft Stocks).

Implementation

- The project starts by collecting news headlines and financial data from websites using web scraping or APIs provided by these websites.
- Then the required libraries are imported into the code.
- The collected news headlines data is preprocessed to remove special characters, normalize text (e.g., convert to lowercase), tokenize sentences into words, remove stopwords, and apply stemming or lemmatization to standardize word forms.
- This preprocessing enhances the quality of data used in the project.
- Then, financial data and news headlines data are merged and stored in the '`merged_df`' dataframe.
- Next, sentiment scores of news headlines on a particular date are calculated using SentimentIntensityAnalyzer(). Based on these values, sentiment categories and movements are determined.
- As the number of neutral sentiments is much less compared to positive and negative sentiments, only two movements are considered: 1 for positive and 0 for negative and neutral.
- Now, the data is split into two datasets: a training dataset (75%) and a testing dataset (25%).
- Models are implemented on the training datasets, and their predictions are made on the testing dataset to predict stock prices and movements.
- Based on the predictions, various calculations such as Accuracy, Precision, F1 score, Recall, ROC AUC, Sharpe Ratio, Maximum Drawdown, No. of Trades Executed, Win Ratio, MAE, and MSE are performed.
- Graphs are plotted based on these predictions and calculations accordingly.



Models Tried (Predicting Stock Price Movements)

The main models tried are:-

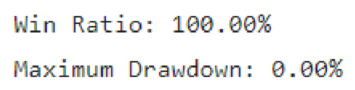
- Random Forest Classifier
- Support Vector Machine (SVM)
- Neural Networks

Random Forest Classifier Model

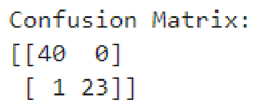
A Random Forest Classifier is an ensemble learning method used for classification tasks, which operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) of the individual trees.

Results:-

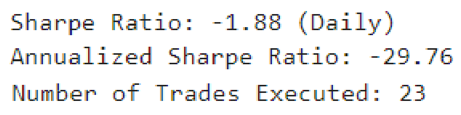
- Accuracy: 0.984375
- Precision: 1.0
- Recall: 0.9583333333333334
- F1-Score: 0.9787234042553191
- ROC AUC: 0.9953124999999999



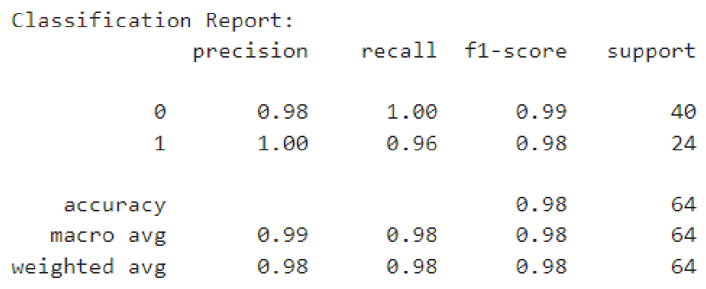
```
Win Ratio: 100.00%
Maximum Drawdown: 0.00%
```



```
Confusion Matrix:
[[40  0]
 [ 1 23]]
```



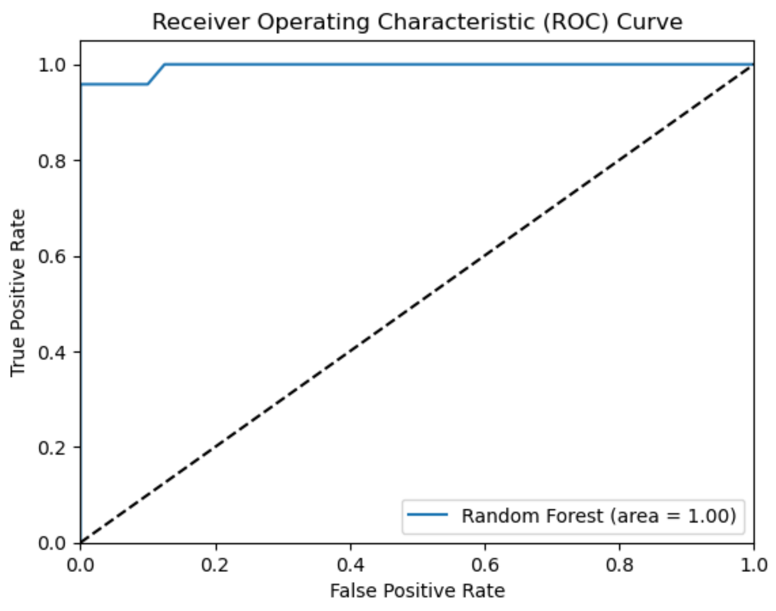
```
Sharpe Ratio: -1.88 (Daily)
Annualized Sharpe Ratio: -29.76
Number of Trades Executed: 23
```



```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        40
           1       1.00      0.96      0.98        24

    accuracy                           0.98        64
   macro avg       0.99      0.98      0.98        64
weighted avg       0.98      0.98      0.98        64
```



Support Vector Machine (SVM) Model

A Support Vector Machine (SVM) is a supervised machine learning model used for classification and regression tasks. The primary objective of an SVM is to find the optimal hyperplane that best separates the data into different classes. This is achieved by maximizing the margin between the closest data points of each class, known as support vectors.

Results:-

- Accuracy: 0.640625
- Precision: 0.6666666666666666
- Recall: 0.08333333333333333
- F1-Score: 0.14814814814814814
- ROC AUC: 0.39166666666666666

```
SVM Win Ratio: 66.67%
SVM Maximum Drawdown: 0.00%
```

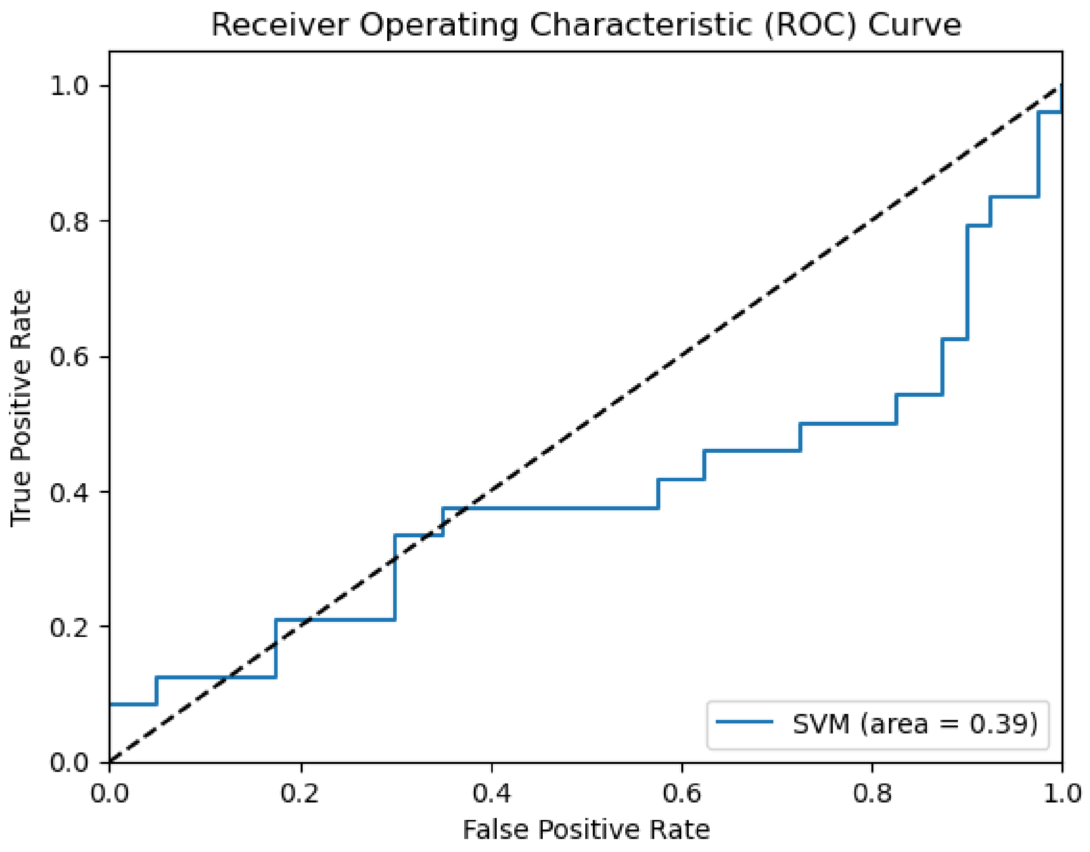
```
Confusion Matrix:
[[39  1]
 [22  2]]
```

```
SVM Sharpe Ratio: -5.52 (Daily)
SVM Annualized Sharpe Ratio: -87.69
SVM Number of Trades Executed: 3
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.97      0.77        40
           1       0.67      0.08      0.15        24

    accuracy                           0.64        64
   macro avg       0.65      0.53      0.46        64
weighted avg       0.65      0.64      0.54        64
```



Neural Networks Model

A Neural Network is a computational model inspired by the human brain, designed to recognize patterns and learn from data. It consists of interconnected layers of nodes, or neurons, where each node represents a mathematical function. Neural networks are powerful tools for deep learning, particularly in areas requiring complex feature extraction and high-dimensional data analysis.

Results:-

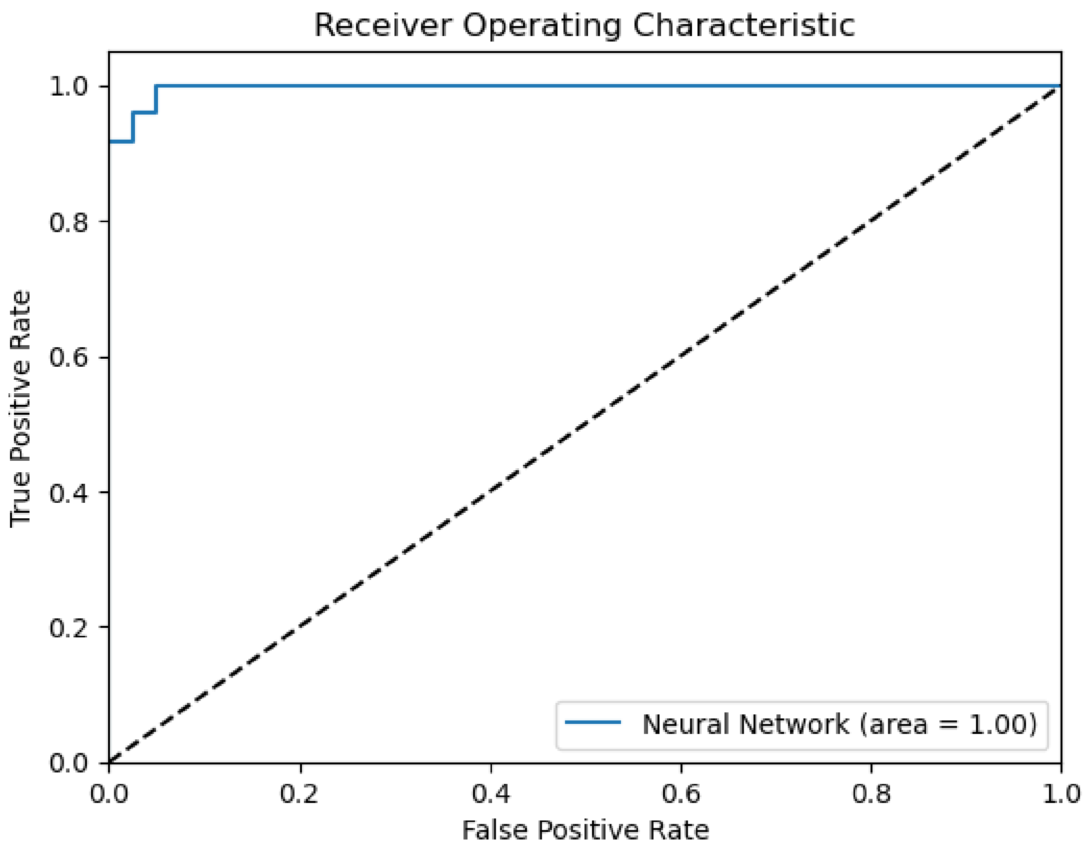
- Accuracy: 0.953125
- Precision: 0.9565217391304348
- Recall: 0.9166666666666666
- F1-Score: 0.9361702127659574
- ROC AUC: 0.9947916666666666

```
Confusion Matrix:
[[39  1]
 [ 2 22]]
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.97      0.96        40
           1       0.96      0.92      0.94        24

    accuracy                           0.95        64
   macro avg       0.95      0.95      0.95        64
weighted avg       0.95      0.95      0.95        64
```



Final Model Used (Predicting Stock Price Movements)

Random Forest Classifier Model

- *Model Architecture Overview:*
  The model uses an ensemble of decision trees to perform classification. Each tree is trained on a random subset of the data and uses random feature selection at each split to reduce overfitting and increase robustness.

- *Bootstrapping:*
  Creates multiple subsets of the original training data by sampling with replacement. Each subset is used to train a different decision tree.

- *Random Feature Selection:*
  Selects random feature subsets at each split to reduce correlation between trees.

- *Aggregation:*
  For classification tasks, each tree in the forest votes for a class, and the majority vote determines the final prediction.

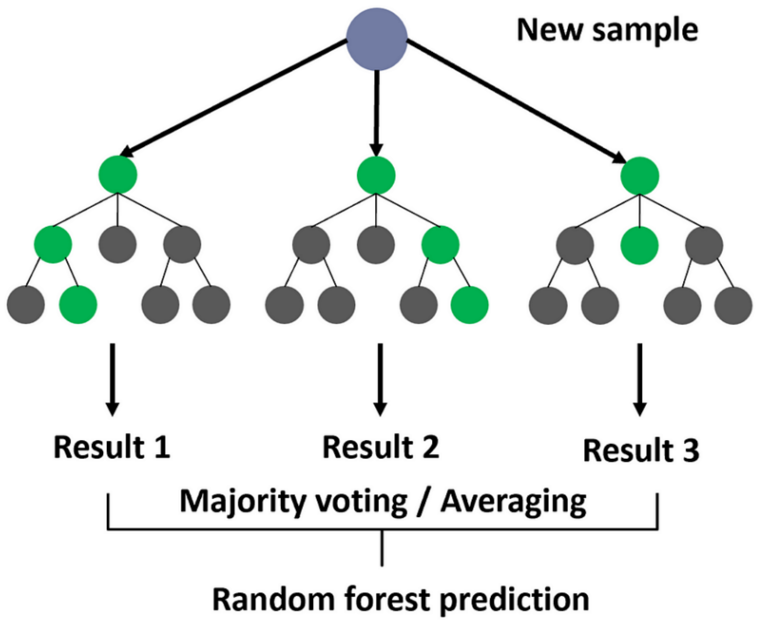
- *Bias-Variance Tradeoff:*
  Balances bias and reduces variance through ensemble averaging.

- *Feature Importance:*

  Random forests provide measures of feature importance, indicating which features are most influential in predicting the target variable.

- *Parallelization:*

  The training of multiple trees can be done in parallel, making the model training process faster and more efficient.



Code:

- **Importing Libraries**

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score, roc_auc_score, roc_curve,
confusion_matrix,classification_report
import matplotlib.pyplot as plt
```

- **Initializing Random Forest Classifier**

```
rf=RandomForestClassifier(random_state=42)
```

- **Prepare Training Data**

```
X_train = train[['Open', 'High', 'Low', 'Volume',
'Positive', 'Negative','Neutral']]
y_train = train['Movement']
```

- **Fitting Training Dataset in Random Forest Classifier Model**

```
rf.fit(X_train, y_train)
```

- **Prepare Testing Data and Predictions**

```
X_test = test[['Open', 'High', 'Low', 'Volume', 'Positive',
'Negative','Neutral']]
predictions_rf = rf.predict(X_test)
predictions_prob_rf = rf.predict_proba(X_test)[:, 1]
```

- **Calculate Evaluation Metrics**

```
accuracy_rf = accuracy_score(test['Movement'],
predictions_rf)
precision_rf = precision_score(test['Movement'],
predictions_rf)
recall_rf = recall_score(test['Movement'], predictions_rf)
f1_rf = f1_score(test['Movement'], predictions_rf)
roc_auc_rf = roc_auc_score(test['Movement'],
predictions_prob_rf)
```

```
# Assuming predictions_svm are the predicted movements (-1 for
sell, +1 for buy)
daily_returns_svm = test['Movement'] * predictions_svm - 1
daily_returns_svm.fillna(0, inplace=True)
sharpe_ratio_svm = daily_returns_svm.mean() /
daily_returns_svm.std()
annual_sharpe_ratio_svm = sharpe_ratio_svm * np.sqrt(252)
print(f"SVM Sharpe Ratio: {sharpe_ratio_svm:.2f} (Daily)")
print(f"SVM Annualized Sharpe Ratio:
{annual_sharpe_ratio_svm:.2f}")

daily_returns_svm = test['Movement'] * predictions_svm - 1
cumulative_returns_svm = daily_returns_svm.cumsum()
max_drawdown_svm = (cumulative_returns_svm /
cumulative_returns_svm.cummax() - 1).min()
print(f"SVM Maximum Drawdown: {max_drawdown_svm:.2%}")

num_trades_svm = np.count_nonzero(predictions_svm)
print(f"SVM Number of Trades Executed: {num_trades_svm}")

# Assuming predictions_svm are the predicted movements (-1 for
sell, +1 for buy)
profit_loss_per_trade = test['Movement'] * predictions_svm
num_winning_trades = np.sum(profit_loss_per_trade > 0)
win_ratio_svm = num_winning_trades / num_trades_svm if
num_trades_svm > 0 else 0
print(f"SVM Win Ratio: {win_ratio_svm:.2%}")
```

Models Tried (Predicting Stock Prices)

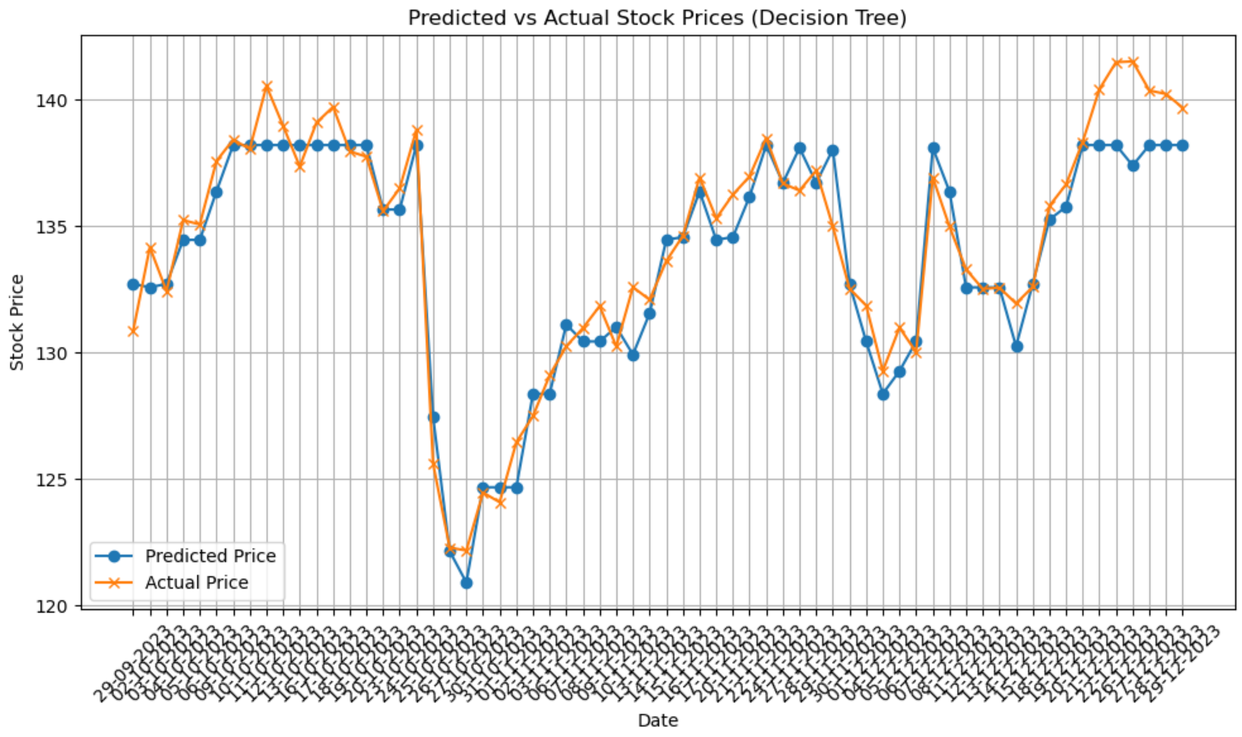
Decision Tree Regressor:

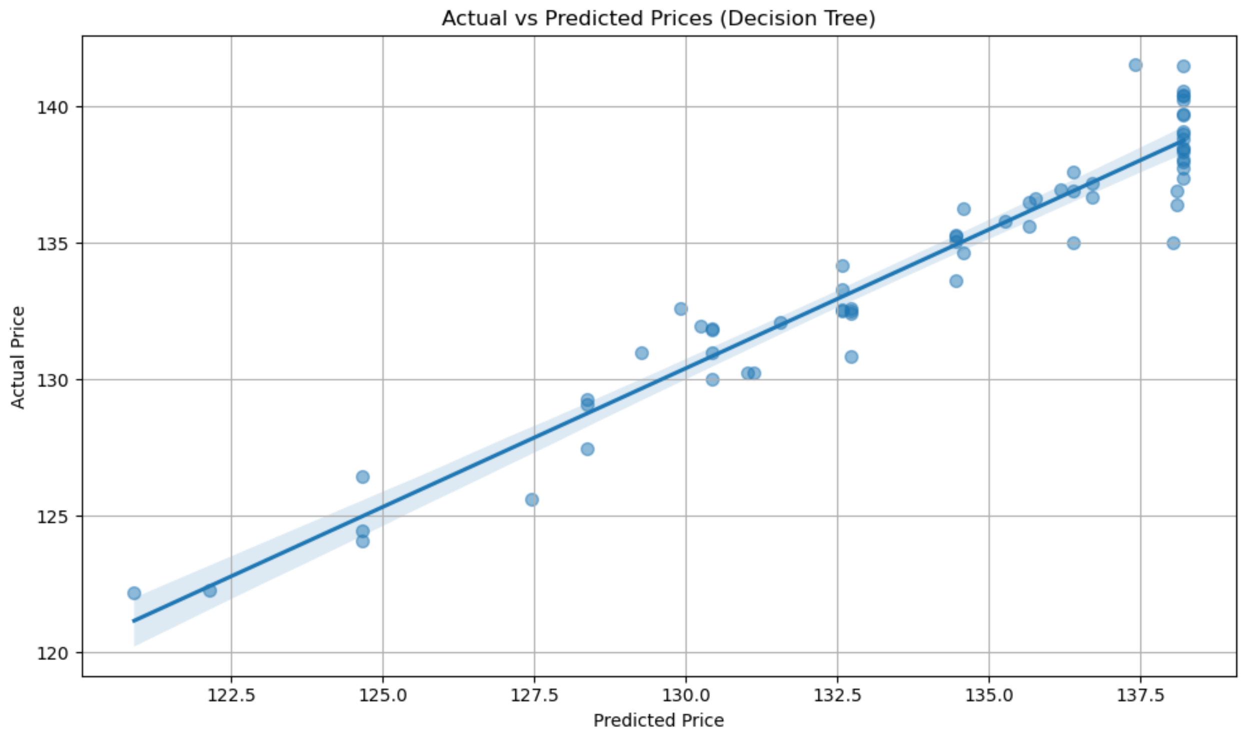
A Decision Tree Regressor is a machine learning model used for predicting continuous numerical values. It builds a tree-like structure where each internal node represents a decision based on a feature, and each leaf node represents a predicted value.

Results:-

Decision Tree Mean Squared Error: 1.806934765624994

Decision Tree Mean Absolute Error: 1.0372656249999983

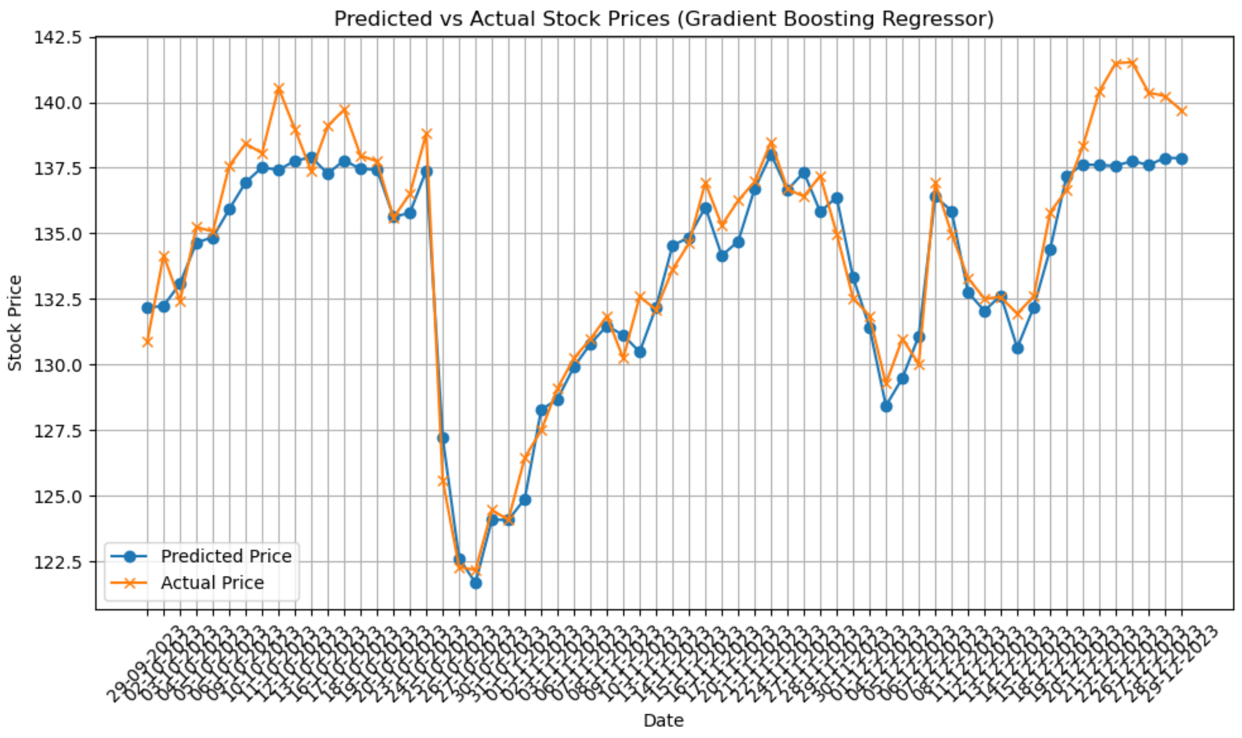


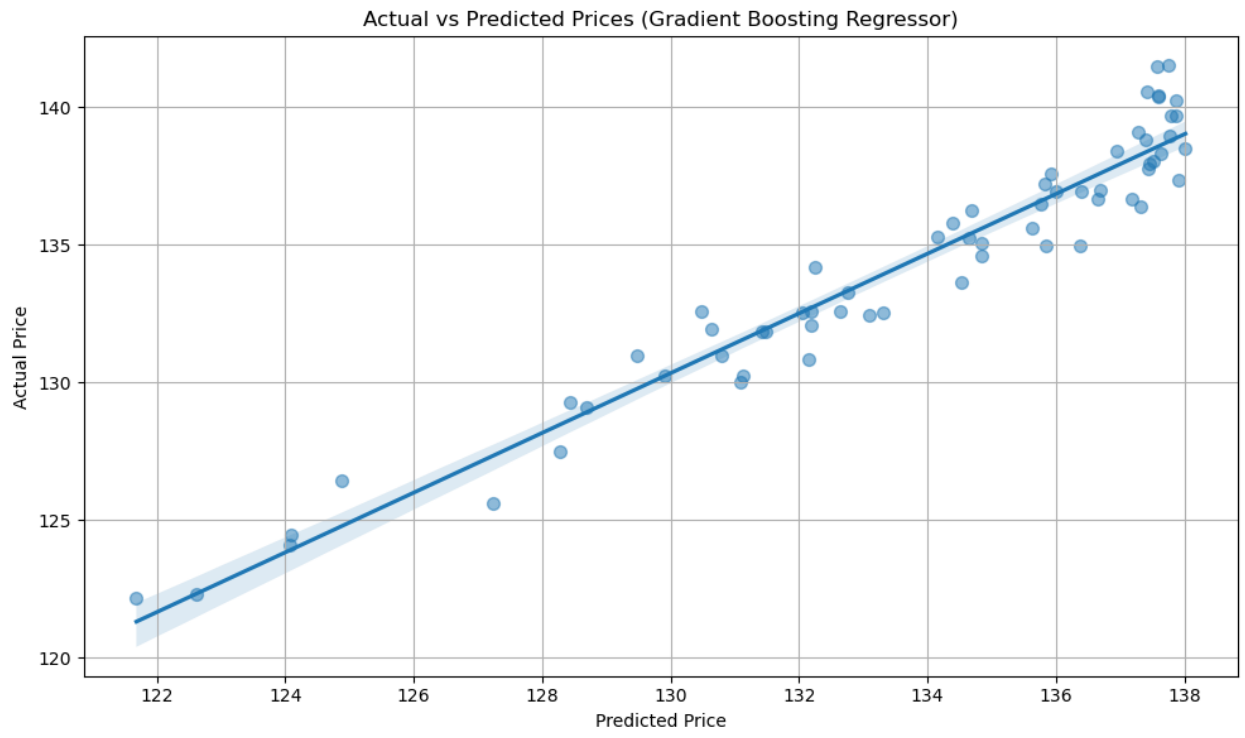


Gradient Boosting Regressor:

Gradient Boosting Regressor is a powerful machine learning technique that builds an ensemble of decision trees sequentially. It corrects the errors made by the previous trees and improves the model's accuracy iteratively.

Results:-
Gradient Boosting Mean Squared Error: 1.8795789814192767
Gradient Boosting Mean Absolute Error: 1.0577908588233875





Potential Improvements:

- Twitter data should also be utilized as it can influence stock price movements. I was unable to extract data from Twitter due to API limits.
- Changes in market conditions, investor behavior, and external events should influence stock price movements and should also be considered.

Application

- Investment Decision Making
- Risk Management
- Market Research and Strategy

References

- https://www.investopedia.com/terms/s/sentimentindicator.asp
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8659448/
- https://www.insightbig.com/post/stock-market-sentiment-prediction-with-openai-and-python
- Stock Market Sentiment Analysis Using Python & Machine Learning