# Stock Market Prediction using Machine Learning

1st Jaheem Edwards
*dept. of computing and information technology*
*University of the West Indies*
St. Augustine, Trinidad and Tobago
jaheem.edwards@my.uwi.edu

2nd Liam Ramjewan
*dept. of computing and information technology*
*University of the West Indies*
St. Augustine, Trinidad and Tobago
liam.ramjewan@my.uwi.edu

*Abstract*— **In the complicated and unpredictable world of financial markets, individual investors have considerable hurdles in digesting huge volumes of raw data and making educated judgments. This work covers the design and implementation of a web-based application that merges stock market prediction with data visualization utilizing historical and real-time data from S&P 500 firms. Leveraging machine learning models such as Random Forest, Linear Regression and XGBoost alongside technical indicators and sentiment analysis of financial news, the system intends to give accessible and actionable insights for both instructional and practical application. The program, created using Streamlit, has interactive dashboards that enable users to examine market patterns, assess individual stock performance, and review sentiment-driven projections. Despite extensive integration of multiple data sources, results suggest that sentiment analysis did not substantially boost model accuracy, underscoring the challenges of textual data interpretation in financial situations. The project stresses instructional value via its open-source design and modular architecture, highlighting the actual use of Big Data, machine learning, and web technologies in financial analytics.**

***Keywords—Stock Market Prediction, Big Data Analytics, Financial Visualization, Sentiment Analysis, Machine Learning, Streamlit, S&P 500, Web Application, Technical Indicators, Predictive Modeling.***

## I. INTRODUCTION (*HEADING 1*)

### A. *Problem Statement*

In the ever evolving and unpredictable world of financial markets, making educated investing choices is a continual struggle, particularly for individual investors. Although much market data is publicly accessible, accurate interpretation requires technical skill, specialized tools, and access to forecasting models. Conventional tools often lack clear interfaces and actionable insights from real-time data, resulting in a disconnect between raw market information and effective investing strategies.

This project seeks to address that gap by creating a web application that employs historical and real-time stock data from S&P 500 businesses to facilitate stock market prediction. The program analyses possible market trends and visualizes data via interactive dashboards, allowing users to examine top-performing stocks or search for individual firms to get granular stock information. This amalgamation of forecasting and visualization fulfils the need for accessible, data-informed investing assistance.

### B. *Project Motivation*

The reason for this project derives from the increased need for accessible financial solutions that utilize Big Data and predictive analytics. With the development of retail investment and interest in algorithmic trading, more consumers are seeking systems that simplify complicated market data. Our group spotted an opportunity to build a product that provides users with both foresight and insight combining stock prediction algorithms with interactive visualization capabilities in a single, user-friendly online interface. This connection enables users not just to forecast market trends but also to examine and understand the parameters behind such predictions.

Unlike many commercial systems that concentrate primarily on analytics or visual exploration, our program unites both functions, making it particularly helpful for customers who seek instant, actionable information without requiring additional tools. Additionally, the project uses an open-source and instructional approach, aiming to benefit students, researchers, and independent developers. By making the codebase and technique open, we seek to stimulate learning, repeatability, and future innovation in the area of financial data analysis.

Moreover, this project acts as a realistic application of Big Data technologies, enabling the team to use ideas such as data engineering, machine learning, and data visualization in a real-world setting.

## II. RELATED WORK

Numerous platforms and tools exist for financial market research, each providing distinct mixes of stock data visualization, analytics, and in some instances, prediction. Notable examples are Yahoo Finance, Google Finance, TradingView, and Finviz.

Yahoo Finance and Google Finance give searchable stock information with historical charts and financial news, but they do not provide predictive tools or configurable analytics. TradingView excels in technical charting and offers community-driven scripting using Pine Script, although it lacks built-in predictive modelling for customers without sophisticated coding knowledge. Finviz offers strong stock screeners and visuals such as heatmaps but focuses more on exploration than on predicting.

More sophisticated systems like QuantConnect, Alpaca, and Tickeron provide AI-driven forecasts or algorithmic trading capabilities. However, these platforms are generally aimed at experienced developers or come with commercial limits that limit educational usage or model transparency. In addition, systems like Seeking Alpha and Zacks give human-analyst-based projections, which, although informative, do not use the full possibilities of real-time machine learning models.

Compared to them, our project differs itself in numerous significant ways:

- It unifies both prediction and visualization in a single user-friendly online interface.

- It emphasizes educational accessibility, developed as an open-source platform for students and independent researchers.

- It gives openness in technique, revealing the underlying machine learning process and designed features that promote learning and future experimentation.

This mix of interactive insights, predictive modelling, and educational value is not typically seen in current platforms, establishing our application as a hybrid tool for learning, analysis, and decision-making.

## III. Methodology

### A. Data Sources

The cornerstone of this project is founded on gathering and integrating several credible data sources that are routinely utilized in financial forecasting. The data sources are as follows:

- S&P 500 Tickers: The list of firms in the S&P 500 index was collected straight from Wikipedia. This dynamic technique guarantees the ticker list is constantly updated, which is vital for up-to-date analysis.
- Historical Stock Prices: Historical daily stock data for the S&P 500 firms was gathered from Yahoo Finance. This dataset comprises open, high, low, close, adjusted close, and volume numbers for a user specified time frame. The API's accessibility and coverage make it a realistic solution for accurate time-series financial data.
- News Headlines: To integrate textual sentiment, current headlines were gathered from Finviz, a financial visualizations and screening tool. For each ticker, news headlines were retrieved using BeautifulSoup by parsing the Finviz "news-table" component. The headlines were timestamped and matched with their appropriate ticker and date, allowing for sentiment analysis with price movement data.

Each of these data components had a unique purpose: price and volume indicators were utilized as inputs to the prediction models, while the news headlines were subsequently employed in a separate NLP pipeline for sentiment scoring. The mix of numerical and textual data sources allows for a more complete examination of market behavior.

### B. Technologies Used

This project uses a range of programming languages, libraries, frameworks, and tools to assist data gathering, processing, modeling, and deployment. The primary technologies are outlined below:

Languages

- Python: Core programming language used throughout the project for data collection, processing, modeling, and backend logic.

Frameworks and Tools

- Streamlit: Used to build the interactive web interface for the application. It allows quick deployment of data-driven apps with minimal front-end coding.
- Scikit-learn: Used for machine learning model implementation (Random Forest, Linear Regression), data preprocessing (e.g., scaling), and model evaluation.
- XGBoost: An optimized gradient boosting library used for classification tasks, specifically to predict stock price direction based on news headlines.

Data Collection & APIs

- Yahoo Finance (via yfinance): Used for collecting historical stock data, including prices and volumes.
- Web Scraping (BeautifulSoup & requests): Utilized to gather financial news headlines from public sources to perform sentiment analysis.

Natural Language Processing (NLP)

- TextBlob: Used for basic sentiment analysis on news headlines.
- NLTK: Provides tools for text preprocessing (e.g., tokenization, stopword removal) to clean and prepare news data.
- TF-IDF Vectorizer: Converts news headlines into numerical vectors for model input.
- Latent Dirichlet Allocation (LDA): Applied for topic modeling on financial news content.

Data Visualization

- Plotly: Enables interactive charts such as candlestick plots, sentiment distributions, and prediction percentage visuals.
- Matplotlib & Seaborn: Used for static data visualizations during analysis and model evaluation.

### C. Website Architecture

The project website is designed using Streamlit, a Python-based web framework that is well-suited for constructing interactive data applications. The architecture of the system is modular, separating the user interface, data processing, and prediction logic to ensure clarity and maintainability.

The frontend is developed completely in Streamlit, delivering a simple and engaging user experience. Users may browse via the program using a sidebar, which allows access to different sections, including Home, Stock Prediction, Sentiment Analysis, and Topic Modeling. Input options enable users to provide stock tickers and customize appropriate analytic parameters.

The backend handles the processing logic. When a user inputs information, the system obtains historical stock data using the Yahoo Finance API (yfinance) and retrieves relevant news items using web scraping, utilizing tools such as BeautifulSoup and requests. This data is preprocessed by normal text-cleaning approaches and feature engineering procedures, including technical indicator synthesis with the ta library. The processed data is subsequently converted to comply with the input format needed by the prediction models.

The machine learning component loads pre-trained models using Joblib. These models, which include XGBoost and Random Forest, perform both classification and regression tasks to anticipate stock direction and price changes. Predictions combine both historical stock data and current news emotion to boost accuracy.

Visualization of findings is handled by Plotly, which creates interactive charts such as candlestick plots and prediction overlays. Additional static visualizations are made using Matplotlib and Seaborn. These outputs assist users comprehend model forecasts in the light of current market movements.

## IV. IMPLEMANTATION

The implementation of the project comprises of both frontend and backend components, created mostly in Python using the Streamlit framework. The website is intended to be modular and simple to update, with well-defined scripts handling diverse duties such as UI rendering, model inference, data collecting, and visualization.

### A. *Backend and Frontend Details*

The frontend is developed using Streamlit's high-level APIs, which allow quick construction of a multi-page web application. The primary pages: Home, Stock Prediction, Sentiment Analysis, and Topic Modeling are created in distinct Python scripts and accessible using a navigation menu. Each page script is responsible for drawing user input components, running backend services, and showing results using visual elements such as charts, tables, and metrics.

The backend is made of utility modules that handle core logic. These contain routines for obtaining stock data using the Yahoo Finance API (yfinance), scraping news articles using BeautifulSoup, and performing sentiment analysis using pre-trained models from the transformers library. A suite of bespoke modules performs technical indicator computation, data cleaning, and formatting for model input. Model inference is done out using pre-trained Random Forest, Linear Regression and XGBoost models, which are loaded at runtime using Joblib. All backend code is meant to be lightweight and stateless, utilizing Streamlit's caching to enhance speed.

### B. *Data Processing Pipelines*

The data processing pipeline is important to the system's operation. When a user conducts an analysis, the system first obtains historical stock values for the given ticker. This raw time series data is then supplemented with technical indicators like as moving averages, RSI, and MACD using the ta library. Simultaneously, the system gathers relevant news items using keyword-based web scraping, extracts their textual content, and applies a sentiment analysis model to give sentiment ratings.

Both the financial and sentiment datasets are then integrated on a time aligned basis to generate a full feature set. This data is cleaned, standardized, and converted into the input structure needed by the machine learning models. The models then create predictions either as categorical market direction (up/down) or as numerical future prices which are later shown to the user via interactive charts and textual summaries.

This end-to-end pipeline provides a smooth transition from user input to actionable insights, embodying the key value of the platform: to give customers a combined technical and sentiment based view on stock performance.

## V. ANALYSIS AND RESULTS

### A. *Visualizations*

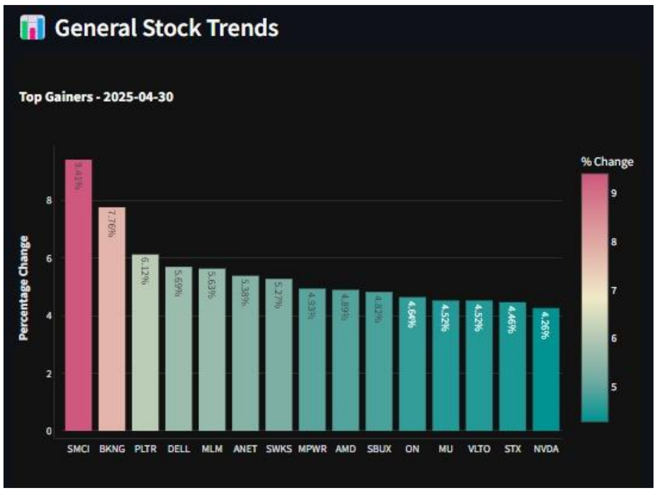


Fig. 1. Bar chart of general stock trends

In Fig. 1 it can be seen that the general stock trends were plotted in a bar graph showing the top 15 gainers for a specified date with the x-axis representing the stock name using its respective ticker and the y-axis represents percentage changed. There is also a key on the right representing the percent change by the use of a color gradient.

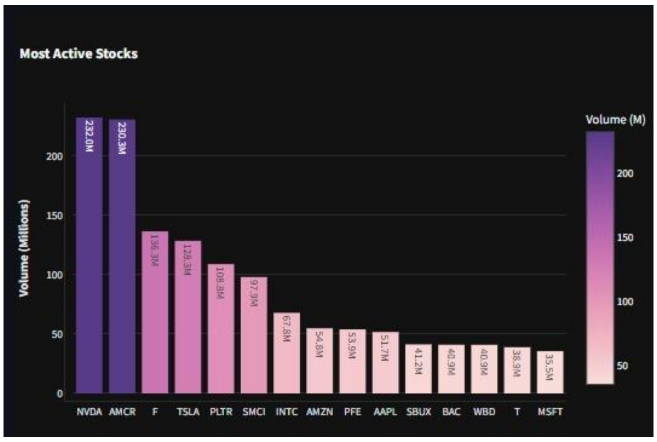


Fig. 2. Bar chart showing top most active stock

In Fig. 2 we see that in the bar chart it shows the top 15 stocks which are most active. The x-axis represents the name of the stock, and the y-axis represents the volume of the stock using a scale of 1 unit to 50 million. There is also a key on the right showing a volume gradient.

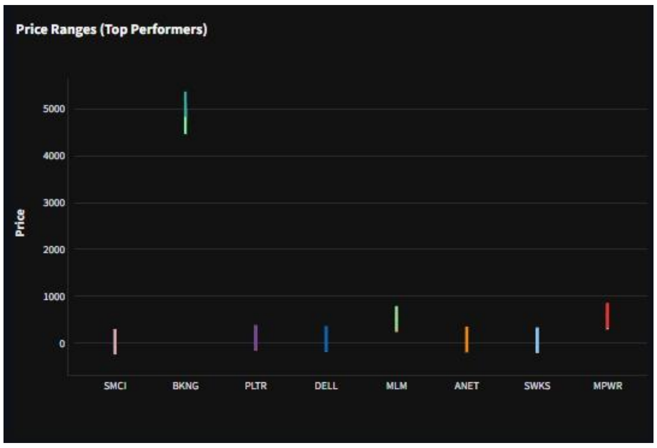


Fig. 3. Graph showing the price range of stocks

Fig. 3 shows the price ranges of the top performing stock in a chart, the price is shown on the y-axis and the names of the stock on the x-axis.

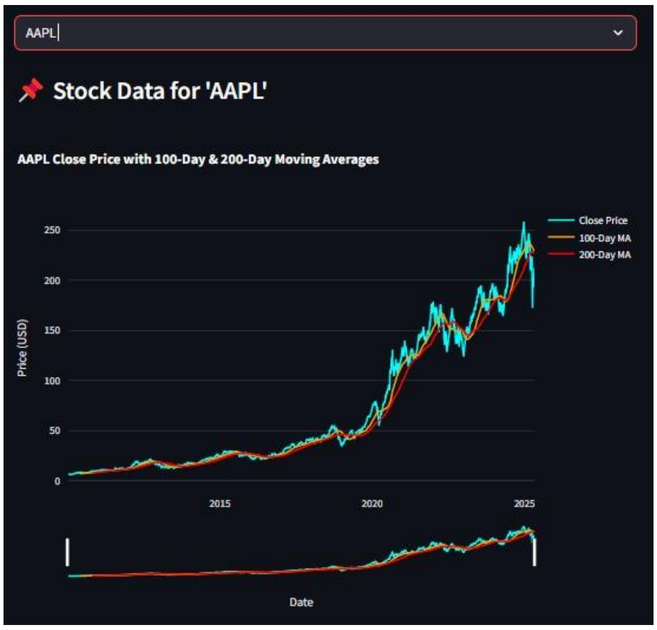


Fig. 4. Graph showing close price and 100-day and 200-day moving average of a selected stock (in this case Apple).

Fig. 4 shows a graph of the Apple stock and it highlights the close price, 100-day moving average and 200-day moving average from the earliest record to the latest record. It can also be send on the top of the graph there is a drop down menu from which a person can select the stock they want to view and below the graph there is a slider which can zoom in or out of the graph to show a specific timeframe.

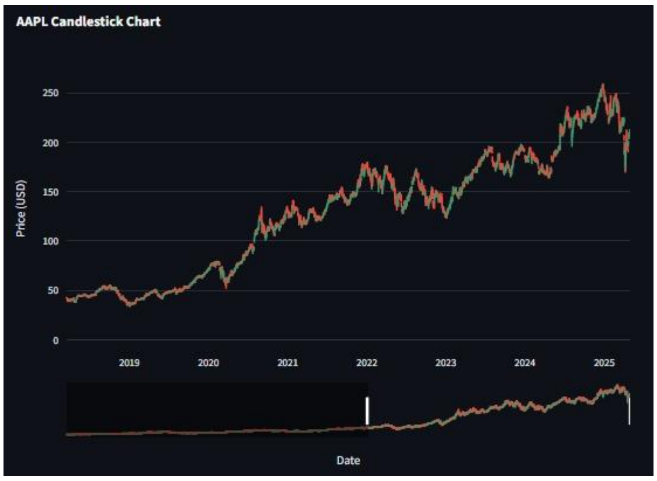
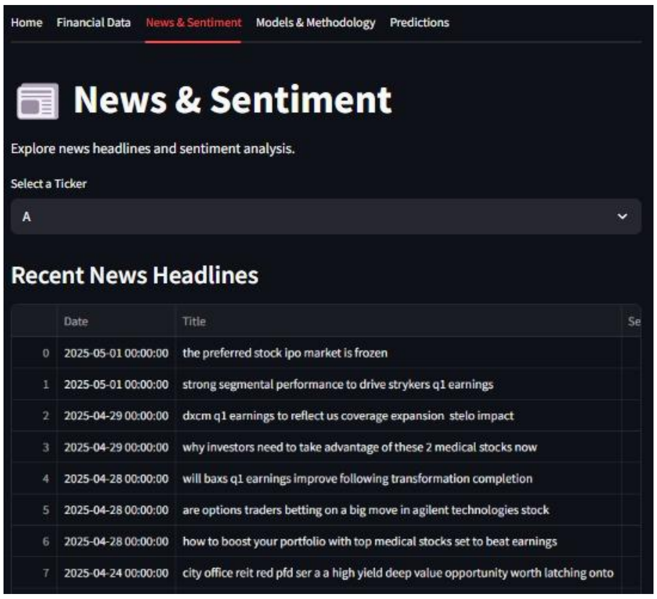


Fig. 5. A candle stick graph for a selected stock (in this case Apple).

Fig. 5 shows a candlestick graph for the Apple stock (ticker AAPL), a candlestick graph shows the high, low, opening, and closing prices of a stock over a time period. The graph in Fig. 5 shows for the earliest date to last recorded date and the scroll bar on the bottom can be used to view a specific time period.



News & Sentiment

Explore news headlines and sentiment analysis.

Select a Ticker: A

Recent News Headlines

| | Date | Title | Se |
|---|---|---|---|
| 0 | 2025-05-01 00:00:00 | the preferred stock ipo market is frozen | |
| 1 | 2025-05-01 00:00:00 | strong segmental performance to drive strykers q1 earnings | |
| 2 | 2025-04-29 00:00:00 | dxcm q1 earnings to reflect us coverage expansion  stelo impact | |
| 3 | 2025-04-29 00:00:00 | why investors need to take advantage of these 2 medical stocks now | |
| 4 | 2025-04-28 00:00:00 | will baxs q1 earnings improve following transformation completion | |
| 5 | 2025-04-28 00:00:00 | are options traders betting on a big move in agilent technologies stock | |
| 6 | 2025-04-28 00:00:00 | how to boost your portfolio with top medical stocks set to beat earnings | |
| 7 | 2025-04-24 00:00:00 | city office reit red pfd ser a a high yield deep value opportunity worth latching onto | |

Fig. 6a. Show the news sentiment section.

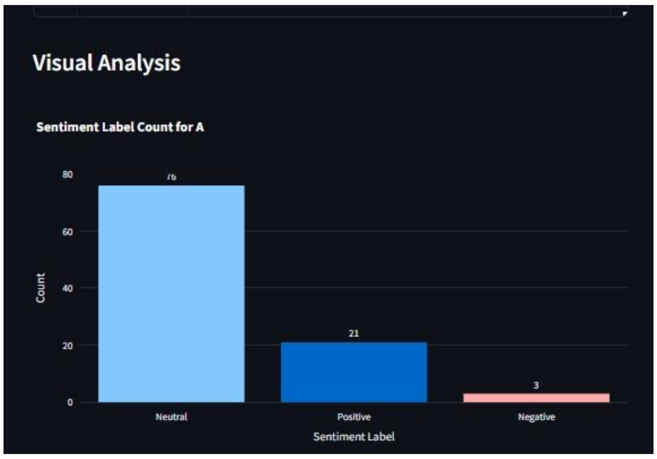


Fig. 6b. Shows an analysis of the new sentiment.

Fig. 6a shows an image of the news sentiment section, in this image it shows all news along with the date it was published about a specified stock. At the top of the image, you can see the drop-down menu where a stock is chosen by its ticker. In Fig. 6b the analysis in the new sentiment in Fig. 6a can be seen as a bar chart where it quantifies and classifies the news as either neutral, positive or negative.

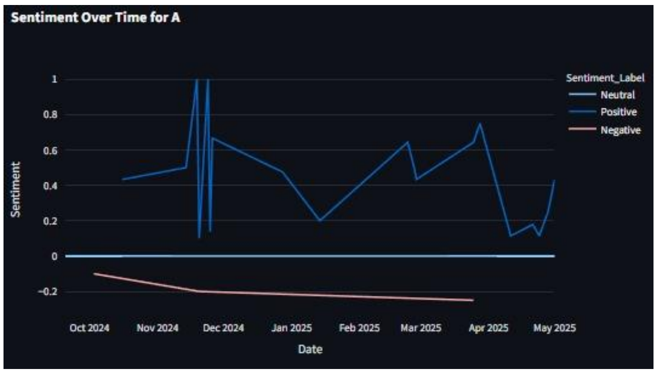


Fig. 7. Line graph showing sentiment over time.

Fig. 7 shows a line graph of sentiment over time for a given stock, it allows for the visualization of how news sentiment affect the stock positively, negatively or not at all.

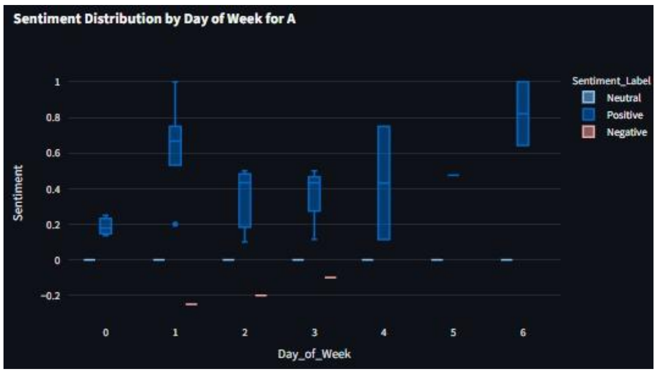


Fig. 8. Graph showing sentiment distribution.

Fig. 8 is a graph which shows the distribution of sentiments for a given stock for a week, it quantifies the sentiment of news articles for each day of the week allowing a user to see how the news may positively, or negatively affect the stock prices.

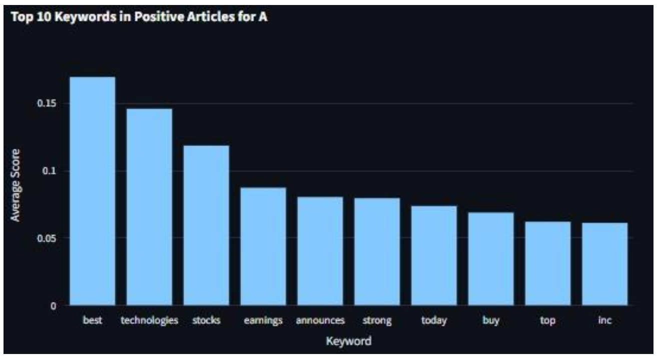


Fig. 9. Shows a graph of the keywords in positive articles.

Fig. 9 is a simple bar graph which shows the top 10 key words which are used in positive news articles for a selected stock. The words are shown on the x-axis and the average score they achieved is shown on the y-axis.

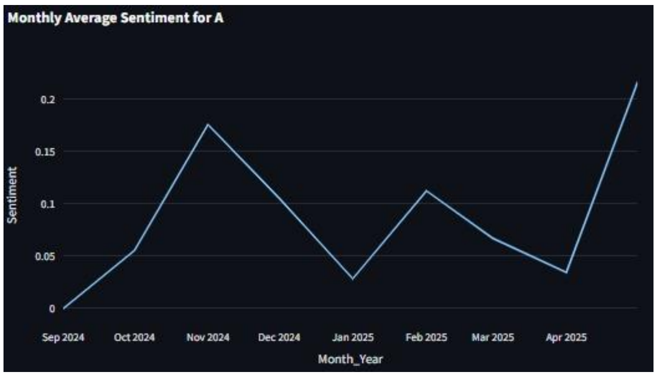
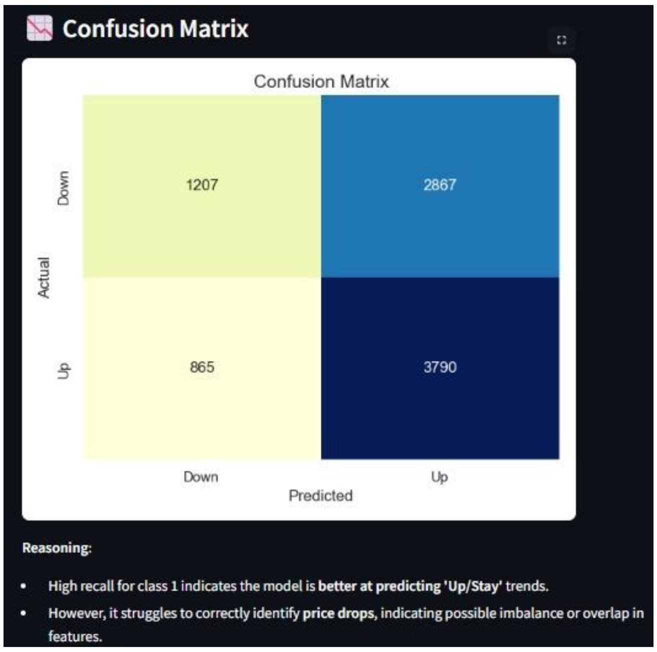


Fig. 10. A line graph of the monthly average sentiment.

The line graph in Fig. 10 shows the average sentiment rating for the news on a particular stock, this would give the user a better insight into how the new sentiment would affect the stock on a monthly basis.

## B. *Performace Metrics*



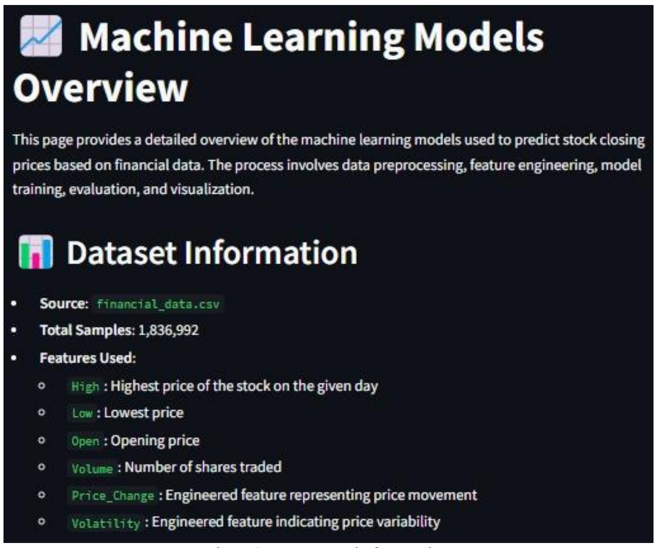
**Reasoning:**

- High recall for class 1 indicates the model is **better at predicting 'Up/Stay'** trends.
- However, it struggles to correctly identify **price drops**, indicating possible imbalance or overlap in features.

Fig. 11. A confusion matrix

Fig. 11 shows a confusion matrix of actual changes versus predicted changes, from the confusion matrix to it can be shown that the prediction model is fairly good at predicting whether the stock prices are to go up but struggles when it comes to predicting the stock prices going down

Show system performance, data processing speed, model accuracy, etc.



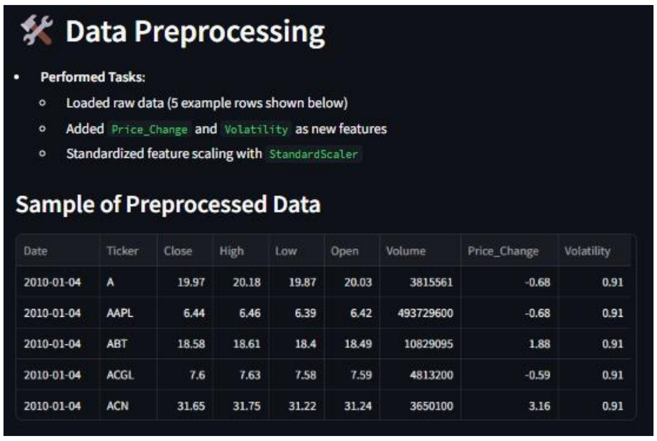
Machine Learning Models Overview

This page provides a detailed overview of the machine learning models used to predict stock closing prices based on financial data. The process involves data processing, feature engineering, model training, evaluation, and visualization.

Dataset Information

- **Source:** `financial_data.csv`
- **Total Samples:** 1,836,992
- **Features Used:**
  - `High` : Highest price of the stock on the given day
  - `Low` : Lowest price
  - `Open` : Opening price
  - `Volume` : Number of shares traded
  - `Price_Change` : Engineered feature representing price movement
  - `Volatility` : Engineered feature indicating price variability

Fig. 12a. Dataset information



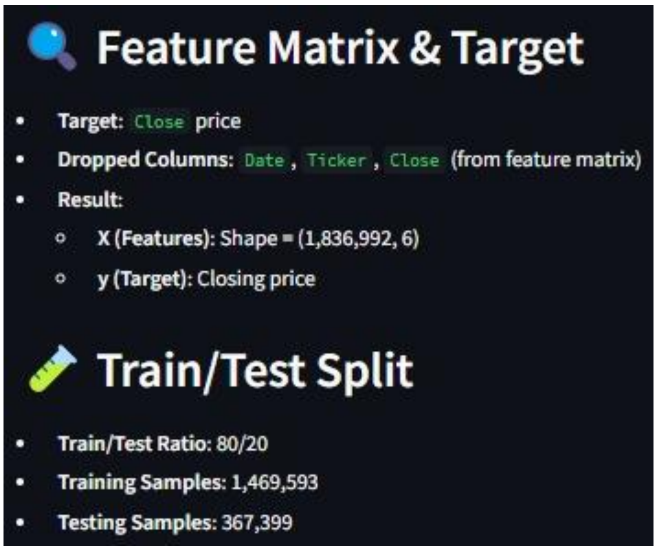
Data Preprocessing

- **Performed Tasks:**
  - Loaded raw data (5 example rows shown below)
  - Added `Price_Change` and `Volatility` as new features
  - Standardized feature scaling with `StandardScaler`

Sample of Preprocessed Data

| Date | Ticker | Close | High | Low | Open | Volume | Price_Change | Volatility |
|---|---|---|---|---|---|---|---|---|
| 2010-01-04 | A | 19.97 | 20.18 | 19.87 | 20.03 | 3815561 | -0.68 | 0.91 |
| 2010-01-04 | AAPL | 6.44 | 6.46 | 6.39 | 6.42 | 493729600 | -0.68 | 0.91 |
| 2010-01-04 | ABT | 18.58 | 18.61 | 18.4 | 18.49 | 10829095 | 1.88 | 0.91 |
| 2010-01-04 | ACGL | 7.6 | 7.63 | 7.58 | 7.59 | 4813200 | -0.59 | 0.91 |
| 2010-01-04 | ACN | 31.65 | 31.75 | 31.22 | 31.24 | 3650100 | 3.16 | 0.91 |

Fig. 12b. Data preprocessing



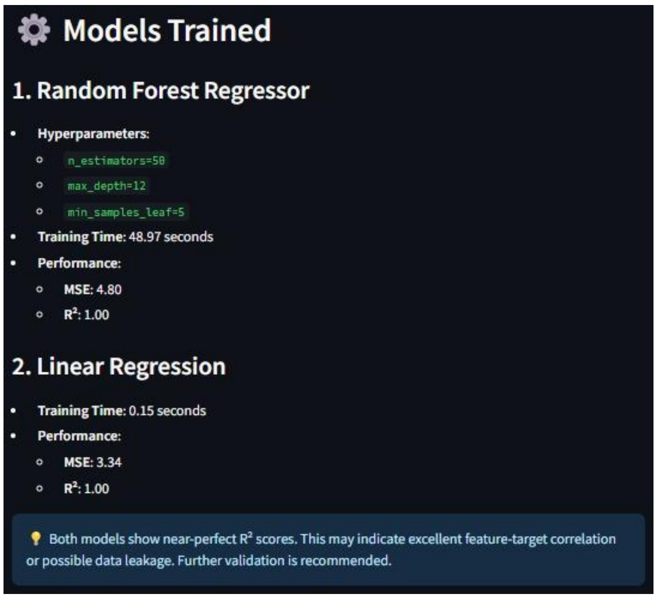
Feature Matrix & Target

- **Target:** `Close` price
- **Dropped Columns:** `Date` , `Ticker` , `Close` (from feature matrix)
- **Result:**
  - X (Features): Shape = (1,836,992, 6)
  - y (Target): Closing price

Train/Test Split

- **Train/Test Ratio:** 80/20
- **Training Samples:** 1,469,593
- **Testing Samples:** 367,399

Fig. 12c. Feature matrix and target along with Train/test split



Models Trained

1. Random Forest Regressor

- **Hyperparameters:**
  - `n_estimators=50`
  - `max_depth=12`
  - `min_samples_leaf=5`
- **Training Time:** 48.97 seconds
- **Performance:**
  - **MSE:** 4.80
  - **$R^2$:** 1.00

2. Linear Regression

- **Training Time:** 0.15 seconds
- **Performance:**
  - **MSE:** 3.34
  - **$R^2$:** 1.00

Both models show near-perfect $R^2$ scores. This may indicate excellent feature-target correlation or possible data leakage. Further validation is recommended.

Fig. 12d. Trained models

Figures 12a, 12b, 12cand 12d all show the logs of the code. Fig. 12a shows an overview of the machine learning models, information about the dataset which includes the source, total number of samples and the features which were used.

Fig. 12b shows the tasks performed while preprocessing the data, price change and volatility were added as a feature and feature scaling was standardized with the use of StandardScaler. A sample of the dataset was also shown.

Fig. 12c shows the target of the feature matrix and the dropped columns along with the result. It also shows information on the train/test split like the ratio used and how many samples were used for training and testing.

Fig. 12d shows the result of training the random forest model and the linear regression model. The random forest model had a training time of 48.97 seconds with an MSE of 4.80 and a $R^2$ of 1.00 while the linear regression model had a training time of 0.15 seconds and an MSE of 3.34 and $R^2$ of 1.00. Since both models had an $R^2$ score of 1.00 it can be said that the models may have very good feature correlation.

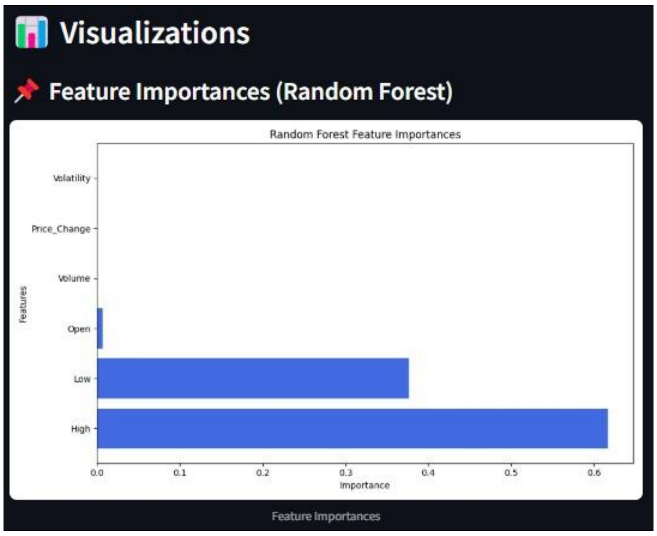
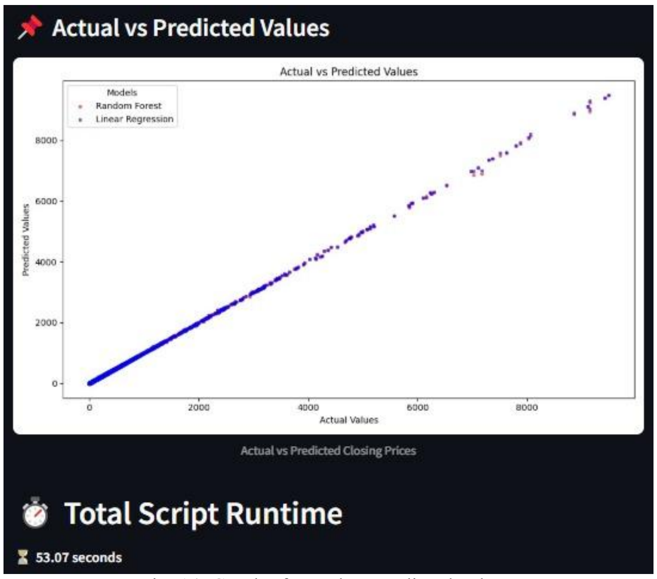


Fig. 13. Random forest feature importance

Fig. 13 shows a graph with the features of the random forest model, and it indicates how important these features are to the model. High is the most important feature according to the graph with a score of about 0.62, followed by low with a score of 0.37. This is then followed by open with a very low score of 0.01 and the other features volume, price change and volatility all have a score of 0.

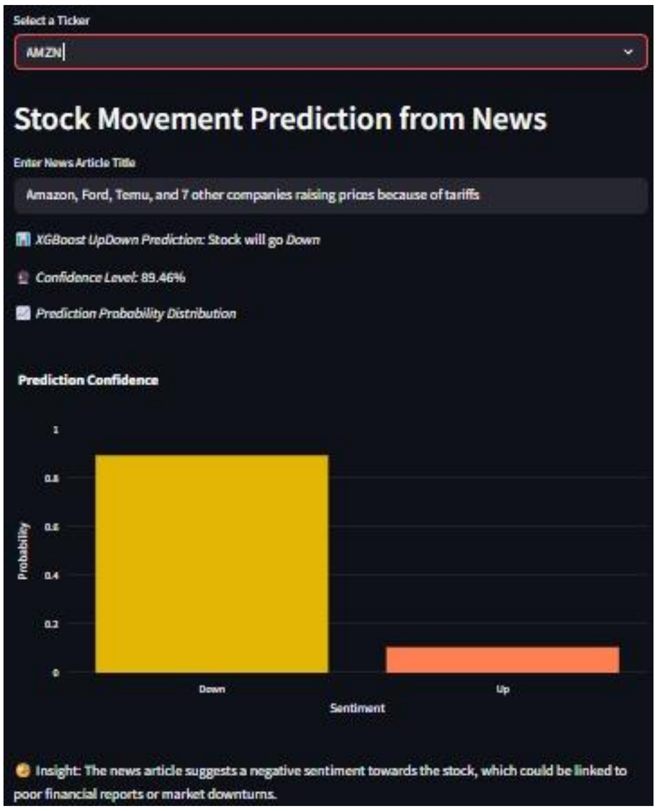

📌 Actual vs Predicted Values

⏱️ Total Script Runtime

⌛ 53.07 seconds

Fig. 14. Graph of actual vs predicted values.

In Fig. 14 it can be seen that the graph when fully plotted forms a straight line, this indicates that the values which were predicted are similar to the actual values shows the accuracy of the models.


Select a Ticker

AMZN

Stock Movement Prediction from News

Enter News Article Title

Amazon, Ford, Temu, and 7 other companies raising prices because of tariffs

*XGBoost UpDown Prediction: Stock will go Down*

*Confidence Level: 89.46%*

*Prediction Probability Distribution*

Insight: The news article suggests a negative sentiment towards the stock, which could be linked to poor financial reports or market downturns.

Fig. 15. Prediction based on news sentiment.

Fig. 15 shows the prediction of whether a stock would go up or down based on a selected news article. In this case the amazon stock is predicted to go down based on the article selected, the prediction percentage is also listed and there is a probability distribution chart below showing the probability of the stock increasing or decreasing in price.

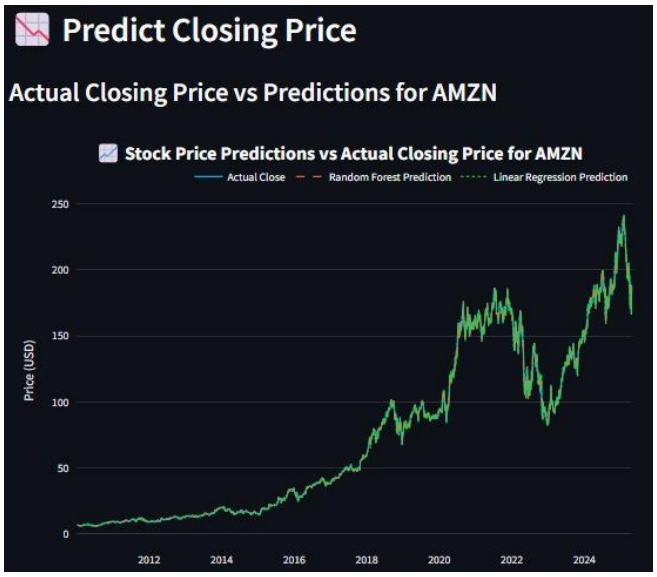


Fig. 16a. A chart showing the actual closing price vs the predicted price.

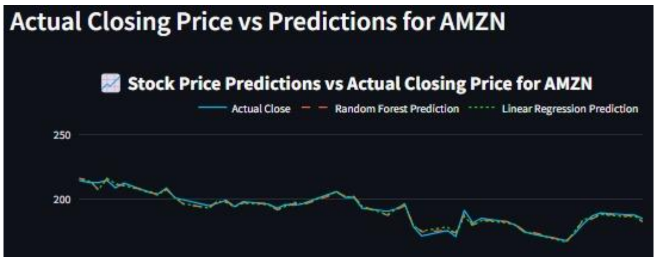


Fig. 16b. A zoomed in version of Fig. 16a showing the lines

Fig. 16a graphs the actual closing price of the amazon stock and the predicted price of the two models. In Fig. 16b we can clearly see the lines are fairly close together, implying that the models work very effectively.

## VI. Discussion

### A. *Insights Gained*

What we gained from this project was that historical price data remain effective predictors for market behaviour. However, one interesting discovery was that the incorporation of news sentiment data extracted using simple natural language processing methods on article headlines did not appreciably boost prediction accuracy. This showed the complexity of market mood and the difficulties of depending on headlines alone.

This discovery shows that sentiment data must be treated with more complexity to be beneficial. A future path is employing a Large Language Model (LLM) to undertake deeper sentiment analysis. By summarizing whole news items into concise paragraphs that convey the important content, and then extracting sentiment from such summaries, it may be feasible to collect more actionable signals from news data. This might considerably boost the prediction ability of the model.

## *B. Limitations*

A key drawback was the efficacy of the sentiment analysis application. The fundamental methodology, centred on headlines, lacked the level of detail necessary to encapsulate complex financial feeling. As a result, the news based features may have introduced noise rather than substantive insights.

The project's scope was confined to firms within the S&P 500 index. This constrained the model's capacity to generalize to other industries or international marketplaces. Enhancing the dataset to include a wider array of organizations and marketplaces might augment the model's resilience.

A further constraint was the absence of real-time data streaming. The system depended on pre-gathered, fixed datasets. Future enhancements may include the incorporation of live data streams for financial and news sentiment analysis, facilitating real-time evaluation and decision-making.

## VII. Conclusion

This project successfully produced a modular, interactive online application for stock market prediction and visualization, integrating machine learning models with technical and sentiment-based analysis. By accessing historical stock data and real-time financial news, the system allowed users to acquire both predictive and contextual insights on stock performance. The combination of data pipelines, predictive modeling, and visualization into a single platform makes the product not only effective but also user friendly, especially for students, independent researchers, and beginner investors.

Key successes include the implementation of Random Forest, Linear Regression and XGBoost models, technical indicator-based feature engineering, and an interactive interface utilizing Streamlit. The program provides visual tools for monitoring stock performance, assessing news sentiment, and examining the link between market developments and public perception.

However, the experiment also showed some limits, including the low predictive benefit from sentiment analysis and restrictions imposed by the quantity and breadth of accessible datasets. In future iterations, expanding the dataset to include more tickers and real-time data streams, refining sentiment analysis with advanced NLP models, incorporating LSTM-based deep learning architectures, and enabling automatic model retraining would significantly improve system accuracy and scalability.

Overall, this effort illustrates the real-world relevance of Big Data technology and predictive modeling in finance while creating a solid platform for additional academic investigation and practical advances in algorithmic trading and investment decision-making tools

## VIII. Reflection

### *A. Project Process*

The project followed a disciplined procedure encompassing four primary phases: planning, data collecting and preprocessing, model building, and system integration. During the planning stage, the team set goals, identified relevant tools and technologies, and allocated work based on individual talents and interests. Regular meetings were conducted to monitor progress, debate design choices, and overcome barriers jointly.

Execution involves iterative development, commencing with exploratory data analysis and feature engineering, followed by establishing baseline models. Once the models were verified, the group went on to constructing the front-end application using Streamlit and connecting the data pipelines. Each member contributed to diverse areas of the system including model training, frontend design, documentation, and back testing ensuring a balanced effort and successful cooperation.

The initiative fostered cooperation by fostering clear communication, version control via GitHub, and mutual assistance when obstacles occurred. Overall, the process emphasized the significance of cooperation, flexibility, and constant learning in a real-world data science project.

### *B. Challenges and Solutions*

Several technical and operational obstacles developed throughout the project. One of the primary concerns identified was the financial news API periodically stalling out during data retrieval. This disturbed the sentiment analysis process and hindered down progress. The approach involves establishing a delay between queries to prevent overloading the API server, therefore stabilizing the scraping process.

Another important issue was the instability of the Yahoo Finance API, which ceased operating halfway through the project. This hampered the real-time data intake process and model upgrades. To circumvent this, all relevant stock data was collected and kept locally, guaranteeing that the program could continue to work independently of third-party data sources.

These experiences underlined the significance of fault-tolerance and backup procedures when interacting with external APIs. By executing these workarounds, the project was able to preserve continuity and deliver on its original goals.

## References

[1] A. Subasi, F. Amir, K. Bagedo, A. Shams and A. Sarirete, “Stock market prediction using machine learning,” *Procedia Computer Science*, vol. 199, pp. 956–963, 2022.

[2] J. Maqbool, P. Aggarwal, R. Kaur, A. Mittal and I. Ganaie, “Stock Prediction by Integrating Sentiment Scores of Financial News and MLP-Regressor: A Machine Learning Approach,” *Procedia Computer Science*, vol. 215, pp. 761–770, 2023.

[3] K. Du, F. Xing, R Mao and E. Cambria, “Financial sentiment analysis: Techniques and applications,” *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–35, Jan. 2024.

[4] T. Phuoc, P. T. K. Anh, P. H. Tam and C. V. Nguyen, “Applying machine learning algorithms to predict the stock price trend,” *Humanities and Social Sciences Communications*, vol. 11, no. 1, pp. 1–10, Jan. 2024.

[5] S. Vadlamudi, “Stock market prediction using machine learning: A systematic literature review,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 9, pp. 456–463, 2021.

[6] T. Jiang and Q. Zeng, “Financial sentiment analysis using FinBERT with application in predicting stock movement,” *arXiv preprint arXiv:2306.02136*, Jun. 2023.

## Appendicies

Link to YouTube video demonstrating the use of the web application: https://youtu.be/60bXAOetOQs

Github repository: https://github.com/jaheemedwards/COMP3610_Project