# Group 4 Final Project Report
# Stock Market Sentiment Analysis

Course IS688102 – Web Mining
Project Title: *Predicting Stock Movements Using News Sentiment and Machine Learning.*
Team Members:  Sai Leela Kuragayala, Akhil Karri, Karthikeya Rao Bollamala, Ramita Deshmukh.

## 1. Introduction:

The stock market is a dynamic and complex system influenced by a wide range of factors, from economic indicators and corporate performance to investor sentiment and public perception. In recent years, the availability of financial news and social media commentary in digital form has made it possible to extract public sentiment using Natural Language Processing (NLP) techniques. This project explores the impact of news sentiment on the short-term price movement of two widely followed stocks: **Apple (AAPL) and Tesla (TSLA).**

The core idea behind this research is that investor emotions—fear, excitement, uncertainty—are often reflected in financial news headlines and stories. These emotional cues, when aggregated and analyzed using machine learning models, can provide predictive insights into how stock prices may behave soon. While traditional stock prediction models rely heavily on historical pricing data and technical indicators, this project takes a hybrid approach by incorporating both structured financial data and unstructured news text data.

We used FinBERT, a domain-specific variant of the BERT transformer model fine-tuned on financial corpora, to classify each news headline into positive, negative, or neutral sentiment. We then combined this sentiment with daily trading data (open price, close price, volume, and moving averages) to build a feature-rich dataset. Multiple machine learning algorithms including Random Forest, XGBoost, Logistic Regression, SVM, and KNN were tested for their predictive power. Finally, a Streamlit dashboard was developed to allow real-time stock sentiment analysis using live news data.

This project is a practical demonstration of applying Web Mining, NLP, and Machine Learning to the financial domain. It not only shows that sentiment analysis can be quantitatively useful but also builds an interactive tool that can serve as the foundation for larger trading or decision-support systems in the future.

*GitHub Repository: https://github.com/Kleela3498/stockprice*

## 2. Project Motivation:

Financial markets are highly sensitive to public perception and global events, and this sensitivity is often captured in the tone and content of financial news. For retail and institutional investors alike, understanding sentiment trends can offer a competitive edge. While technical indicators like moving averages and historical price trends remain foundational, they fail to capture the why behind sudden market movements—especially those triggered by news, policy changes, or unexpected announcements.

This project was motivated by the desire to bridge the gap between numerical stock indicators and qualitative news sentiment. By integrating sentiment scores from real-world financial headlines with pricing data, we aim to improve the quality and realism of short-term stock movement predictions. This is particularly useful for developing intelligent trading tools or alert systems that can respond to market news in real time.

Furthermore, with APIs like NewsAPI and Alpha Vantage, it's now feasible to build fully automated real-time systems that process, analyze, and react to streaming data. This project captures that potential by not only training predictive models, but also deploying them in an interactive, real-world application.

## 3. Dataset Collection:

This project uses two major types of datasets—financial news data and stock market trading data—focusing specifically on Apple Inc. (AAPL) and Tesla Inc. (TSLA). These companies were chosen because they are two of the most publicly discussed and traded stocks, often responding sharply to market sentiment and headlines.

### 3.1 News Data Collection:

The financial news datasets were sourced from Kaggle and curated from reputable publications such as The Wall Street Journal, The New York Times, and Yahoo Finance. These datasets contain historical financial news headlines and article bodies, tagged with publication dates and company references.

Each record included the following fields:
**date:** Publication timestamp
**title:** News headline (used for sentiment classification)
**content:** Full text of the article
**link:** Source URL
**symbols:** Mentioned stock tickers (Apple dataset)
**tags:** Article tags or keywords
**sentiment_polarity, sentiment_pos, sentiment_neg, sentiment_neu:** Precomputed sentiment scores (Apple dataset)
**source:** News publisher (Tesla dataset only)
**company:** A custom label for Apple or Tesla (manually added in preprocessing)

We filtered both datasets to the year 2022 to align them with stock price data. After filtering:
Apple News: ~**29,752 articles** → 8,953 for 2022
Tesla News: 763 → **632 valid articles** after cleaning

*link to tesla data → https://www.kaggle.com/datasets/saleepshrestha/newspapers*
*link to apple data → https://www.kaggle.com/datasets/frankossai/apple-stock-aapl-historical-financial-news-data?resource=download*

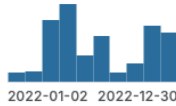*Figure 1: Raw Dataset Sample*

## 3.2 Stock Data Collection:

To complement the sentiment analysis, we collected daily stock market data from Yahoo Finance using the yfinance Python library. The datasets include:

**date:** Trading Day
**open:** Opening price
**close:** Closing price.
**volume:** Number of shares traded.
**company:** Added manually to identify Apple or Tesla

Data was extracted only for 2022, matching the news datasets for both companies. We retained only essential features for modeling and renamed columns for consistency (Open → open_price, Close → close_price).



Tesla Stock Data:

|   | Date | Open | High | Low | Close | Adj Close | Volume | Company |
|---|------|------|------|-----|-------|-----------|--------|---------|
| 0 | 2022-12-30 | 119.95 | 124.48 | 119.75 | 123.18 | 123.18 | 157,777,300 | Tesla |
| 1 | 2022-12-29 | 120.39 | 123.57 | 117.5 | 121.82 | 121.82 | 221,923,300 | Tesla |
| 2 | 2022-12-28 | 110.35 | 116.27 | 108.24 | 112.71 | 112.71 | 221,070,500 | Tesla |
| 3 | 2022-12-27 | 117.5 | 119.67 | 108.76 | 109.1 | 109.1 | 208,643,400 | Tesla |
| 4 | 2022-12-23 | 126.37 | 128.62 | 121.02 | 123.15 | 123.15 | 166,989,700 | Tesla |

Apple Stock Data:

|   | Date | Open | High | Low | Close | Adj Close | Volume | Company |
|---|------|------|------|-----|-------|-----------|--------|---------|
| 0 | 2022-12-30 | 119.95 | 124.48 | 119.75 | 123.18 | 123.18 | 157,777,300 | Apple |
| 1 | 2022-12-29 | 120.39 | 123.57 | 117.5 | 121.82 | 121.82 | 221,923,300 | Apple |
| 2 | 2022-12-28 | 110.35 | 116.27 | 108.24 | 112.71 | 112.71 | 221,070,500 | Apple |
| 3 | 2022-12-27 | 117.5 | 119.67 | 108.76 | 109.10 | 109.10 | 208,643,400 | Apple |
| 4 | 2022-12-23 | 126.37 | 128.62 | 121.02 | 123.15 | 123.15 | 166,989,700 | Apple |

*Figure 2 Stock price data*

# 4. Dataset Preprocessing:

After collecting the raw datasets for news articles and stock prices, several preprocessing steps were essential to ensure consistency, remove noise, and prepare the data for sentiment extraction and modeling. This section outlines the two main phases of preprocessing: **data cleaning and news dataset merging.**

## 4.1 News Data Cleaning:

Raw data often contains inconsistencies such as missing values, formatting issues, or unnecessary metadata. Cleaning was performed independently for the Apple news, Tesla news, and stock price datasets using pandas.

**Cleaning the News Datasets:**
1. *Date Format Standardization:*
    1.1. Apple news entries had **time zone-aware ISO 8601 strings** (T00:00:00+00:00)
    1.2. Tesla news dates were formatted inconsistently across sources.
    1.3. All dates were converted to uniform time zone-naive datetime objects using:
2. *Null & Invalid Rows:*
    2.1. **132 entries in Tesla news had invalid** or null date values.
    2.2. These rows were dropped using dropna(subset=['date'])
3. *Volume Parsing (Stock Data):*
    3.1. Stock volume values were stored as strings with commas (e.g., "221,923,300")
    3.2. We used **.str.replace(',', '')** and converted them to float
4. *Irrelevant Column Removal:*
    4.1. Fields such as **tags, symbols, and link were dropped** as they were not required for modeling.
    4.2. Columns were renamed consistently (Open → open_price, Close → close_price)
5. *Data Type Enforcement:*
    5.1. Numeric fields (open_price, close_price, volume) were cast to float64 for modeling compatibility.

Tesla Dataset Sample:

| | Unnamed: 0 | date | source | title | link | content |
|---|---|---|---|---|---|---|
| 0 | 0 | 28 Apr 2022 | New York Times, Late Edition (East Coast) | How Tesla Stock Price Might Affect Markets: [B... | https://www.proquest.com/usnews/docview/265597... | Tesla is highly traded, and Elon Musk's sale o... |
| 1 | 1 | 18 May 2022 | Wall Street Journal (Online) | Elon Musk Calls ESG 'An Outrageous Scam' After... | https://www.proquest.com/usnews/docview/266576... | \nTesla Inc. was recently dropped from an equi... |
| 2 | 2 | 14 Nov 2022 | Wall Street Journal (Online) | Elon Musk's Control Over Tesla's Board in Focu... | https://www.proquest.com/usnews/docview/273573... | \nWILMINGTON, Del. —Opening-day testimony in th... |
| 3 | 3 | 15 Nov 2022 | Wall Street Journal (Online) | Elon Musk's Influence Over Tesla's Board in Fo... | https://www.proquest.com/usnews/docview/273604... | \nWILMINGTON, Del. —Opening-day testimony in th... |
| 4 | 4 | 21 May 2022 | New York Times, Late Edition (East Coast) | Tesla's Star Dims as Stock Plunges, Highlighti... | https://www.proquest.com/usnews/docview/266713... | Production problems in China and Elon Musk's p... |

*Figure 3 Tesla Dataset Before Cleaning*

Tesla Reduced Sample:

| | date | title | content | company |
|---|---|---|---|---|
| 0 | 28 Apr 2022 | How Tesla Stock Price Might Affect Markets: [B... | Tesla is highly traded, and Elon Musk's sale o... | Tesla |
| 1 | 18 May 2022 | Elon Musk Calls ESG 'An Outrageous Scam' After... | \nTesla Inc. was recently dropped from an equi... | Tesla |
| 2 | 14 Nov 2022 | Elon Musk's Control Over Tesla's Board in Focu... | \nWILMINGTON, Del.—Opening-day testimony in th... | Tesla |
| 3 | 15 Nov 2022 | Elon Musk's Influence Over Tesla's Board in Fo... | \nWILMINGTON, Del.—Opening-day testimony in th... | Tesla |
| 4 | 21 May 2022 | Tesla's Star Dims as Stock Plunges, Highlighti... | Production problems in China and Elon Musk's p... | Tesla |

Apple Reduced Sample:

| | date | title | content | company |
|---|---|---|---|---|
| 0 | 2024-11-27T16:39:00+00:00 | Berkshire Stock Hits Record Even as Company Re... | Warren Buffett's caution, his advancing age, a... | Apple |
| 1 | 2024-11-26T00:00:00+00:00 | What Is a Stock Market Index? | What Is a Stock Market Index? | Apple |
| 2 | 2024-11-26T00:00:00+00:00 | Could Investing $1,000 in Apple Make You a Mil... | Could Investing $1,000 in Apple Make You a Mil... | Apple |
| 3 | 2024-11-26T00:00:00+00:00 | Dow Jones Industrial Average | Dow Jones Industrial Average | Apple |
| 4 | 2024-11-26T00:00:00+00:00 | What Is the S&P 500 Index? | What Is the S&P 500 Index? | Apple |

*Figure 4 Apple & Tesla Dataset After Cleaning*

## 4.2 News Dataset Merging:

After cleaning both Apple and Tesla news datasets, we aligned them by time to prepare a unified sentiment analysis base.

1. Date Range Filtering:
    1.1. The Tesla dataset covered only the year 2022.
    1.2. Apple news ranged from 2016 to 2024.
    1.3. To ensure temporal alignment, Apple news was filtered:
    *apple_df = apple_df[(apple_df['date'] >= '2022-01-03') & (apple_df['date'] <= '2022-12-31')]*
2. Company Column Addition:
    2.1. Both Apple and Tesla datasets were assigned a new column company with values Apple and Tesla respectively. This enabled merging and grouping operations later.
3. Concatenation:
    3.1. The two datasets were combined using:
    *combined_df = pd.concat([apple_df_filtered, tesla_df_filtered], ignore_index=True)*
    3.2. Records were sorted chronologically by date to form a unified timeline of news events.

Final Combined Dataset Sample:

| | date | title | content | company |
|---|---|---|---|---|
| 0 | 2022-01-03 00:00:00 | Tesla Crushed Every Expectation in 2021, Kicki... | A Tesla Model X on display at the 3rd China In... | Tesla |
| 1 | 2022-01-03 10:10:56 | 5 predictions for the stock market in 2022: Mo... | This article first appeared in the Morning Bri... | Apple |
| 2 | 2022-01-03 11:00:58 | 7 of the Best Robinhood Stocks for 2022 to Buy... | Robinhood Markets (NASDAQ:HOOD) is a pioneer i... | Apple |
| 3 | 2022-01-03 11:03:00 | Have $2,000? These 2 Stocks Could Be Bargain B... | The good news for investors is that the stock ... | Apple |
| 4 | 2022-01-03 11:45:00 | 3 Surefire Metaverse Stocks That Could Make Yo... | The metaverse has created quite a buzz as comp... | Apple |

*Figure 5 Final Merged News Dataset*

This merged news dataset, now clean and aligned, was passed forward into the sentiment analysis pipeline, where FinBERT was applied to generate sentiment labels and scores.

## 4.3 Stock Data Cleaning:

Stock price data was downloaded in raw form with several unnecessary fields, including High, Low, and Adj Close. Before merging it with sentiment data, we applied a series of cleaning steps to retain only essential attributes and ensure consistent formatting.

**Steps Performed:**
- Column Reduction:
    - Only the following columns were retained:
        - Date
        - Open
        - Close
        - Volume
    - Company (manually added)
- Column Renaming:
    - For consistency with the news dataset and to prepare for merging, the stock data columns were renamed:
        - Date → date
        - Open → open_price
        - Close → close_price
        - Volume → volume
        - Company → company
- Consistent Field Formatting:

o All columns were cast to appropriate data types. Dates were converted using pd.to_datetime(), and the volume field—originally stored as a string with commas—was cleaned and cast to float after removing commas.

### Cleaned Tesla Stock Data:

| | date | open_price | close_price | company | volume |
|---|---|---|---|---|---|
| 0 | 2022-12-30 | 119.95 | 123.18 | Tesla | 157,777,300 |
| 1 | 2022-12-29 | 120.39 | 121.82 | Tesla | 221,923,300 |
| 2 | 2022-12-28 | 110.35 | 112.71 | Tesla | 221,070,500 |
| 3 | 2022-12-27 | 117.5 | 109.1 | Tesla | 208,643,400 |
| 4 | 2022-12-23 | 126.37 | 123.15 | Tesla | 166,989,700 |

### Cleaned Apple Stock Data:

| | date | open_price | close_price | company | volume |
|---|---|---|---|---|---|
| 0 | 2022-12-30 | 119.95 | 123.18 | Apple | 157,777,300 |
| 1 | 2022-12-29 | 120.39 | 121.82 | Apple | 221,923,300 |
| 2 | 2022-12-28 | 110.35 | 112.71 | Apple | 221,070,500 |
| 3 | 2022-12-27 | 117.5 | 109.10 | Apple | 208,643,400 |
| 4 | 2022-12-23 | 126.37 | 123.15 | Apple | 166,989,700 |

*Figure 6 Cleaned Stock Prices*

## 5. Sentiment Analysis and Data Merging:

After preprocessing the news and stock price datasets, the next critical phase involved generating sentiment scores for each article using a financial-domain-specific language model and merging those results with daily stock metrics. This section outlines the process of sentiment classification using **FinBERT**, followed by **merging sentiment data with stock price data** to build the final dataset used for machine learning.

### 5.1 Sentiment Analysis Using FinBERT:

To analyze the sentiment of financial news headlines, we used FinBERT, a domain-specific transformer model pre-trained on financial text corpora. Unlike generic sentiment models trained on movie or product reviews, FinBERT is fine-tuned to capture the subtle tone and context of financial language, making it particularly effective for interpreting investor sentiment from market news.

**FinBERT Setup Overview:**
- Model: yiyanghkust/finbert-tone from Hugging Face
- Pipeline: transformers.pipeline("sentiment-analysis", model=..., tokenizer=...)
- Input: News article titles (title column)
- Output per article:
  - sentiment_label: One of Positive, Neutral, or Negative
  - sentiment_score: Model's confidence score
- Each headline was passed through this pipeline to generate sentiment predictions, which were then stored in new columns. These values formed the basis for calculating daily average sentiment and integrating textual signals with numerical trading data.

✅ News Dataset After FinBERT Sentiment Analysis:

| | date | company | title | sentiment_label | sentiment_score |
|---|---|---|---|---|---|
| 0 | 2022-01-03 00:00:00 | Tesla | Tesla Crushed Every Expectation in 2021, Kicki... | neutral | 0.999439 |
| 1 | 2022-01-03 10:10:56 | Apple | 5 predictions for the stock market in 2022: Mo... | neutral | 0.999901 |
| 2 | 2022-01-03 11:00:58 | Apple | 7 of the Best Robinhood Stocks for 2022 to Buy... | positive | 0.993369 |
| 3 | 2022-01-03 11:03:00 | Apple | Have $2,000? These 2 Stocks Could Be Bargain B... | neutral | 0.922492 |
| 4 | 2022-01-03 11:45:00 | Apple | 3 Surefire Metaverse Stocks That Could Make Yo... | neutral | 0.991376 |

*Figure 7 News Data After Sentiment Analysis*

## 5.2 Daily Sentiment Aggregation:

Many days featured multiple news articles per company. We computed the mean sentiment score per day per company:
This step ensured each company had a single, aggregate sentiment score per trading day.

## 5.3 Merging with Stock Data and Feature Engineering:

We now merged the cleaned stock price data with the daily sentiment scores using date and company as keys. This created a comprehensive dataset that aligned public sentiment with daily trading data.

To enrich the dataset further, we engineered lag and trend features:
- **prev_sentiment:** Sentiment score from the previous day
- **prev_close_price:** Previous day's close
- **ma_5, ma_10**: Rolling averages of close price over 5 and 10 days.

We also created the classification label:
- **movement_label:** Based on percentage change in next day's close:
  - 1: Up (significant increase)
  - 0: Stable (within threshold)

| | date | open_price | close_price | company | volume | sentiment_score | prev_sentiment | prev_close_price | ma_5 | ma_10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2022-01-04 | 396.52 | 383.20 | Apple | 100248300.0 | 0.961389 | 0.943321 | 399.93 | 391.565000 | 391.565000 |
| 1 | 2022-01-05 | 382.22 | 362.71 | Apple | 80119800.0 | 0.963846 | 0.961389 | 383.20 | 381.946667 | 381.946667 |
| 2 | 2022-01-06 | 359.00 | 354.90 | Apple | 90336600.0 | 0.945914 | 0.963846 | 362.71 | 375.185000 | 375.185000 |
| 3 | 2022-01-07 | 360.12 | 342.32 | Apple | 84164700.0 | 0.986895 | 0.945914 | 354.90 | 368.612000 | 368.612000 |
| 4 | 2022-01-10 | 333.33 | 352.71 | Apple | 91815000.0 | 0.930174 | 0.986895 | 342.32 | 359.168000 | 365.961667 |

*Figure 8 Final Dataset After Merging*

# 6. Model Training and Evaluation:

## 6.1 Label Encoding:

Following the construction of the final merged dataset—which combined daily sentiment, technical indicators, and historical stock prices—we proceeded to train and evaluate multiple machine learning models to predict short-term stock movement. This section outlines the label encoding, feature scaling, model selection, training methodology, and evaluation metrics.

The target column movement_label originally consisted of three classes:
- -1: Price went down
- 0: Price remained stable
- 1: Price went up

To make this compatible with most scikit-learn models, we mapped these labels to non-negative integers:

As a result:
- 0 → Down
- 1 → Stable
- 2 → Up

## 6.2 Feature Selection and Scaling:

We selected seven key features for training:
- open_price
- sentiment_score
- prev_sentiment
- prev_close_price
- ma_5
- ma_10
- volume

To ensure equal contribution across all features especially important for models like SVM and KNN we standardized the features using **StandardScaler()**

## 6.3 Chronological Train- Test Split:

To maintain temporal integrity and simulate real-world forecasting, we used a chronological 80/20 split:

- Training samples: 274
- Testing samples: 69

```python
# Mapping movement labels to non-negative integers
label_mapping = {
    -1: 0,   # Down → 0
    0: 1,    # Stable → 1
    1: 2     # Up → 2
}

# Apply mapping
y_mapped = merged_df['movement_label'].map(label_mapping)

# Feature Selection
features = ['open_price', 'sentiment_score', 'prev_sentiment', 'prev_close_price', 'ma_5', 'ma_10' , 'volume']
scaler = StandardScaler()

X_scaled = scaler.fit_transform(merged_df[features])

X =  pd.DataFrame(X_scaled, columns=features)
y = y_mapped

# Chronological Train-Test Split (80%-20%)
split_index = int(len(X) * 0.8)
X_train, X_test = X.iloc[:split_index], X.iloc[split_index:]
y_train, y_test = y.iloc[:split_index], y.iloc[split_index:]

print("Training samples:", len(X_train))
print("Testing samples:", len(X_test))
```

```
Training samples: 274
Testing samples: 69
```

*Figure 9 Code for Mapping Labels, Feature Selection & Train-Test Split*

## 6.4 Model Selection:

We experimented with five widely used classification algorithms:

- **Logistic Regression:** Linear classifier, interpretable and fast.
- **Random Forest Ensemble of decision trees:** robust to noise.
- **XGBoost Classifier:** Gradient-boosted trees optimized for performance.
- **Support Vector Machine (SVM):** Kernel-based margin classifier.
- **K-Nearest Neighbors (KNN):** Distance-based voting classifier.

## 6.5 Evaluation Metrics:

For each model, we computed:

- Accuracy
- Precision, Recall, F1-score (using classification_report)
- Support per class (Down, Stable, Up)

```
=== Logistic Regression ===
              precision    recall  f1-score   support

     Down(0)       0.65      0.41      0.50        27
    Stable(1)      0.36      0.32      0.34        20
       Up(2)       0.43      0.67      0.52        22

    accuracy                           0.47        69
   macro avg       0.48      0.47      0.45        69
weighted avg       0.50      0.47      0.47        69

Accuracy: 0.4678

=== Random Forest ===
              precision    recall  f1-score   support

     Down(0)       0.65      0.74      0.69        27
    Stable(1)      0.75      0.65      0.69        20
       Up(2)       0.72      0.65      0.68        22

    accuracy                           0.70        69
   macro avg       0.71      0.68      0.69        69
weighted avg       0.70      0.70      0.70        69

Accuracy: 0.6997

=== XGBoost Classifier ===
              precision    recall  f1-score   support

     Down(0)       0.62      0.78      0.69        27
    Stable(1)      0.63      0.70      0.66        20
       Up(2)       0.60      0.41      0.49        22

    accuracy                           0.63        69
   macro avg       0.62      0.63      0.61        69
weighted avg       0.62      0.63      0.62        69

Accuracy: 0.6272

=== Support Vector Machine ===
              precision    recall  f1-score   support

     Down(0)       0.56      0.70      0.62        27
    Stable(1)      0.82      0.35      0.49        20
       Up(2)       0.55      0.68      0.61        22

    accuracy                           0.57        69
   macro avg       0.64      0.58      0.57        69
weighted avg       0.61      0.57      0.56        69

Accuracy: 0.5693

=== K-Nearest Neighbors ===
              precision    recall  f1-score   support

     Down(0)       0.63      0.78      0.70        27
    Stable(1)      0.64      0.57      0.60        20
       Up(2)       0.63      0.52      0.57        22

    accuracy                           0.64        69
   macro avg       0.63      0.62      0.62        69
weighted avg       0.63      0.64      0.63        69

Accuracy: 0.6417
```

*Figure 10 Classification Table*

## 6.6 Model Accuracy Comparison:

*Table 1 Model Accuracy Comparison*

| Model | Accuracy |
|---|---|
| Logistic Regression | 46.78% |
| Random Forest | 69.97% |
| XGBoost Classifier | 62.72% |
| Support Vector Machine | 56.93% |
| K – Nearest Neighbors | 64.17% |

## 6.7 Final Model Selection:

The Random Forest classifier delivered the best balance of accuracy and class-wise performance. It handled non-linear relationships well and was relatively robust to overfitting.
We serialized this model using Python's pickle library for deployment in the Streamlit app:

```python
import pickle


# Save the trained model to a pickle file
with open('best_rf.pkl', 'wb') as f:
    pickle.dump(best_rf, f)
```

*Figure 11 Pickle File Creation*

# 7. Feature Importance Analysis:

After training our machine learning models, we analyzed feature importances to understand which variables had the greatest influence on predicting stock movement. This step is crucial for evaluating the interpretability and trustworthiness of the models, especially in financial decision-making.

## 7.1  Methodology:

Each model provides feature importance differently:
- Tree-based models (Random Forest, XGBoost): use built-in .feature_importances_
- Linear models (Logistic Regression): use absolute value of model coefficients.
- SVM: Feature importance not directly supported for non-linear kernels.
- KNN: No inherent feature importance but can be approximated via permutation or other post hoc techniques.

The features evaluated were:
- open_price
- sentiment_score
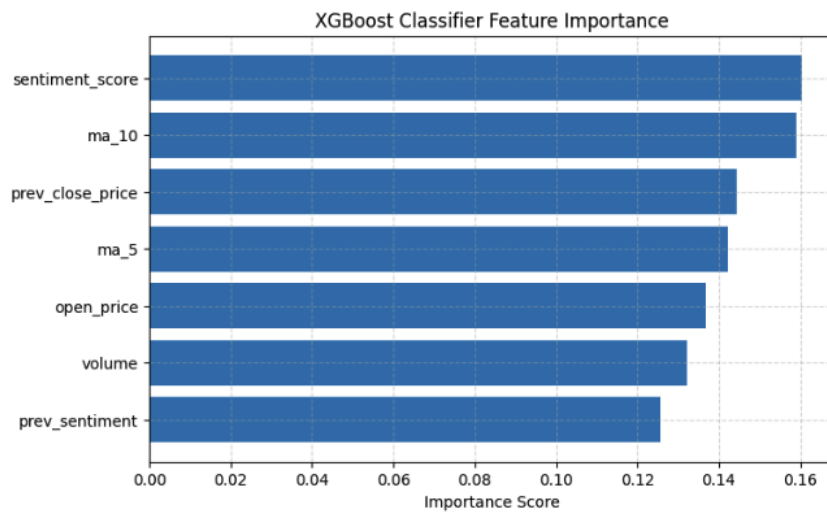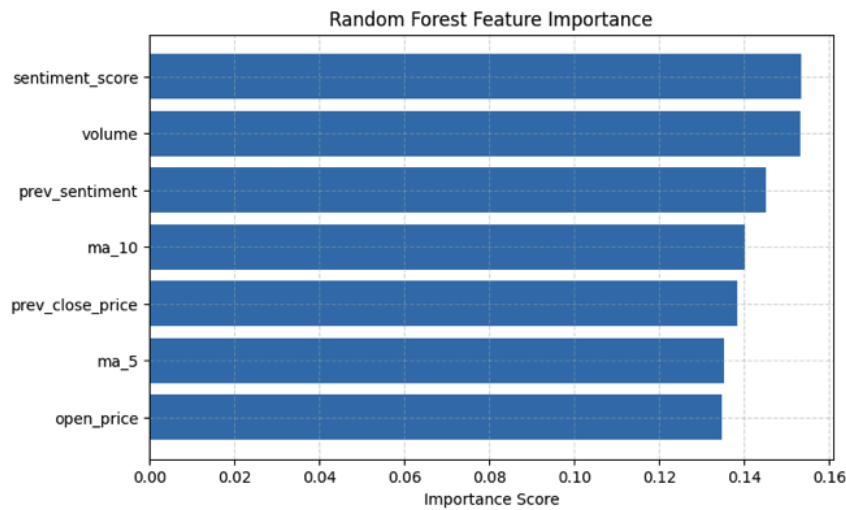- prev_sentiment
- prev_close_price
- ma_5
- ma_10
- volume

*Figure 12 Random Forest & XGBoost Classifier feature importance*

## 7.2 Random Forest Insights:

Top Features:
- sentiment_score
- volume
- prev_sentiment
- ma_10

Interpretation:

Sentiment-based features (sentiment_score, prev_sentiment) were among the highest weighted predictors, confirming the hypothesis that market sentiment strongly correlates with short-term price movement. Traditional indicators like moving averages also contributed meaningfully.

## 7.3 XGBoost Classifier Insights:

- Top Features:
- sentiment_score
- ma_10
- prev_close_price

Interpretation:

XGBoost favored a mix of technical and sentiment features. It particularly weighted longer-term moving averages (ma_10) alongside sentiment scores, suggesting that momentum combined with sentiment may offer better predictive power.

No feature importance for SVM



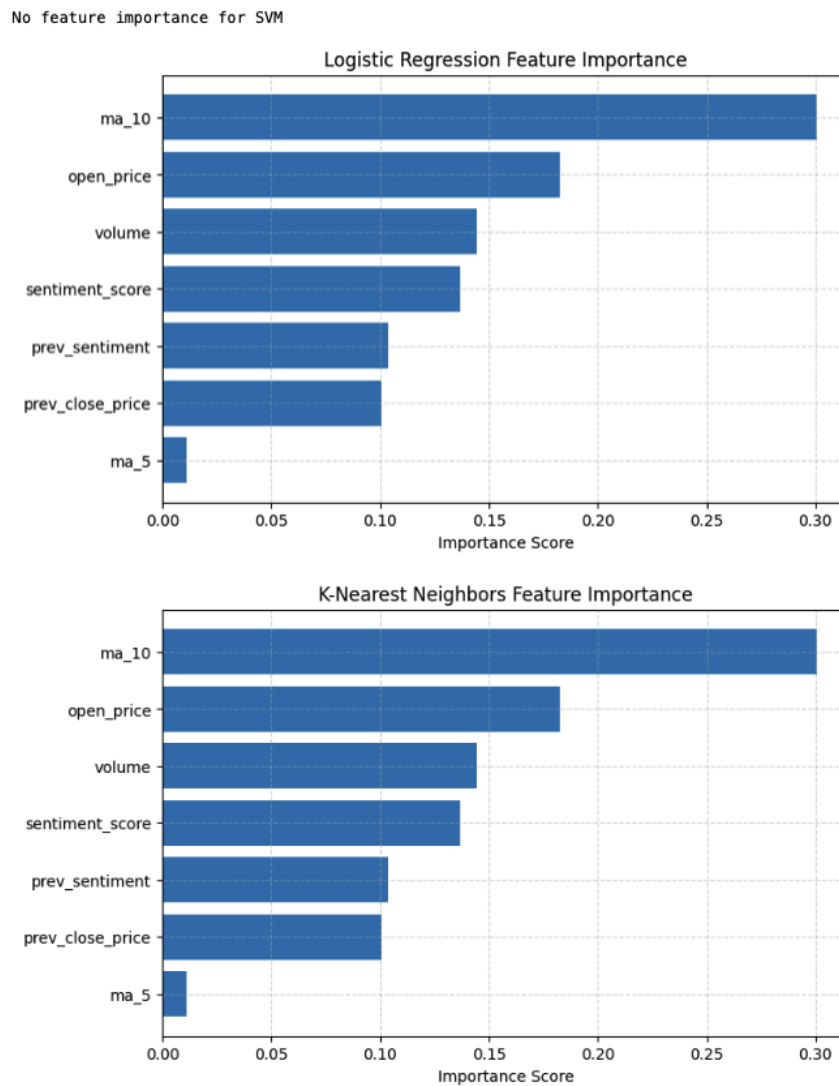Figure 13 Logistic Regression & K-Nearest Neighbors feature importance

## 7.4 Logistic Regression Insights:

Top Features (by coefficient magnitude):
- ma_10
- open_price
- volume

Interpretation:

Being a linear model, Logistic Regression emphasized features with broader scale variability, like price and volume. While sentiment played a secondary role, ma_10 consistently emerged as a top feature across models.

## 7.5 K-Nearest Neighbors Insights:

Top Features:
- ma_10
- open_price
- volume
- sentiment_score

Interpretation:
KNN performed similarly to tree-based models but showed greater sensitivity to distance-based features like ma_10 and open_price. Sentiment still held importance but was weighted slightly lower.

## 7.6 Support Vector Machine:

SVM with RBF kernel does not expose meaningful feature importance via coefficients. As such, it was excluded from the feature importance visualizations.

# 8. Live Prediction & Streamlit Dashboard:

To bring the machine learning model into a real-world usable form, we developed a Streamlit dashboard capable of performing live stock sentiment analysis and price movement prediction. This dashboard integrates several components:

- **Alpha Vantage API**: For real-time intraday stock prices (Open, Close, Volume).
- **Yahoo Finance API (yfinance):** For historical stock data to calculate 5-day and 10-day moving averages.
- **NewsAPI:** For fetching the latest 5 news headlines about the entered stock.
- **FinBERT Model:** To perform live sentiment analysis on news headlines.
- **Random Forest Model:** Our trained model predicts the movement (Up, Down, Stable).

## 8.1 User Workflow:

The user enters a stock ticker (e.g., TSLA, AAPL) into the dashboard.
On clicking Predict Now:

- Live stock data and recent news headlines are fetched.
- Sentiment scores are generated using FinBERT for the news.
- Technical indicators (moving averages, volume) are calculated.
- Features are engineered live and fed into the Random Forest model.
- The system predicts whether the stock is likely to Go Up, Stay Stable, or Go Down.

## 8.2 Feature Displayed:

- Predicted Movement (Up, Stable, Down)
- Sentiment Score (Aggregated from latest headlines)
- Open Price
- Previous Close Price
- 5-day Moving Average
- 10-day Moving Average

- Volume
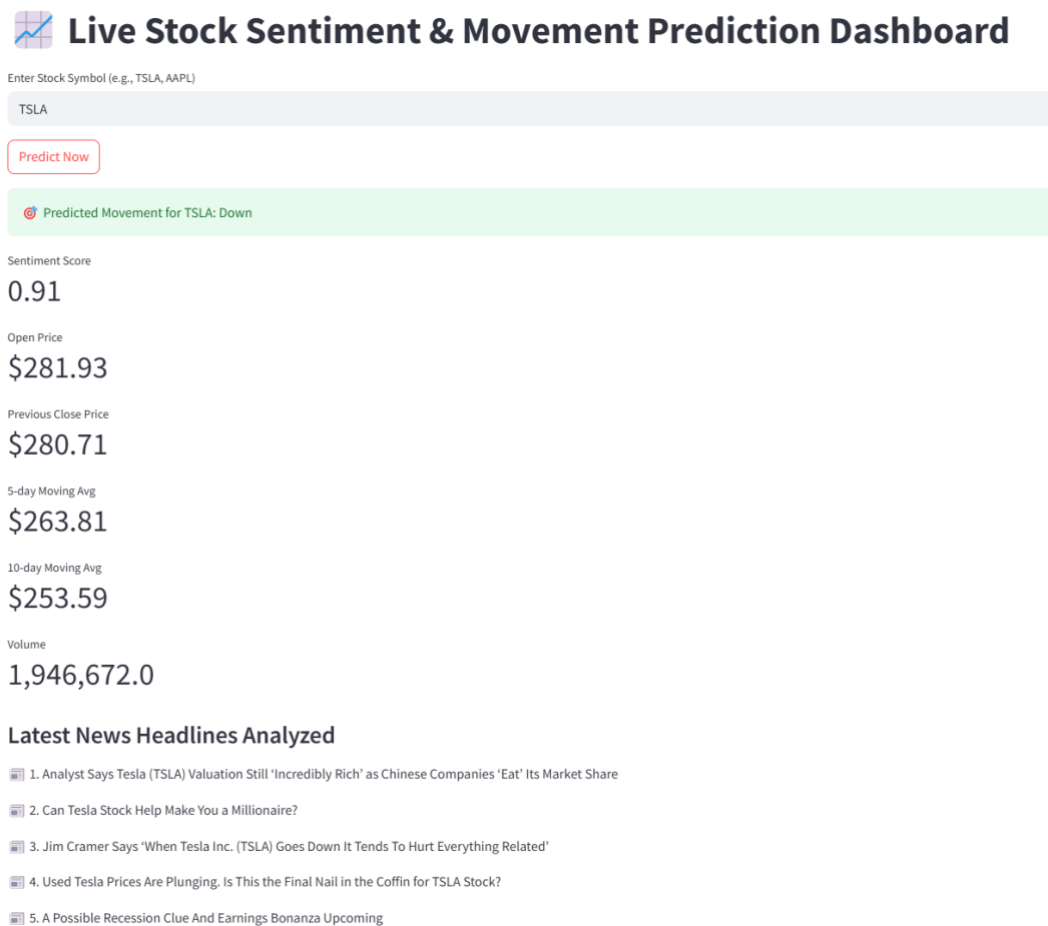- Top 5 Latest News Headlines (analyzed for prediction)



**📈 Live Stock Sentiment & Movement Prediction Dashboard**

Enter Stock Symbol (e.g., TSLA, AAPL)

TSLA

Predict Now

🎯 Predicted Movement for TSLA: Down

Sentiment Score

0.91

Open Price

$281.93

Previous Close Price

$280.71

5-day Moving Avg

$263.81

10-day Moving Avg

$253.59

Volume

1,946,672.0

**Latest News Headlines Analyzed**

📰 1. Analyst Says Tesla (TSLA) Valuation Still 'Incredibly Rich' as Chinese Companies 'Eat' Its Market Share

📰 2. Can Tesla Stock Help Make You a Millionaire?

📰 3. Jim Cramer Says 'When Tesla Inc. (TSLA) Goes Down It Tends To Hurt Everything Related'

📰 4. Used Tesla Prices Are Plunging. Is This the Final Nail in the Coffin for TSLA Stock?

📰 5. A Possible Recession Clue And Earnings Bonanza Upcoming

*Figure 14 Streamlit Dashboard*

# 9. Conclusion and Future Work:

In this project, we demonstrated that integrating financial news sentiment with stock technical indicators can improve short-term stock movement prediction. Using FinBERT for sentiment extraction and training traditional machine learning models like Random Forest, we achieved a competitive accuracy of around 70%.

The deployment of the model through a Streamlit dashboard provides a real-world, interactive experience for predicting live stock movements based on the latest data. This shows the practical potential of combining Web Mining, Natural Language Processing, and Machine Learning techniques for financial analytics.

**Key Achievements**
- Collected and cleaned large-scale financial news and stock datasets.
- Applied FinBERT, a domain-specific transformer model, for accurate sentiment classification.
- Engineered meaningful features including moving averages, volume, and past sentiment.
- Trained and compared multiple classifiers, selecting Random Forest as the best performer.

- Built a real-time, API-driven dashboard that predicts live stock sentiment and movement.

**Future Enhancements**
- Expand to include Twitter and Reddit sentiment analysis for broader sentiment coverage.
- Implement deep learning models like LSTM or transformers for time-series prediction.
- Incorporate multi-day forecasting (1-day, 5-day, 1-week stock movement).
- Deploy the application using Docker and AWS for scalability.
- Integrate SHAP value analysis for deeper explainability of predictions.