STOCK MOVEMENT ANALYSIS BASED ON SOCIAL MEDIA SENTIMENT

REPORT

SHREYASH SINGH

shreyashsingh99@gmail.com

INTRODUCTION

Background

The stock market is highly dynamic and influenced by a wide array of factors, including company fundamentals, macroeconomic conditions, and increasingly, public sentiment. Social platforms like Reddit have emerged as critical spaces where retail investors discuss and debate stock performance. Discussions on subreddits like r/wallstreetbets have been shown to impact market behavior, often driving significant stock movements. This phenomenon highlights the growing importance of integrating alternative data sources, such as social media sentiment, into predictive analytics.

Objective

This project combines sentiment analysis of Reddit discussions on AMD stock with technical stock indicators to predict stock price movements and values using machine learning techniques. The primary objectives are:

1. **Sentiment Analysis**: Quantifying sentiment in Reddit discussions about a particular stock, specifically AMD.
2. **Feature Engineering**: Merging sentiment data with technical stock indicators such as moving averages, RSI, and price volatility.
3. **Machine Learning Models**: Developing models to:
    - Predict the **closing stock price** (regression).
    - Predict the **direction of stock movement** (classification).

Scope

The project employs Natural Language Processing (NLP) for sentiment extraction from Reddit discussions and machine learning models for predictive analytics. Historical stock data is combined with sentiment scores to create a comprehensive dataset for training and testing. While this approach provides valuable insights, challenges such as noise in social media data, sarcasm detection, and the inherent unpredictability of stock markets remain. Despite these limitations, the integration of alternative data sources like Reddit discussions opens new avenues for improving stock market prediction accuracy.

This detailed analysis of AMD stock sentiment and price movement serves as a case study, demonstrating the potential of combining public sentiment with traditional market indicators for enhanced predictive performance.

DATA SCRAPEING REDDIT FOR AMD STOCK

The goal of this project is to scrape data from Reddit subreddits to gather discussions related to specific stock-related keywords. This data can be analysed to identify trends, sentiments, and insights that may influence stock movement predictions. The script utilizes the Python Reddit API Wrapper (PRAW) to extract posts and their top comments

Etraction Process Overview

1. **Setup and Initialization**:
    - API credentials (client_id, client_secret, user_agent) are used to authenticate and interact with Reddit's API securely.
    - The script initializes a Reddit object using PRAW.
2. **Data Extraction**:
    - Subreddits of interest are specified in a list.
    - Posts are searched using a keyword filter.
    - Each post's metadata (ID, timestamp, title, content) and the top 5 comments are extracted.
    - Posts are processed in batches with periodic pauses to respect API rate limits.
3. **Data Storage**:
    - Extracted data is structured into a Pandas Data Frame for ease of analysis.
    - The dataset is exported to a CSV file for storage and future use.

2. Subreddit and Keyword Selection

1. **Subreddit List**:
    - A list of subreddits to search is defined. These subreddits represent communities where discussions relevant to the stock market occur
    - (r/WallStreetBets, r/investing r/finance,r/stocks,r/investing,r/amdstock).
2. **Keyword**:
    - A specific keyword ("AMD") is used to filter posts related to that topic. This ensures that the script retrieves discussions relevant to the selected stock or market trend.

Features Extracted

Feature	Description	Relevance to Stock Predictions
Post Title	Title of the post.	Indicates the primary topic or sentiment of the discussion.
Post Body	Main content of the post.	Provides detailed context and user insights related to the stock or market conditions.
Top Comments	Top 5 comments sorted by relevance.	Reflects community sentiment and key opinions, potentially signaling collective investment behavior.
Subreddit	Name of the subreddit where the post was found.	Helps identify the focus of discussions (e speculative trading in r/WallStreetBets vs. technical analysis in r/investing,r/finance,).
Timestamp	Date and time of post creation.	Enables temporal analysis to correlate discussions with stock price movements or market events.

Challenges and Solutions

Challenge	Details	Solution
API Rate Limits	Reddit's API imposes limits on the number of requests within a time window.	Introduced a pause (time.sleep) every 50 posts processed to avoid exceeding limits.
Incomplete Comments	Nested or hidden comments can make it difficult to extract the top 5 relevant comments.	Used submission.comments.replace_more(limit=0) to load all comments and retrieved only the top 5.
Keyword Relevance	Search results might include posts unrelated to stock movement predictions.	Assumed the keyword (search_keyword) is specific enough to filter noise; future iterations could include more sophisticated filtering based on sentiment or keyword co-occurrence.
Large Data Volume	Searching large subreddits can result in	Restricted the number of posts per subreddit with post_limit_per_subreddit.

Challenge	Details	Solution
	excessive data, leading to slow processing.	
Data Quality	Some posts or comments are irrelevant, too lengthy, or poorly structured.	Future enhancements could include filtering based on engagement metrics (e.g., upvotes) and truncating excessively long comments.

The final dataset contains the following columns:

Column	Description
ID	Unique identifier for each Reddit post.
Subreddit	The subreddit where the post was found.
Discussion	Combined content of the post and its top 5 comments.
Datetime	Timestamp of post creation in YYYY-MM-DD HH:MM format.

Scraping Methodology

The scraping process can be broken down into several well-defined steps:

1. Search Method

- **Purpose**: Search posts within specified subreddits using a given keyword.
- **Implementation**:
  - For each subreddit, the subreddit.search() function is used to find posts containing the specified keyword.
  - The limit parameter restricts the number of posts retrieved per subreddit.

For submission in subreddit.search(keyword, limit=post_limit_per_subreddit):

2. Extract Post Metadata

- **Purpose**: Gather essential information from each post for analysis.
- **Extracted Metadata**:
  - **Post ID**: A unique identifier for the post.
  - **Title**: A short summary or the main topic of the post.
  - **Selftext**: The body/content of the post.
  - **Timestamp**: The creation time of the post, converted to a human-readable format using the convert_timestamp() function.

3. Extract Top Comments

1. **Load Comments**:
   - Fetch comments using the submission.comments attribute.
   - Use replace_more(limit=0) to load all top-level comments and replace placeholders with actual content.

     submission.comments.replace_more(limit=0)

2. **Sort Comments**:
   - Sort comments by relevance (submission.comment_sort = "top") to ensure the most valuable or upvoted comments are retrieved.
3. **Select Top 5 Comments**:
   - Extract the first 5 comments using a list comprehension.
     top_comments = [comment.body for comment in submission.comments[:5]]

4. Data Structuring

1. **<u>Combine Data</u>**:
    - Combine post details and the top 5 comments into a single "Discussion" field. Each comment is enumerated for clarity.

```
discussion_content = f"{post_content}\n\nTop Comments:\n" + "\n".join(

 [f"{i + 1}. {comment}" for i, comment in enumerate(top_comments)])
```

2. **<u>Append to Data List</u>**:
    - Add each post's combined data to a list of dictionaries.

```
posts_data.append

    "ID": post_id,

    "Subreddit": subreddit_name,

    "Discussion": discussion_content,

    "Datetime": post_datetime
```

<u>5. Handle API Rate Limits</u>

1. **API Constraints**:
    - Reddit's API imposes limits on the number of requests within a specific time.
2. **Pause After Batches**:
    - To avoid exceeding API limits, the script pauses for a specified duration after processing every 50 posts.

```
 if idx % 50 == 0:

time.sleep(sleep_time)
```

6. Store Data in DataFrame

1. **Convert to DataFrame**:
   - Organize the collected data into a Pandas DataFrame for easier manipulation and analysis.

     ```
     df = pd.DataFrame(data)
     ```

2. **Export to CSV**:
   - Save the DataFrame as a CSV file for further analysis or integration with other systems.

```
df.to_csv(output_file, index=False)
```

7. Output

- The final dataset contains:
  - **Post ID**
  - **Subreddit Name**
  - **Combined Discussion** (post content + top 5 comments)
  - **Human-Readable Timestamp**
- The dataset is saved as a CSV file named reddit_stock_keyword_posts_with_comments.csv.

Stock Movement Prediction Using Sentiment Analysis and Technical Indicators

1. Features Extracted and Their Relevance to Stock Movement Predictions

The model leverages a combination of sentiment analysis features derived from Reddit discussions and technical stock indicators to predict stock prices and movement direction.

Sentiment Features:

- **Discussion Count**: Indicates public interest in the stock. Peaks in discussions often correlate with significant stock movements.
- **VADER Compound Score**: Measures the overall sentiment polarity (positive/negative/neutral) of Reddit discussions.
- **Transformer Sentiment Score**: A binary sentiment score (positive or negative) based on a transformer-based NLP model. Adds robustness to the sentiment analysis.
- **Negative, Neutral, Positive Scores**: Specific sentiment breakdowns from the VADER model for finer granularity.

Technical Indicators:

- **SMA (Simple Moving Average)**: Tracks price trends over 20 and 50 days, identifying support and resistance levels.
- **EMA (Exponential Moving Average)**: Gives more weight to recent price data, enhancing the model's responsiveness to market trends.
- **RSI (Relative Strength Index)**: Highlights overbought/oversold conditions, often preceding price reversals.
- **Price Change**: Captures the daily percentage change in stock price, essential for momentum-based predictions.
- **Volatility**: Measures price fluctuation over a rolling window, indicating market uncertainty.

These features are relevant because they combine public sentiment with quantitative metrics, providing a holistic view of factors influencing stock movement.

MEATHODLOGY

1. Importing Libraries

The program imports the necessary libraries for data handling, visualization, NLP, and machine learning:

- **Data Handling**: pandas, numpy for managing and processing data.
- **Visualization**: matplotlib, seaborn for plotting trends and evaluation metrics.
- **Sentiment Analysis**:
  - nltk for classical sentiment analysis (VADER).
  - transformers for modern Transformer-based sentiment analysis.
- **Stock Data**: yfinance to fetch historical stock data.
- **Machine Learning**: xgboost for predictive modeling.

2. Class Definition

The core logic is encapsulated within a class called AdvancedStockSentimentPredictor, which organizes functionalities for modular and reusable code.

Attributes:

- stock_symbol: The stock ticker to analyze.
- reddit_data: DataFrame containing Reddit discussions.
- stock_data: DataFrame containing historical stock prices.
- features: A consolidated DataFrame for machine learning.

Methods:

- **Data Cleaning**: Prepares and merges Reddit discussions and stock prices.
- **Sentiment Analysis**: Generates sentiment scores for Reddit data.
- **Feature Engineering**: Extracts stock indicators and combines them with sentiment features.
- **Model Training**: Trains regression and classification models.
- **Evaluation**: Evaluates models and visualizes results.

3. Data Preprocessing

Reddit Sentiment Data:

- **Input**: A CSV file containing columns like Datetime (timestamp) and Discussion (text).
- **Cleaning Process**:
    - Removes non-alphabetic characters, URLs, and converts text to lowercase.
    - Aggregates daily discussions and counts the total number of comments per day.

Stock Market Data:

- **Input**: Fetched using yfinance for the specified stock ticker.
- **Data Adjustments**:
    - Date alignment with Reddit sentiment data.
    - Handling missing values (e.g., using forward-fill for stock prices).

4. Sentiment Analysis

The program integrates two sentiment analysis techniques:

VADER Sentiment:

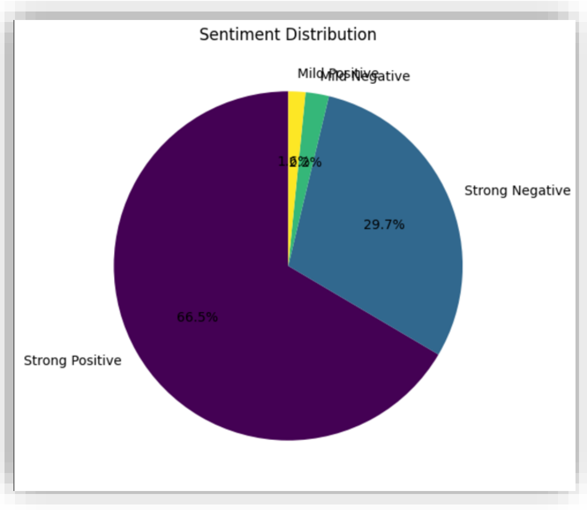
- Returns positive, negative, neutral, and compound scores.
- The compound score summarizes the overall sentiment.

Transformer-Based Sentiment:

- Utilizes Hugging Face models (e.g., distilbert-base-uncased) to classify discussions as **positive** or **negative**.

Combining Sentiments:

- A **weighted average** combines VADER and Transformer scores for better accuracy.
- The final sentiment label categorizes sentiment into:
    - Strong Positive
    - Mild Positive
    - Neutral
    - Mild Negative
    - Strong Negative



5. Feature Engineering

Key features are extracted from both datasets:

Sentiment Features:

- vader_compound: The compound score from VADER.
- transformer_sentiment: Binary sentiment from the Transformer model.
- discussion_count: Total Reddit discussions per day.

Stock Market Indicators:

- **Simple Moving Average (SMA)**: 20-day and 50-day averages to identify trends.
- **Exponential Moving Average (EMA)**: Provides more recent price emphasis.
- **Relative Strength Index (RSI)**: Momentum indicator, ranges between 0-100.
- **Price Change**: Day-to-day percentage change in stock price.
- **Volatility**: Standard deviation of price over 14 days.

Merged Dataset:

- Merges sentiment and stock features on the Date column.
- Creates a feature-rich dataset for predictive modelling.

6. Machine Learning Models

Model Selection:

Two models are implemented to address the predictive tasks:

1. **Regression**:
    - **Model**: XGBRegressor
    - **Objective**: Predict the next day's closing stock price.
2. **Classification**:
    - **Model**: XGBClassifier
    - **Objective**: Predict whether the stock price will rise (1) or fall (0) the next day.

Training and Testing:

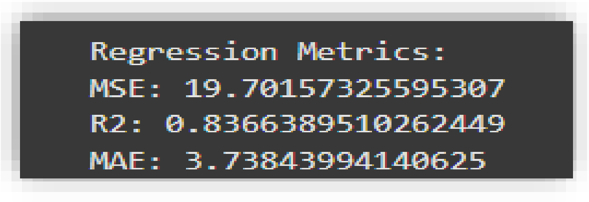
- Splits data into training (80%) and testing (20%).
- Applies scaling using StandardScaler to normalize feature values.
- Evaluates models using appropriate metrics.

Model Evaluation Metrics and Performance Insights

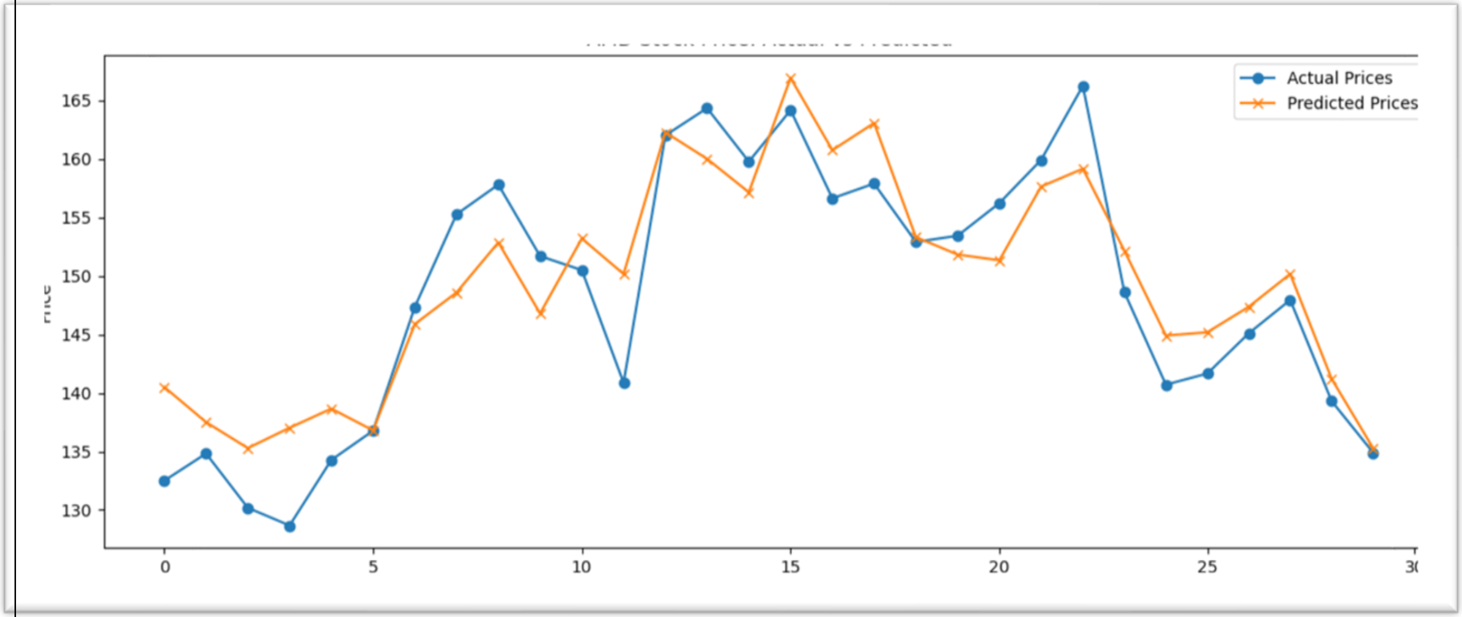
Regression Metrics

The regression model was evaluated on its ability to predict stock price trends using sentiment data. The following metrics summarize its performance:

- **Mean Squared Error (MSE):**
  19.7019.7019.70 – Indicates the average squared difference between the predicted and actual stock prices. A lower MSE reflects better model accuracy.
- **R-Squared (R²):**
  0.8370.8370.837 – Suggests that approximately 83.7%83.7\%83.7% of the variance in stock prices can be explained by the model, signifying a strong relationship between the features and the target variable.
- **Mean Absolute Error (MAE):**
  3.743.743.74 – Represents the average absolute difference between predicted and actual prices, giving a clear idea of the prediction error magnitude.



```
Regression Metrics:
MSE: 19.70157325595307
R2: 0.8366389510262449
MAE: 3.73843994140625
```



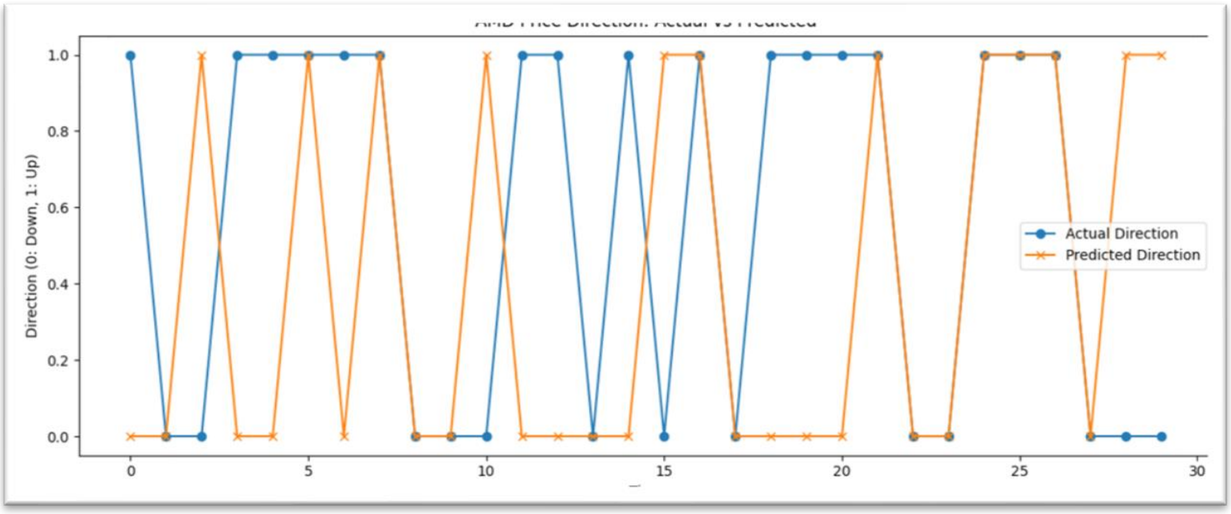
Classification Metrics

For the binary classification task, sentiment data was used to predict whether stock prices would increase or decrease. The performance is summarized below:

- **Accuracy:**
  50% – Indicates the proportion of correctly classified instances out of the total.
- **Detailed Classification Report:**
  The classification report highlights the precision, recall, and F1-score for each class.

Metric	Class 0 (Decrease)	Class 1 (Increase)	Weighted Average
Precision	0.440.440.44	0.580.580.58	0.520.520.52
Recall	0.620.620.62	0.410.410.41	0.500.500.50
F1-Score	0.520.520.52	0.480.480.48	0.500.500.50

  - *Class 0* (Decrease) has better recall, suggesting the model is better at identifying price drops.
  - *Class 1* (Increase) has slightly higher precision, meaning fewer false positives in predicting price increases.



Insights from Metrics

1. **Regression Analysis:**
    - The regression model demonstrates robust performance with a high R2R^2R2 value, indicating it can explain a significant portion of the variance in stock prices.
    - The MSE and MAE values are reasonably low, suggesting predictions are generally close to the actual values.
2. **Classification Task:**
    - The model distinguishes between stock price increases and decreases, as indicated by the 50% accuracy and balanced F1-scores.

The performance could potentially be improved with further feature engineering or by addressing class imbalances in the dataset.

3. **Overall Performance:**
    - The sentiment data shows promise in explaining stock price movements for regression but is less effective for binary classification.
    - Refining the sentiment scoring mechanism and incorporating additional features, such as trading volume or external market indicators, could enhance performance

4. Suggestions for Future Expansions

Integrating Multiple Data Sources:

- **News Sentiment**: Leverage news headlines and articles for market sentiment, as they significantly influence stock prices.
- **Institutional Data**: Include trading volumes, insider trading activities, or macroeconomic indicators for comprehensive analysis.
- **Sentiment Source Expansion**: Integrate data from other platforms like Twitter or news articles for a broader sentiment analysis

Improving Prediction Accuracy:

- **Ensemble Models**: Combination predictions from multiple algorithms (e.g., Random Forest, SVM) to enhance robustness.
- **Dynamic Weighting for Sentiment Analysis**: Adjust sentiment score weights based on the time proximity to the trading day.

Expanding Application Scope:

- **Real-Time Predictions**: Develop a live dashboard to predict intraday stock movements.
- **Portfolio Optimization**: Extend the model to recommend stock portfolios based on predicted movements.
- **Risk Assessment**: Use volatility and sentiment metrics to calculate investment risks.