Stock Movement Analysis Based on Social Media Sentiment

Introduction

The increasing influence of social media platforms on financial markets has led to a surge in sentiment-based stock price prediction. This project leverages Telegram channel discussions to predict stock movements using machine learning and natural language processing (NLP**)** techniques. The main goal is to extract sentiment from social media messages, engineer features, and train machine learning models to predict whether a stock will go up, down, or remain neutral.

Objective

Develop a machine learning model that predicts stock movements by scraping data from social media platforms like Twitter, Reddit, or Telegram. The model should extract insights from user-generated content, such as stock discussions, predictions, or sentiment analysis, and accurately forecast stock price trends.

Task Requirements

1. **Data Scraping:**
    - Select one platform: Twitter, Reddit, or Telegram.
    - Scrape relevant data from specific handles, subreddits, or Telegram channels focused on stock market discussions and predictions.
    - Clean and preprocess the data, ensuring it's ready for model input (e.g., removing noise, handling missing data).
2. **Data Analysis (Optional):**
    - Perform sentiment analysis or topic modeling on the scraped data.
    - Extract key features such as sentiment polarity, frequency of mentions, or any other indicators relevant to stock movements.
3. **Prediction Model:**
    - Build a machine learning model that takes the processed data as input and predicts stock movements.
    - Test the model's accuracy on historical data or known stock trends.
    - Provide a detailed evaluation of the model's performance, including metrics like accuracy, precision, recall, and any improvements that can be made.
4. **Technical Skills Required:**
    - Proficiency in Python, with experience in web scraping (using libraries such as BeautifulSoup, Scrapy, or Selenium).
    - Knowledge of Natural Language Processing (NLP) techniques for sentiment analysis and text mining.
    - Experience in building and evaluating machine learning models using libraries such as scikit-learn, TensorFlow, or PyTorch.

Scraping Process

1. **Data Source**
   The data for this project was extracted from Telegram channel '**Stock Phoenix**' that discuss stock prices, stock analysis, and trading strategies. The platform was chosen due to its high volume of messages and real-time user engagement.
2. **Scraping Tools**
    - **Telethon**: Used to connect with the Telegram API and scrape channel messages.
    - **Pandas**: Used to store scraped data in a structured CSV file for further processing.

3. **Data Extraction Process**
   - **API Setup**
     - API ID and API Hash were obtained from my.telegram.org.
     - A session was created to connect to the Telegram server.
   - **Message Collection**
     - Extracted 50k messages, sender details, date, and timestamps.
     - Messages were stored in telegram_channel_messages.csv with columns like: Date, Message, Message ID, Views, Forwards, Reactions.
   - **Data Storage**
     - Data was saved as a CSV file for further preprocessing and analysis.

Challenges & Resolutions during scraping

Challenge	**Description**	**Resolution**
API Rate Limits	Telegram limits the number of API calls per second.	Added delays and batch requests to avoid limits.
Duplicate Messages	Same message sent multiple times by users.	Used unique message IDs to remove duplicates.
Message Length	Short messages often have no meaningful content.	Removed messages with fewer than 3 words.
Data Privacy	Protecting user data and PII.	Only stored public messages, excluding PII.

Feature Extraction

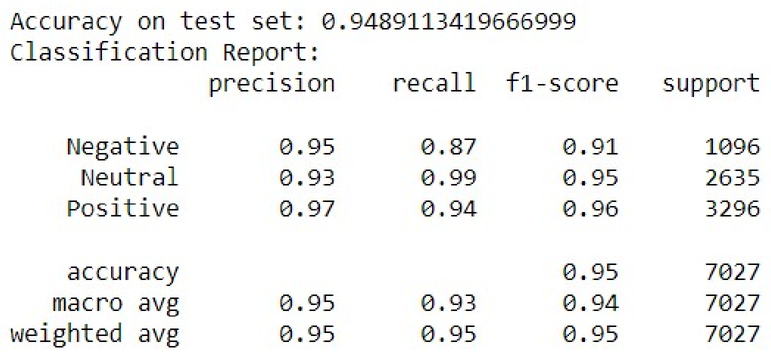
- **Text Cleaning**
  - Removed URLs (e.g., "https://...")
  - Removed Emojis (🎉, 😂, etc.)
  - Removed Stopwords (is, the, and, etc.)
  - Lowercased all words for uniformity.
  - Used WordNetLemmatizer for lemmatization.
- **Sentiment Analysis**
  - Used SentimentIntensityAnalyzer to classify each message as Positive, Negative, or Neutral.
  - Sentiment labels were used as the target variable for machine learning.
- **Feature Engineering**

Feature	**Description**	**Relevance**
Sentiment Polarity	Float score (-0.05 to 0.05) indicating sentiment.	Captures user sentiment, critical for stock trends.
Message Length	Total number of words in the message.	Longer messages often contain detailed insights.
Frequent Words	Word frequency counts (nifty, stop, profit, etc.).	Captures trading signals in discussions.
Word Embeddings	TF-IDF vectors of the text content.	Converts text into a machine-readable format.

Models and Training

- **Models Trained**
    - **Random Forest Classifier:** Simple, robust, and non-linear.
    - **Multinomial Naive Bayes:** Suitable for text classification problems.
    - **Support Vector Classifier (SVC):** Works well in high-dimensional spaces and it's relatively memory efficient.
    - **Logistic Regression:** High model interpretability, handles multiple feature types.
- **Model Training Process**
    - Data Split by Date
        - Data was split into training and testing datasets.
        - Used a time-based split (not random) to simulate a real-world prediction scenario.
    - Feature Vectorization
        - Used TF-IDF (Term Frequency-Inverse Document Frequency) to convert text to numerical vectors.
- **Model Evaluation**

  Support Vector Classifier (SVC) performed best with 95% accuracy.



```
Accuracy on test set: 0.9489113419666999
Classification Report:
              precision    recall  f1-score   support

    Negative       0.95      0.87      0.91      1096
     Neutral       0.93      0.99      0.95      2635
    Positive       0.97      0.94      0.96      3296

    accuracy                           0.95      7027
   macro avg       0.95      0.93      0.94      7027
weighted avg       0.95      0.95      0.95      7027
```

Fig 1. Classification report of SVC

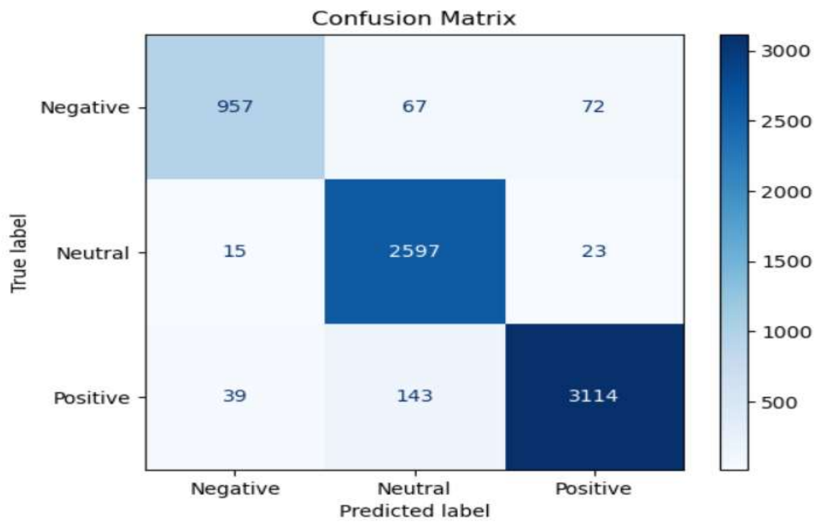
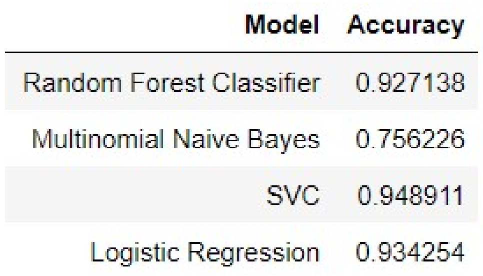


Fig 2. Confusion Matrix of SVC



Model	Accuracy
Random Forest Classifier	0.927138
Multinomial Naive Bayes	0.756226
SVC	0.948911
Logistic Regression	0.934254

Fig 3. Results Table

- **Key Insights**

- Random Forest achieved an accuracy of 93% on the test set.
- Multinomial Naive Bayes had an accuracy of 76%, but was faster to train.
- Support Vector Classifier outperformed other models with an accuracy of 95%.
- Logistic Regression performed similar to Random Forest with accuracy of 93%.

Visualizations

The WordCloud below shows the most frequently used words in stock-related messages.
Common words like **'stock', 'market', 'nifty'** dominate, reflecting trading signals in social media.

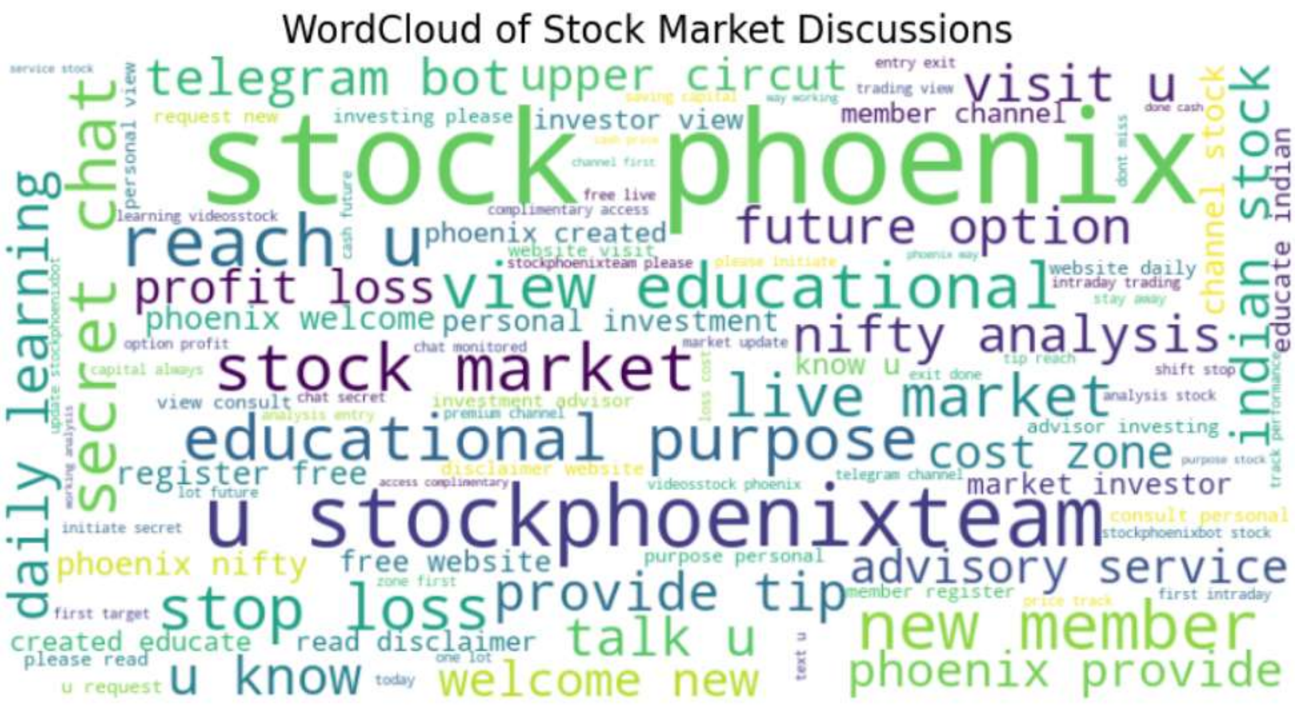


Fig 4. WordCloud

The column chart below shows the 20 most important features in stock-related messages.
Features like **'nifty', 'stop', 'profit'** are having the greatest feature importance value.

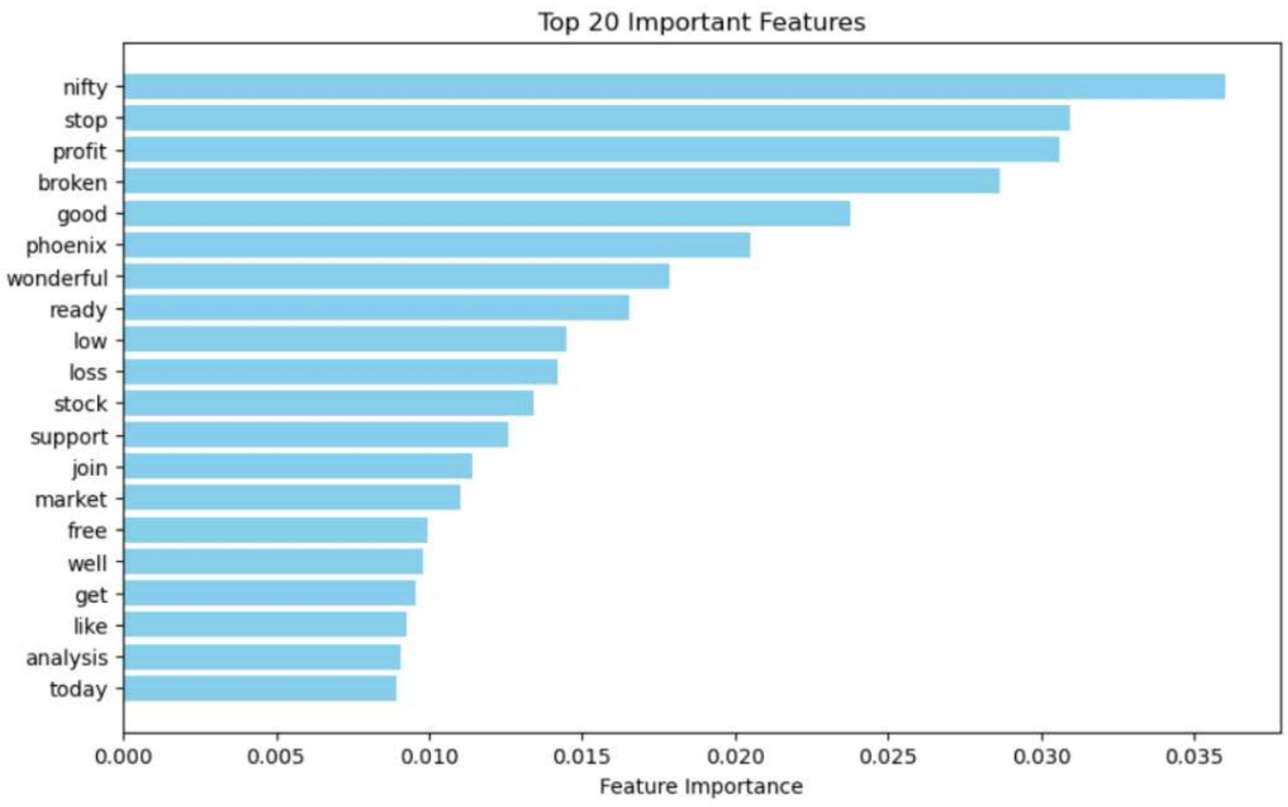


Fig 5. Top Features

The bar chart below shows the count of sentiment categories (Positive, Negative, or Neutral).
The categories are created using sentiment scores calculated using SentimentIntensityAnalyzer and then grouping them according to sentiment scores.

Sentiment score having score greater than 0.05 are grouped as Positive.

Sentiment score having score lesser than -0.05 are grouped as Negative, and others as Neutral.

This grouping tells us whether a particular message posted in a Telegram channel is positive about the stock market trend or news, or negative, or neutral. This helps the users to decide whether they should follow the owner of the channel and invest their hard earned money or not.

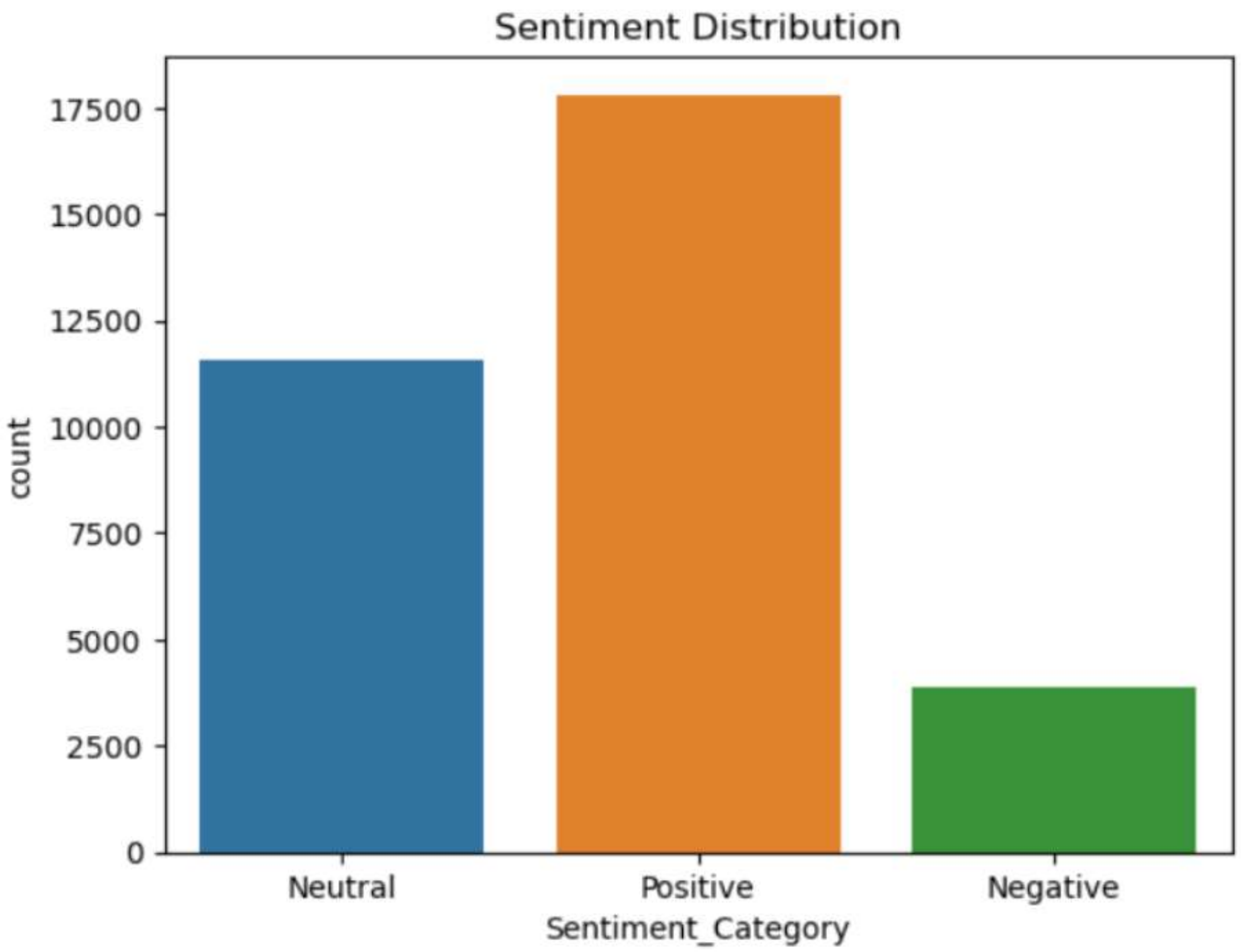


Fig 6. Sentiment Category

Conclusion

- Telegram-based sentiment analysis can effectively predict stock trends.
- Support Vector Classifier (SVC) was the best-performing model with an accuracy of 95%.
- The analysis shows that words like "nifty", "stop", "profit", and "broken" significantly affect stock trends.

Future Work

- Add Sentiment Analysis from more channels of Telegram.
- Use transformer-based models like BERT for better text classification.
- Implement LSTM or GRU models for better time-series forecasting.