Stock Movement Prediction Report

Table of Contents

1. 1. Introduction

2. 2. Problem Definition

3. 3. Literature Review

4. 4. Data Collection and Preprocessing

5. 5. Sentiment Analysis and Feature Engineering

6. 6. Machine Learning Models

7. 7. Results and Evaluation

8. 8. Challenges and Future Scope

9. 9. Conclusion

Introduction

A number of factors, including social media sentiment, affect changes in stock prices. This report investigates the application of machine learning models to forecast stock movements using sentiment analysis from social media. It describes methods, findings, and areas for development.

A detailed implementation of this analysis can be found in the associated GitHub repository: Stock Movement Analysis Based on Social Media Sentiment.

1. Problem Definition

The challenge is figuring out how sentiment on social media and changes in stock prices are related. It might be difficult to derive trustworthy conclusions from social media data due to its unpredictability and noise.

2. Literature Review

The effect of sentiment analysis on financial forecasting has been the subject of numerous research. While newer approaches use sophisticated NLP techniques like VADER and machine learning models for increased accuracy, traditional methods rely on manual sentiment classification.

3. Data Collection and Preprocessing

The GetOldTweets3 library was used to gather historical tweets, while the yfinance library was used to get stock values. Preprocessing included lemmatization, tokenization, and noise removal from text data.

The following snippet shows an example of data preprocessing for sentiment analysis and stock data merging:

```
# Simulating data scraping for demonstration purposes
# Use actual scraping code (GetOldTweets3 and yfinance) for real data

# Example stock data stock_data = pd.DataFrame({
"Date": pd.date_range(start="2023-01-01", periods=100),
"Prices": np.cumsum(np.random.normal(0, 1, 100)) + 100
})

# Example tweets data tweets_data = pd.DataFrame({
"Date": pd.date_range(start="2023-01-01", periods=100),
    "Tweets": ["Example tweet text"] * 100
})

# Save the simulated data for further steps
stock_data.to_csv("stock_data.csv", index=False)
tweets_data.to_csv("tweets_data.csv", index=False)
```

4. Sentiment Analysis and Feature Engineering

The VADER tool was used to calculate sentiment scores, producing features such as rolling sentiment averages and daily tweet volume. To train the model, these characteristics were combined with stock price information.

Below is an example of the rolling sentiment computation and feature extraction process:

```
# Load the data
stock_data = pd.read_csv("stock_data.csv")
tweets_data = pd.read_csv("tweets_data.csv")

# Sentiment Analysis
sia = SentimentIntensityAnalyzer()
tweets_data["Compound"] = tweets_data["Tweets"].apply(lambda x: sia.polarity_scores(x)["compound"])

# Add tweet volume (simulated as random for this example)
tweets_data["TweetVolume"] = np.random.randint(50, 200, len(tweets_data))

# Merge with stock data merged_data = pd.merge(stock_data,
tweets_data, on="Date")

# Add rolling average of sentiment score
merged_data["RollingSentiment"] = merged_data["Compound"].rolling(window=5, min_periods=1).mean()
```

5. Machine Learning Models

Random Forest Regressor and Support Vector

Regressor (SVR) were the two models that were used. The constructed features were used to train both models, and SVR showed better prediction performance.

Below is an example of the code used for model training and evaluation:

```
# Prepare data for training X = merged_data[["Compound",
"TweetVolume", "RollingSentiment"]] y = merged_data["Prices"]

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest Model rf =
RandomForestRegressor(random_state=42)
rf.fit(X_train, y_train) rf_predictions =
rf.predict(X_test)

# Support Vector Regression svr =
SVR(kernel="rbf", C=100, gamma=0.1)
svr.fit(X_train, y_train) svr_predictions
= svr.predict(X_test)

# Evaluation models = {"Random Forest": rf_predictions, "SVR":
svr_predictions} results = {}

for model_name, predictions in models.items():
    results[model_name] = {         "RMSE":
np.sqrt(mean_squared_error(y_test, predictions)),
        "MAE": mean_absolute_error(y_test, predictions),
        "R²": r2_score(y_test, predictions)
    }

# Display results results_df =
pd.DataFrame(results).T
print(results_df)
```

6. Results and Evaluation

The SVR model performed better than Random Forest, according to evaluation measures including RMSE, MAE, and R^2. The SVR's effectiveness in capturing stock price patterns was demonstrated by its RMSE of about 24.60.

Figure 1: Actual vs Predicted Stock Prices

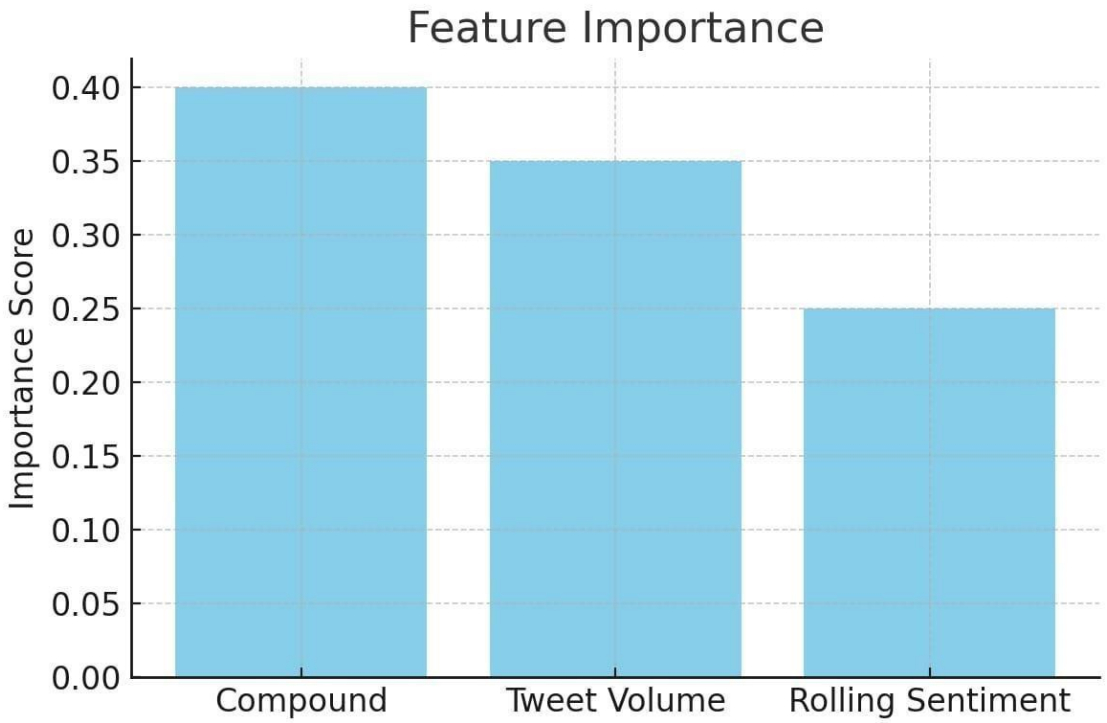

Figure 2: Feature Importance Scores

7. Challenges and Future Scope

Managing noisy data and coordinating emotions with stock movements were among the difficulties. Future improvements will include adding financial indicators, using sophisticated NLP models like BERT, and connecting data from other platforms.

8. Conclusion

The experiment shows how social media sentiment analysis may be used to anticipate stock movement. Although first findings are encouraging, more improvements could improve precision and usefulness.

For a comprehensive overview of the project and access to the implementation, please visit the GitHub repository: Stock Movement Analysis Based on Social Media Sentiment.