User Documentation for Predicting Stock Market Values Using Machine Learning and News Sentiment Analysis

---

Overview

This guide walks you through the process of predicting stock market values by combining machine learning models with sentiment analysis of news articles. The focus is on analysing sentiment data related to the oil industry and using that data to forecast stock prices for Shell (SHEL) and BP (BP). The approach involves gathering and processing data, engineering features, training models, and evaluating their performance to make informed stock predictions.

---

1. Prerequisites

To get started, make sure you have the following:

- Python Version: 3.x
- Required Libraries:
    - pandas for working with data
    - requests for making API requests
    - yfinance for pulling stock data
    - nltk for analysing sentiment
    - scikit-learn for machine learning tasks
    - matplotlib and seaborn for creating visualizations
    - mplfinance for plotting stock charts
    - concurrent.futures for running tasks in parallel

You can install all the necessary libraries with this command:

“pip install pandas requests yfinance nltk scikit-learn matplotlib seaborn mplfinance”

---

2. Data Collection and Pre-processing

2.1. Fetching News Data

- Guardian API:
  - What it does: Pulls articles from The Guardian related to the oil industry and uses the VADER tool to analyse sentiment.
  - How it works: The code collects titles, descriptions, and content from the articles, scores the sentiment, and then aggregates the results by date. The output is saved in a CSV file.
- NewsAPI:
  - What it does: Retrieves and processes articles from the NewsAPI based on a search query and date range.
  - How it works: The articles are cleaned up, duplicates are removed, and sentiment scores are calculated and aggregated by date. The processed data is then saved for further use.

2.2. Combining Sentiment Data

- Purpose: Merges sentiment data from The Guardian and NewsAPI into one dataset.
- How it works: The combined data is grouped by date, and the average sentiment score for each day is calculated. This final dataset is stored in a CSV file.

---

3. Stock Data Collection

- Purpose: Fetches historical stock data for Shell and BP using the yFinance API.
- How it works: The code retrieves key stock metrics like opening, closing, high, and low prices. This data is then aligned with the sentiment data by date and stored in a CSV file.

---

4. Feature Engineering

- Creating Lagged Features:
  - Purpose: Generates lagged features for sentiment scores and stock prices, allowing the models to learn from past data.
  - How it works: The data is shifted to create these lagged features, ensuring that the models can incorporate historical information for better predictions.

---

5. Model Training and Evaluation

5.1. Linear Regression

- Purpose: Trains Linear Regression models to predict stock prices (both opening and closing) using sentiment data and historical prices.
- Evaluation: The model's performance is evaluated using Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) to gauge accuracy.

5.2. Random Forest Regression

- Purpose: Trains Random Forest models as an alternative to Linear Regression, capturing more complex relationships within the data.
- Evaluation: Similar to Linear Regression, the model's effectiveness is measured using MSE and RMSE.

5.3. Model Comparison

- Purpose: Compares the predictive power of the Linear Regression and Random Forest models to identify which model performs better.

---

6. Data Visualization

6.1. Sentiment Distribution

- Purpose: Visualizes how sentiment scores are distributed using Kernel Density Estimation (KDE) plots.
- Insight: These plots help you understand overall sentiment trends during the analysis period.

6.2. Stock Price Visualization

- Purpose: Uses candlestick charts with moving averages to display stock price trends over time.

- Insight: These charts make it easier to spot key trends and potential shifts in the market.

6.3. Actual vs Predicted Prices

- Purpose: Plots the actual stock prices against the predicted prices to visually assess model accuracy.
- Insight: These plots provide a clear picture of how well the model's predictions align with real-world data.

---

7. Learning Curves

- Purpose: Plots learning curves for both the Linear Regression and Random Forest models to see how performance changes as the training data size increases.
- Insight: Learning curves help identify whether the models are overfitting or under-fitting, giving you a deeper understanding of their generalization capabilities.

---

8. Conclusion

This documentation provides a detailed walkthrough of how to analyse news sentiment and use it to predict stock prices with machine learning models. The modular design of the code allows for easy adaptation and scaling, so you can apply these methods to different datasets or financial instruments. The visualizations and model comparisons included in the process offer valuable insights, helping you improve the accuracy of your predictions and better understand the dynamics of the market.