Sentiment Analysis for predicting Tomorrow’s Close Price

Mohamed Ayman Mohamed

Abderlhman Fawzy Shaaban

Adham Hegazy

**Each folder includes:**

**Dataset:**

We have two datasets. The first one is the scraped one and the second is a preprocessed dataset taken from the github (you can find the link in the notebook reference section). We have trained our sentiment analysis model using the scraped one and predicted the second one to know its label.

**Sentiment Analysis notebooks:**

You can find the no**tebooks** which include the preprocessing and training of the model. Also, we have saved the weights for future work.

**Stock Price Prediction Notebooks:**

You can find thorough codes and implementation for the pre-processing, training, and prediction for tomorrow's close price using LSTM. Then, there is a portfolio to advise the investor whether he/she can buy, sell, or reject.

**An explanation for what we have done:**

After we trained the sentiment analyzer model on the dataset we had, we found that it was labeled in a biased way, and we got a very low accuracy (50%) after using its glove embedding to train a few models (Logistic Regression- Naive Bayes- LSTM). That's why we tried the data we have been scraping for two days, with each tweet automatically labeled containing its own sentiment (bearish or bullish), and by training multiple models, we could get an accuracy of 75%. The process includes cleaning the tweets (Stop Words Removal- Stock names removal- Handling special characters, links, emotions, and hashtags- stemming and Lemmatization- Tokenization) followed by applying a pre-trained Glove embedding model on each tweet to be used in the training models (Naive Bayes- Sklearn MLP Classifier).

Moving to the **Stock Price Prediction step**, we have started to construct a model with two features: average tweets sentiment and close Price for that day. Then, we used **LSTM** taking into consideration 60 previous days to predict the close day number 61. We have played with the hyperparameters to get the best accuracy and found that the best optimizer is Adam, and dense layers with 50, 25, 1 respectively with respect to the layers from the hidden to output.

**We** tried implementing the **VAR** but it did not work well (probably due to some missed tests or due to the nature of the data as it wasn't a popular model for stock prediction), even though we tried doing some research to fix it. We left the notebook regardless.

Finally, we used these predicted prices in our trading algorithm. We used the backtrader library to simulate a real portfolio with buying and selling. Currently, we simulate it for a year with $100K, and it is showing some profit. The algorithm uses the difference between the close price of the previous day and the prediction price of the following day and buys or sells accordingly. Then, we ran these steps for the FAAG companies and gave good results.