Spec Doc

Introduction

Stock price forecasting is an important and thriving topic in financial engineering especially since new techniques and approaches on this matter are gaining ground constantly. In this project, we make use of Sentiment Analysis on a set of tweets targeted at a particular company(HDFC in our case) over a period of 9 days to forecast how the market will behave in the future. The score of the sentiment analysis is then used as one of the input features for our neural network which then uses the training set along with the stock price data(obtained using the Yfinance library) for each day to get the optimum set of weights. This final regression function is then used for the prediction of future stock prices.

* The tweet data is taken over a period of 9 days only because of the restrictions of the Twitter API, however in order to make a more accurate prediction (keeping in mind to avoid overfitting) a larger training set would be recommended.

Dataset

To obtain the required Tweets, we first applied for Twitter Developer Access. Then, with the keys provided, we set up our Twitter API. We created an authentication object and an API object. After authenticating credentials, we were ready to extract the required data.

We imported the tweepy library to access the required data. We used the Cursor object and the api.search function to extract tweets which contain a specific hashtag (in our case: #HDFC). We extracted the tweets as well as the timestamps at which they were created.

The daily stock prices of the company(historical data) were obtained using the yfinance library for the same time period over which the tweets were collected. The company was selected by specifying the ticker(hdfcbank.ns in our case).

Also in order to visualize the data in a better way in the form of dataframes, we made use of the pandas and numpy libraries.

Stock data preprocessing

Stock data from the yahoo finance(throughAPI) provided us with certain heads of a stock. We further required HLPCT (stands for ”High-Low Percentage”) and PCTchange (stands for ”Percentage change”) heads as inputs for our model along with Adj CLOSE, VOLUME.

Formulae used: HLPCT = (High-Low)/High

PCTchange = (Close-Open)/Open

Another concern was of stock data for the days when markets were closed like weekends and holidays. So we went for an average value (of each head) of a previous day and the next day and used that as our data for the missing day.

Also we used Adjusted Close values instead of normal closing values of stock prices as they provide a better insight into the stock’s current status after new company offerings etc.

Twitter API and Data Preprocessing

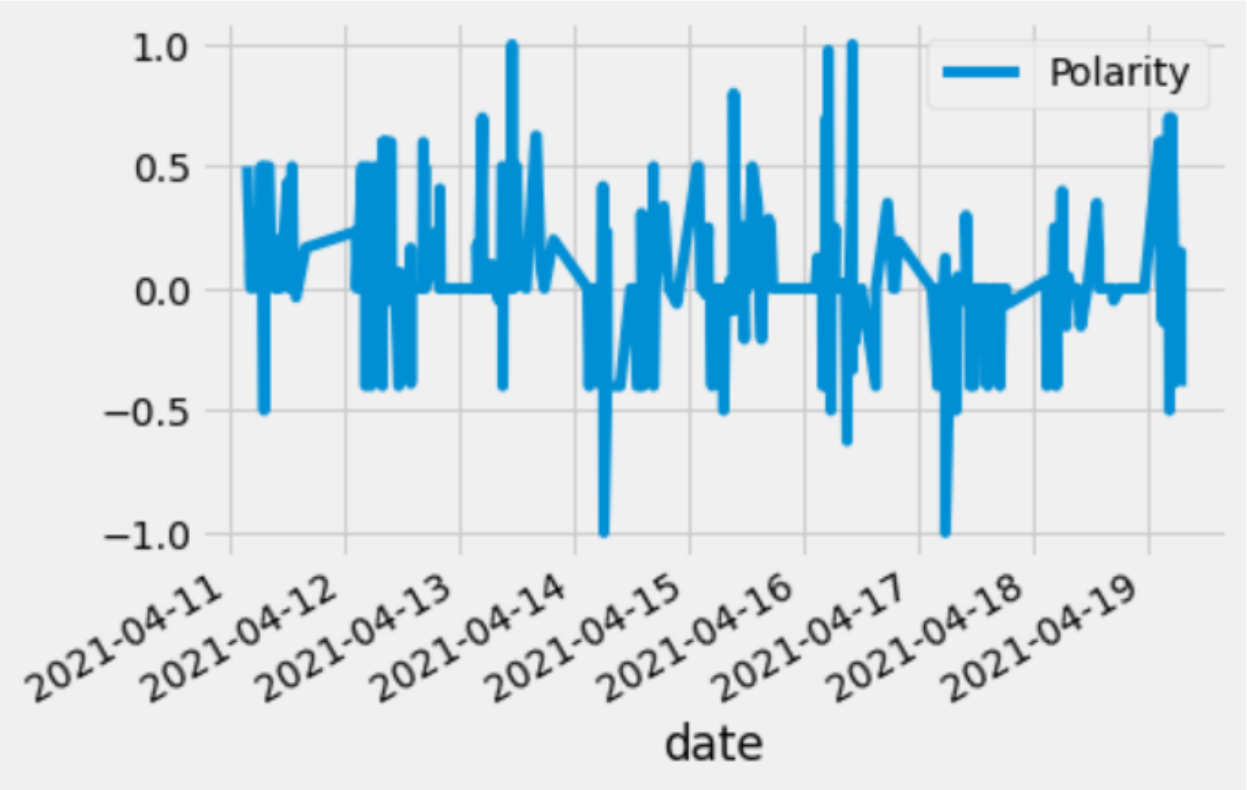
To preprocess the relevant data in order to make it suitable for sentiment analysis, we removed mentions, hashtags, hyperlinks and ‘RT’ separately using the re library (re.sub() function to be exact) to replace redundant characters . All this was enclosed in a CleanTxt() function that we defined.

We also created a word cloud using the WordCloud library to get a visual representation of the most frequently used words in the relevant tweets. This helped better understand what the relevant words of the sentiment analysis were.

Sentiment Analysis

For the Sentiment Analysis, we created two functions called getPolarity and getSubjectivity which return the Polarity and Subjectivity respectively. For this, we used the TextBlob library, which contains an API suited for sentiment analysis. For our sentiment analysis, we considered the parameter Polarity because it showed the extent of sentiment conveyed in a message.(on a scale of -1 to 1, with zero implying neutral sentiment, a positive score meaning a positive sentiment, and similarly a negative score implying a negative sentiment.)

We also created a getAnalysis function to show whether each input was negative, positive or neutral. This gave us a clue about the general sentiment of a tweet. We plotted a graph of Polarity vs. Timestamp to get an idea of the changing sentiment with time using matplotlib.



Prediction

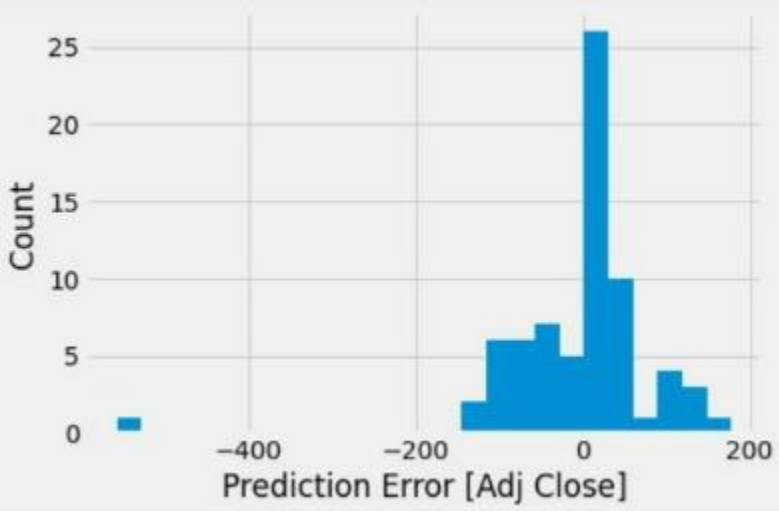
After training our sentiment analysis classifier and completing the preprocessing for the Stock Data , we created a feature matrix which contains the following features

- Polarity of tweets
- Adjusted Close price
- HLPCT
- PCTchange
- Volume

After this we divided the dataset into 2 parts, one(80%) was used for training the hypothesis function and the remaining 20% was used for testing how well the hypothesis generalizes to new data. This was done using Keras which is a part of TensorFlow, the activation function for the network was ReLU(Rectified Linear Unit) and the initial architecture that we chose for the network was 1 hidden layer with 5 neurons. However since we already had a small dataset, this architecture resulted in a high bias output hypothesis, and in order to solve that, we decided to increase the number of parameters and that could be done by increasing the number of hidden

layers and the number of neurons within each hidden layer. The final architecture was thus 8-8-8-8-1 which gave much better predictions and a lower cost value of cost function.

Predictions with 8-5-1 architecture



Predictions with 8-8-8-8-1 architecture