CMPT 353
FINAL PROJECT REPORT
Stock Market Analysis

Jyotiraditya Mayor (301401591)

Ritika Goyal (301401516)

Navjot Kaur (301404765)

Dhairya Kalra (301386482)

Introduction

Stock Market Analysis is a method used by traders to buy/sell stocks based on economic indicators, general market sentiments and technical analysis. It has become ambiguous because of a growing trend of “meme” stocks, which are primarily affected by too many people trading in the same stock at the same time. What this has created is a problem that is not easily perceivable by standard prediction algorithms since it is largely affected by the activity of people around the stock. This got us thinking if the predictions would improve when we based them on how many people were currently talking about the stock on online forums. The aim of this project is to predict the next hyped up stock by using statistical tests and machine learning models. As discussed in the Data Analysis Pipeline lecture, there are some necessary steps to follow while analysing the data. They are:

1. Figure out the question.
2. Find/acquire relevant data
3. Clean and prepare the data.
4. Analyze the data.
5. Interpret and present results.

Data analysis begins with determining what we need the data for and what is the main outcome/question we need the answer for. In this project, the most hyped stock is being examined to determine whether to buy or sell it using statistical tests and machine learning models. All 5 points listed above will be examined in this document along with how they were acquired for this project. The limitations faced during the predictions and tests are added towards the end of the report along with the project experience summary that discusses the project contribution of every project member.

Figure out the question

In response to the covid-19 crisis, countries around the world offered a variety of stimulus packages to boost their economies. Thus, the financial system began to become more liquid, causing the stock market and other asset classes to rise in price. Consequently, traders and investors are building up a heavy bullish (positive) mood.

There has been a lot of talk about GameStop stock (GME), which became popular due to a reddit thread r/WallStreetBets. We used this as a catalyst to find other similar stocks that are likely to become popular and can be a multibagger. The workflow we followed was:

1. Extracted ticker symbols of stock listed on NYSE and NASDAQ from kaggle csv files.
2. Analyzed the stock market sentiment using economic indicators such as CPI inflation, Unemployment Rate, M1 money flow, etc.
3. Extracted the 10 most talked about stocks from popular subreddits such as r/wallstreetbets, r/dogecoin, r/RobinHood and r/stocks.
4. Used Yahoo finance API to get the historical price of the most hyped stock in our list.
5. Statistical tests are used to determine whether a stock's price is related to its hype which will help us determine if the mean of the count rises with the mean of percentage change within a particular time period.
6. For stock price prediction, we have used Random Forest Regressor to perform technical analysis on the stock with indicators such as Simple Moving Average (SMA), Exponential Moving Average (EMA), and etc.
7. For stock Buy/Sell recommendation, we use Random Forest Classifier to perform technical analysis on the stock with technical indicators and predict if the stock price will increase in the next 7 days .
8. We have used Recurrent Neural Network (RNN) to predict short term closing price movements in two ways:
    a. Training RNN on open, low, high, volume and vix.
    b. Training RNN on open, low, high, volume, vix and sentiment analysis of that stock using BERT API.

Find/acquire relevant data

The data for the project is acquired using two formats:

1. Dataset files (.csv's) from kaggle
2. Real-time data (using API's)

This real-time data allows us to hedge against resource changes and thus increase our project's reliability.

1. **Kaggle Datasets**

   The data acquired from Kaggle is in CSV format. The datasets used are:

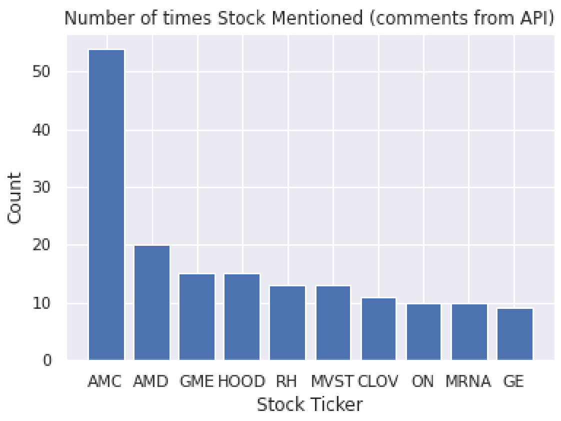
   - Ticker data: Stock ticker symbols are extracted that identify the particular company.
   - Reddit data: In order to find out which stock is being talked about the most, 4 reddit posts data sets are used from Kaggle. From there, 10 stocks are extracted which are talked about the most.

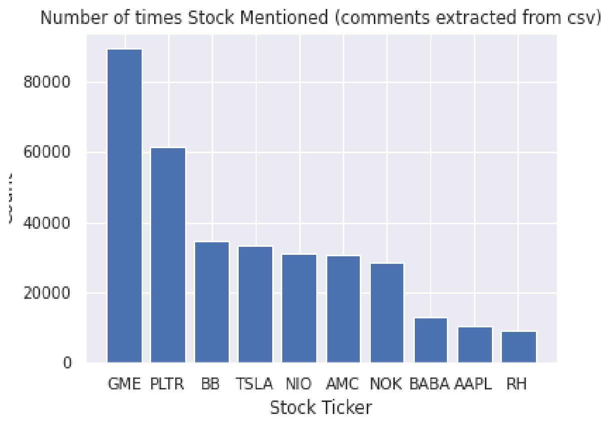
- Economic data: The economic data to find the general market sentiment is extracted.

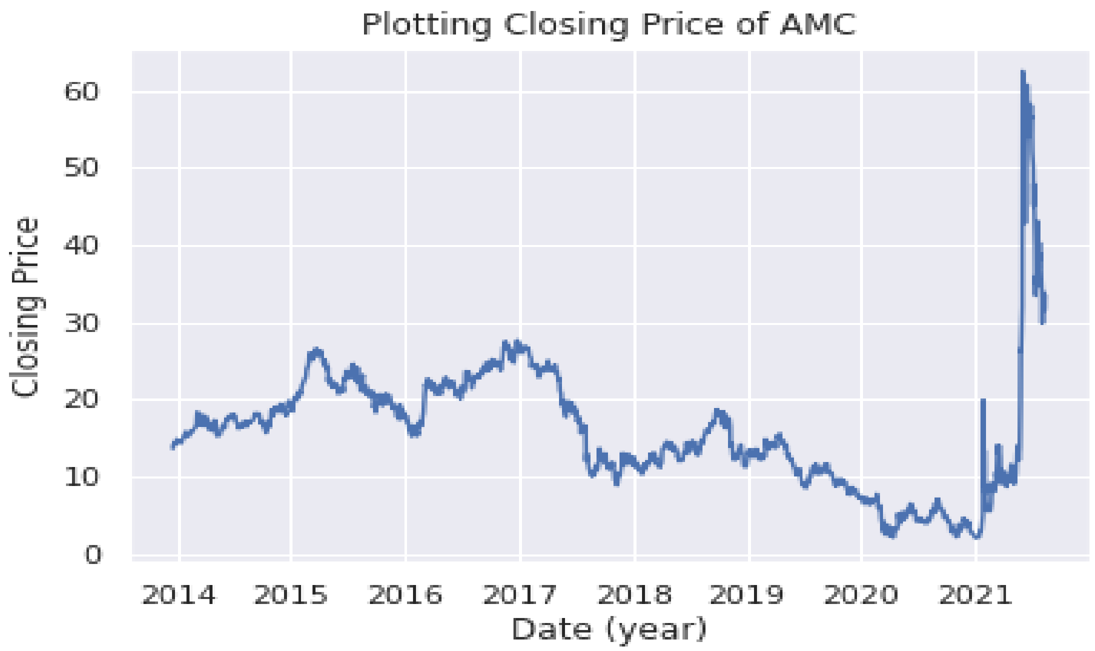
2. **Real-time Data**

The real-time data is obtained using various API's:

- Pushshift API: For reddit data
- Quandl API: For economic data
- Yahoo Finance API: For history of stocks
- BERT API: For Sentiment Analysis
- Ta-lib API: For Technical indicators



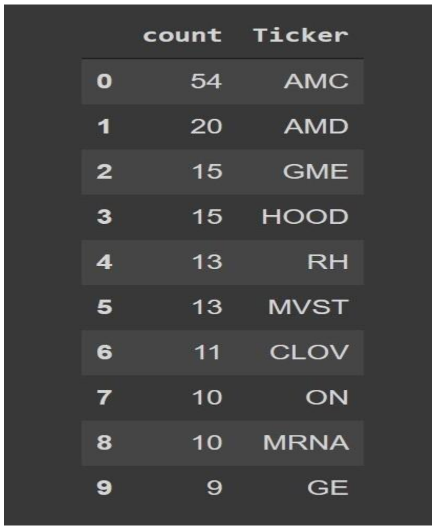




Clean and prepare the data

After the data is extracted, the CSV files are read into the Notebook so that necessary filtration can be performed on them. We used different filtering techniques to clean and prepare the data such as:

- The data files for NASDAQ and NYSE data are merged into a data frame together. An inner join on ticker symbols was performed to concatenate names with their respective stocks.
- We extracted the list of ticker symbols to count the number of times a stock was mentioned. This gives a clear picture of which stocks are in the hype. Different subreddits were used to extract the stocks from the data sets and API's with the help of regular expression to only collect the uppercase letters that are present in the list of tickers. This dataset was sorted by descending count. However, the dataset is not yet clean because of the presence of obvious words such as BUY, A, ALL, CAN, DD, YOU, etc. that were in the ticker dataset but had no relevance to the stock ticker dataset. These outliers were removed and a final list(figure) with only the relevant stocks was extracted using 'groupby' and then aggregating the tickers by their count, then the stock with the highest number of counts was analyzed.



	count	Ticker
0	54	AMC
1	20	AMD
2	15	GME
3	15	HOOD
4	13	RH
5	13	MVST
6	11	CLOV
7	10	ON
8	10	MRNA
9	9	GE

- Now the Stock Dataset that we used consisted of historical data and economic indicators. We replaced all 'NaN' values in the dataset with 0 in order to make the data more consistent for analysis and plotting purposes.

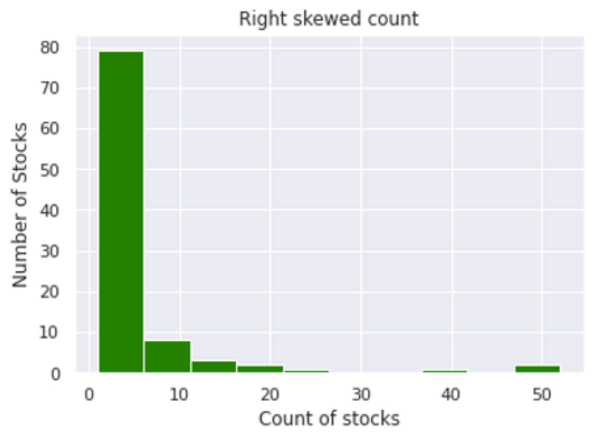
Analyze the data

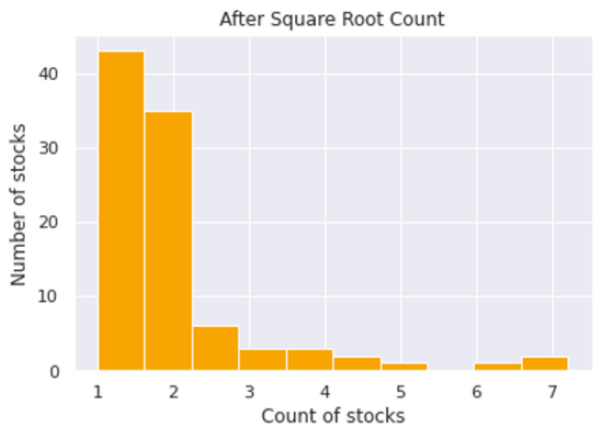
After the data is cleaned, it is ready for analysis. To analyse we used Statistical tests and Machine Learning models which are described as follows:

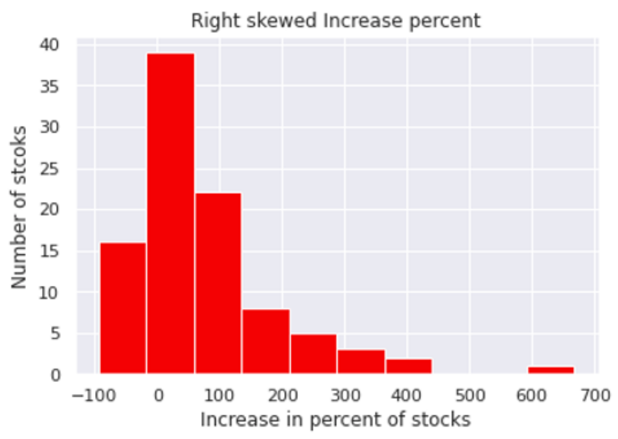
Statistical tests:

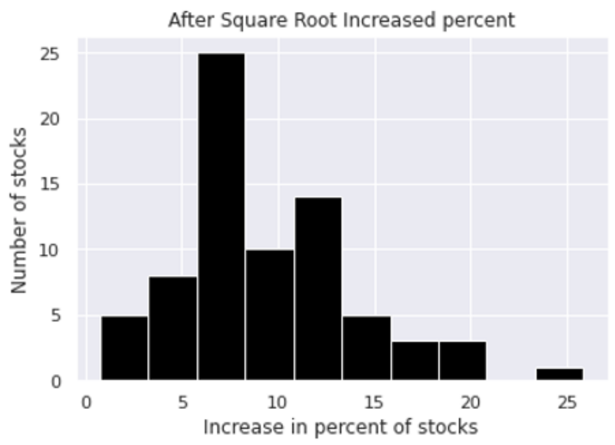
1. **Normality:**

   We do the normality test to figure out the answer for the question that whether increase in count results in increase in percentage. For that, the stocks with the highest count based on subreddit posts were extracted and together with Yahoo Finance API the initial price, final price, increase percent, and volume between a time period of one year are collected in a dataframe. We plotted histograms for count and increase percentage and both of them were right skewed which implied that the overall mean of the data was greater than the median of data. In order to normalize the data we tried transforming our right skewed data into Normal Distribution by applying 'square root' and 'log' functions to our columns so that we can apply t-test to our data.









Normal Test (Continued):-

But in this case we were able to successfully transform "increased percent" into a normal distribution but both the transformations failed on "count". The sample of histograms can be seen above.

2. **T-test:**

T-test interprets that there is sufficient evidence that an increase in count will result in an increase in percentage. Based on our values which were more than 200 in quantity we applied Central Limit Theorem to our data problem and received a $p < 0.05$ thus resulting in our desired conclusion which states that there is some evidence that increase in count results in increase in percentage based on the data.

3. **Mann-Whitney U:**

   We also applied MannWhitney U for our test and received p-value < 0.05 which states that there is some evidence that there is a difference of means between “increase percent and count.”

Outliers:

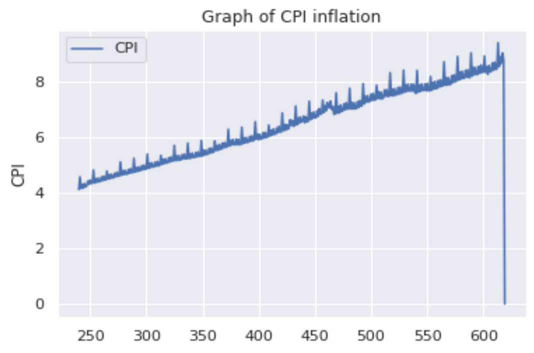
Our count data was not giving optimal results because there is a large amount of data that have count 1 or 2 which makes the results uneven so we decided to remove count less than 2 to yield appropriate predictions. Also, an increase percentage greater than 1000 is removed because in the past, there was a stock surge of a stock named GME where a lot of people invested in it resulting in its price rising by 3000 % which can result in false conclusions upon prediction of other stocks.

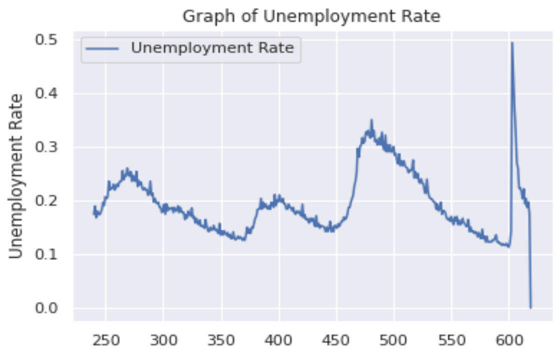
Machine learning models:

1. **Random Forest Classifier for sentiment analysis:**

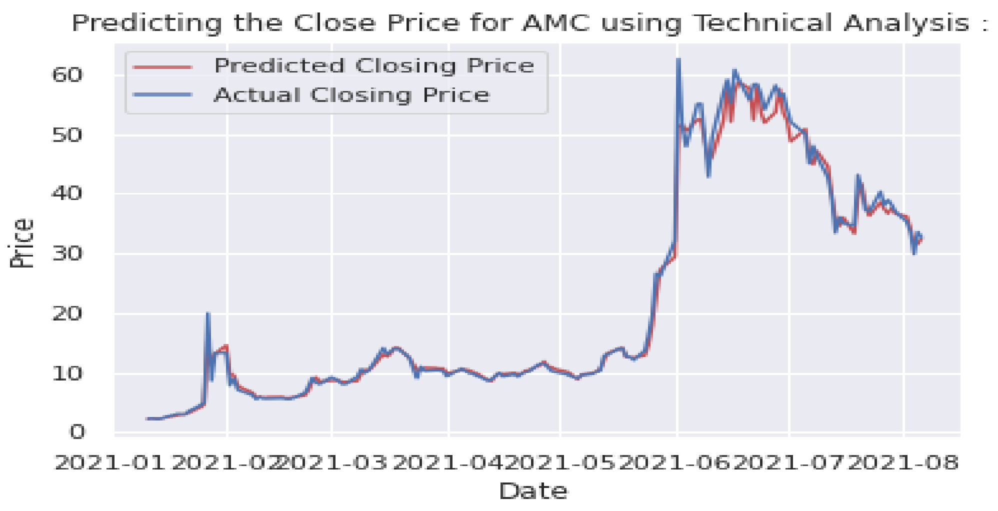
   In this model, Quandl API is used to find out current stock market sentiment. For this, general economic indicators of the US economy such as CPI inflation, Unemployment Rate, Real GDP (Percent Change), Effective Federal Funds Rate, Federal Funds Target Rate, Federal Funds Upper Target, Federal Funds Lower Target and VIX (volatility index) are used. Since most investors and traders use them a lot to gauge market sentiment, these indicators were selected. Then, monthly percentage change in S&P 500 was extracted and if the average percentage change over the month was positive, we added 1 to the sentiment column and 0 otherwise.

   With economic indicators as X_train and sentiment as y_train, the random forest classifier is trained from January 1990 to December 2020. We then average the sentiment (0 for bearish and 1 for bullish) for all of 2021 and predict bullish if the average is greater than 0.5 and bearish otherwise.



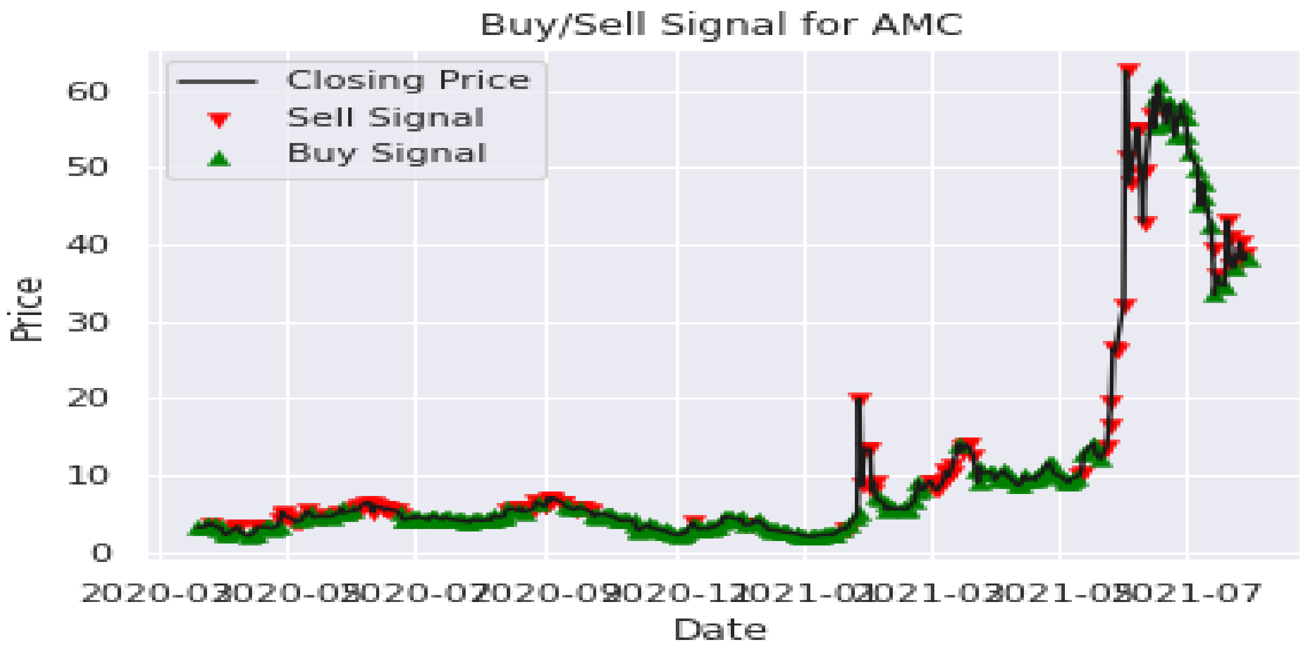


2. **Random forest regressor for stock price prediction using technical analysis:**
   After predicting the market sentiment (bullish/ bearish), the closing price of the most hyped stock was predicted for the year 2021. First, historical data and some technical indicators (SMA, EMA, RSI, ADX, ATR) were extracted from a stable stock (one that remains consistent with market sentiment) like Coca Cola (KO). KO has a history available since 1920. A random forest regressor with 20 maximum depths is used to train and fit the model. Feature scaling is applied to the dataset before fitting to scale all the features in one scale. After training the model on a stable stock , we gathered the historical data of top hyped stock and predicted the price from January 2021 to August 2021. After predicting the closing price, it was compared with the actual price.

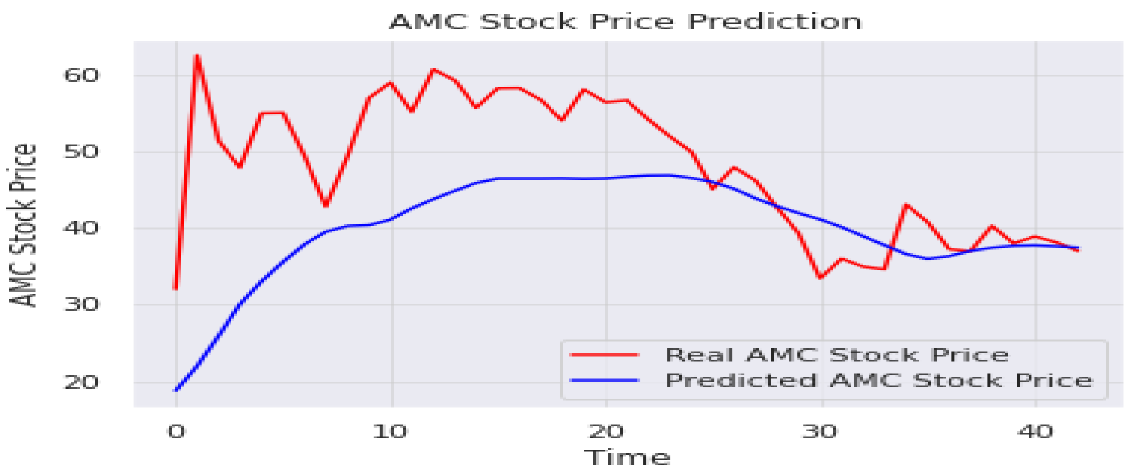


3. **Random forest classifier for buy/sell recommendation using technical analysis:**
   This part of the project included technical indicators such as SMA, EMA, RSI, ADX, and ATR (from Quandl API) for five and fifteen days of Coca Cola because it is a non-volatile stock with a long history of price changes. Here, the weekly increase in stock price is calculated and is used to create the sentiment column with a value of 1 if the increase is positive and 0 if the increase is negative.

   The random forest classifier is then trained on technical indicators as X_train and sentiments as Y_train, using Coca Cola's historical data from 1920 to February, 2020. Next, the sentiment of the hyped stock (AMC here) is predicted using its technical indicators. Based on the stock, the model produces green upward arrows for buy signals and red downward arrows for sell signals with a range of accuracy between 50 and 80 percent.

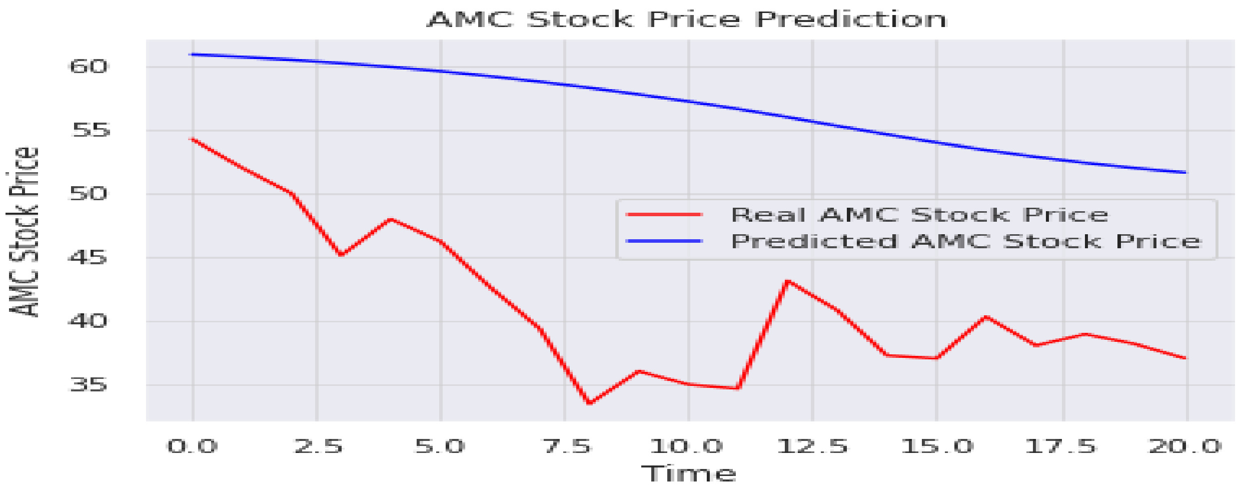


4. **Recurrent Neural Network (RNN) for price prediction:**
   A neural network is constructed in this part of the project to get a more accurate prediction of closing prices of most hyped stocks. Due to the hype factor, we gathered historical data from the last 5 years that preceded 2 months from now. To train the model, all the historical data along with the closing price of volatility (ticker: ^VIX) index were added to a dataframe and scaled accordingly. We used a five layered recurrent neural network (RNN), ranging epoch counts. After some experiments, epoch = 30 was given to it in order to achieve maximum efficiency as epoch values greater than or equal to 30 caused overfitting. The epoch was adjusted so as to cause the loss to be converging when avoiding overfitting. Using the fitted data, we predicted the closing price for the last two months.

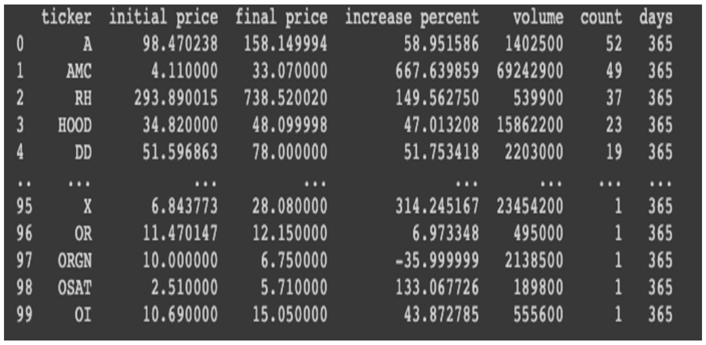


5. **Recurrent Neural Network (RNN) for price prediction using sentiment analysis:**
   This model is motivated by the above recurrent neural network. It uses a similar approach but with an extra feature of sentiment 'score'. To calculate each day's score, PushShift API is used to pull comments from top subreddits about that stock, and then the BERT API checks the sentiment of these comments, returning an integer from 1 to 5. As a feature, these returned values are added and fed into the RNN.

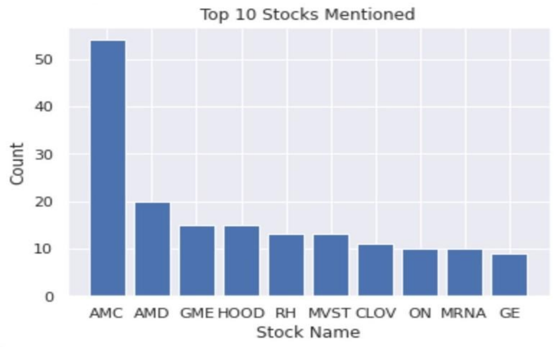
   As a result of the Mann–Whitney U test presented above, our hypothesis that an increase in sentiment would result in an increase in percentage gain was proved to be true.



6. **Linear Regressor for price prediction of stock using their counts hyped up in social media forums:**
   This model calculates the stock predictions based upon the Engineering features which we added to our model i.e, “initial price”, “final price”, “increase percent” , “count” , “volume” and “days". We used Linear Regressor for our prediction because we were focussing on predicting a number rather than predicting a category for our stock. We made a pipeline to fit our Linear Regressor model into the table. An image with the reference to our Linear Regressor model data frame has been presented below. We used the final price to be our predicted value.



```
   ticker  initial price  final price  increase percent    volume  count  days
0       A      98.470238   158.149994         58.951586   1402500     52   365
1     AMC       4.110000    33.070000        667.639859  69242900     49   365
2      RH     293.890015   738.520020        149.562750    539900     37   365
3    HOOD      34.820000    48.099998         47.013208  15862200     23   365
4      DD      51.596863    78.000000         51.753418   2203000     19   365
..    ...            ...          ...               ...       ...    ...   ...
95      X       6.843773    28.080000        314.245167  23454200      1   365
96     OR      11.470147    12.150000          6.973348    495000      1   365
97   ORGN      10.000000     6.750000        -35.999999   2138500      1   365
98   OSAT       2.510000     5.710000        133.067726    189800      1   365
99     OI      10.690000    15.050000         43.872785    555600      1   365
```

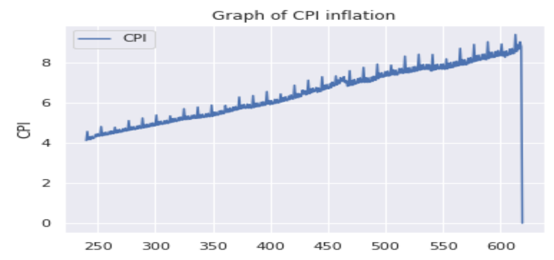


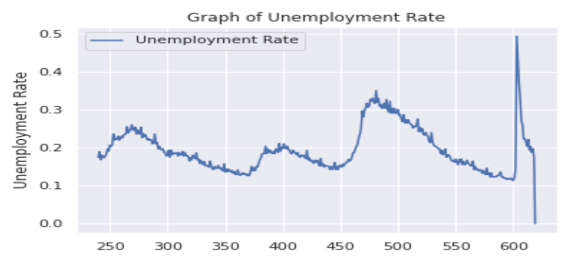
Interpretation of Results

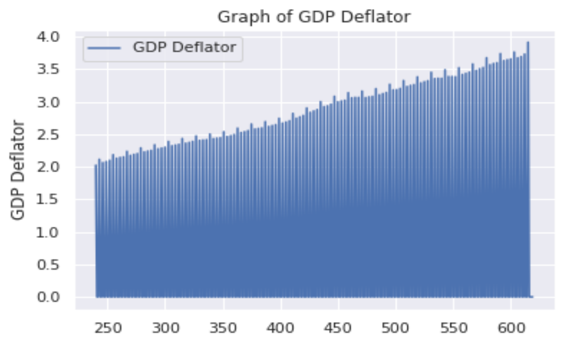
Random Forest Classifier for sentiment analysis:

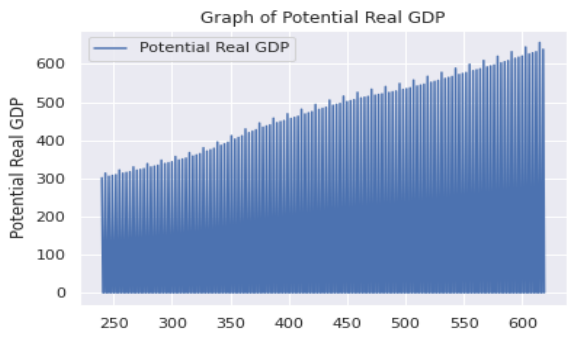
The graphs of general economic indicators are shown below, the Random Forest Classifier predicted Bullish accumulative stock market sentiment. The sudden crash to 0 value is due to lack of data for the current month (i.e 0). We cannot remove the latest month as it is important for further execution of the data.

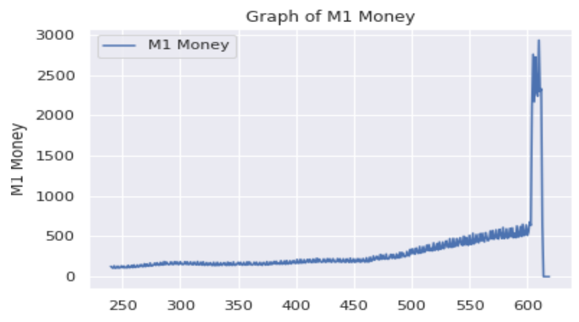
```
Accumulative Stock Market Sentiment for current year : Bullish
```

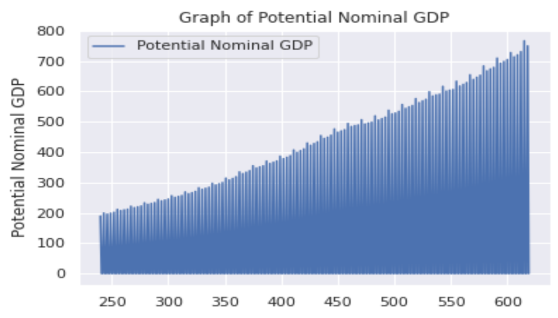


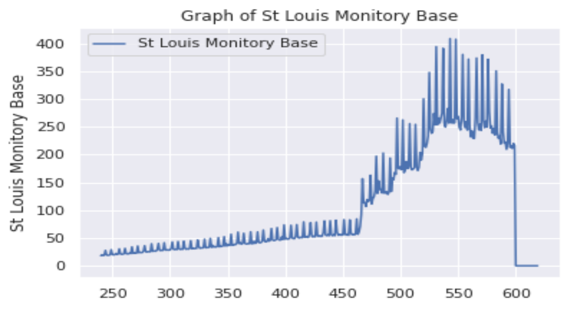


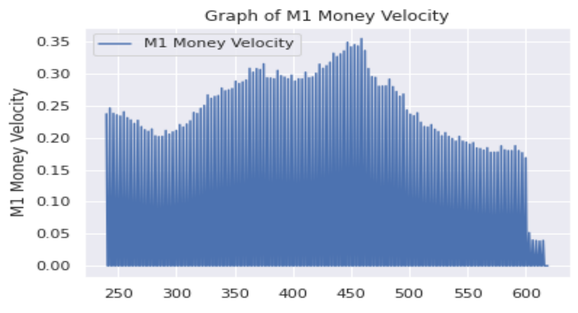


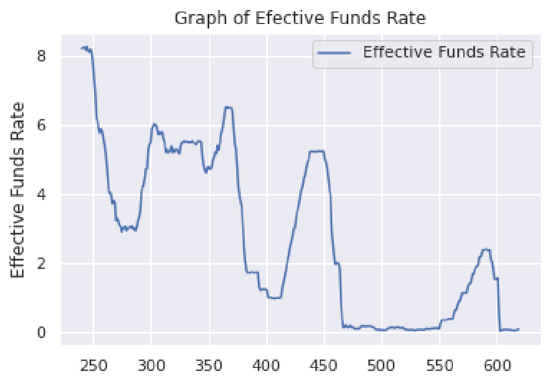


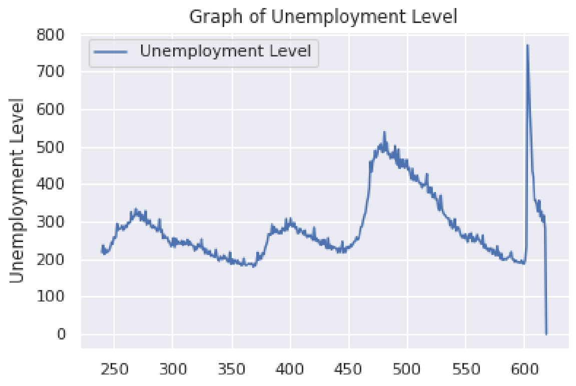


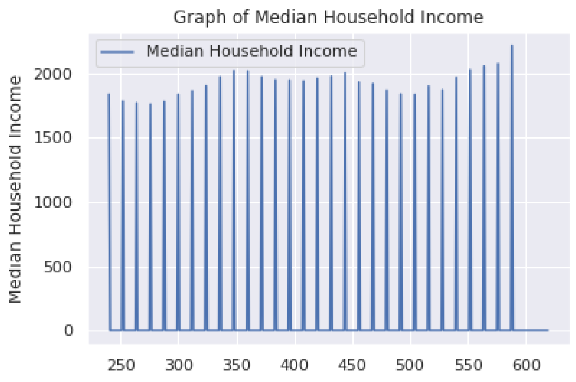


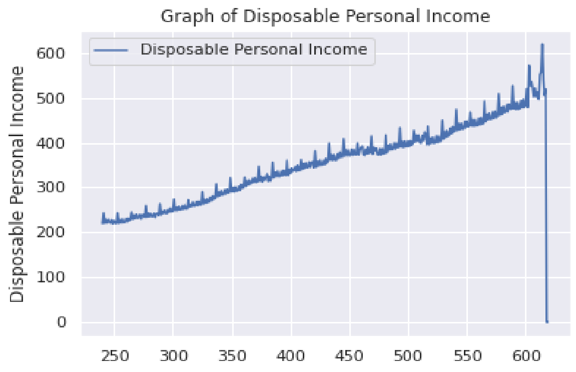


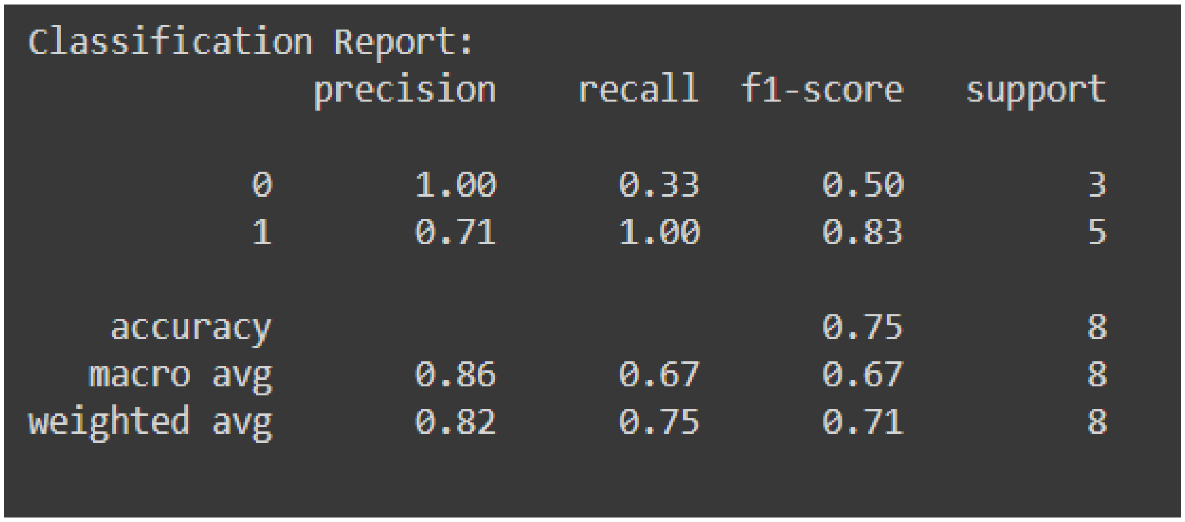






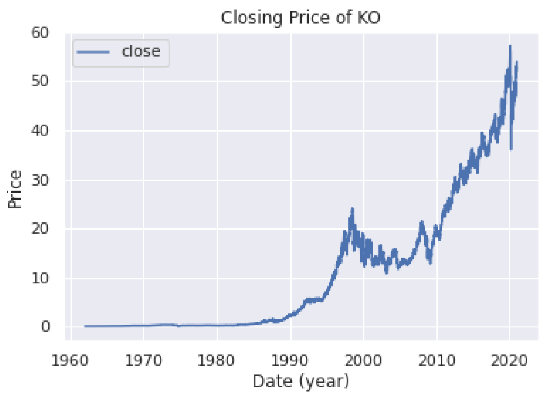


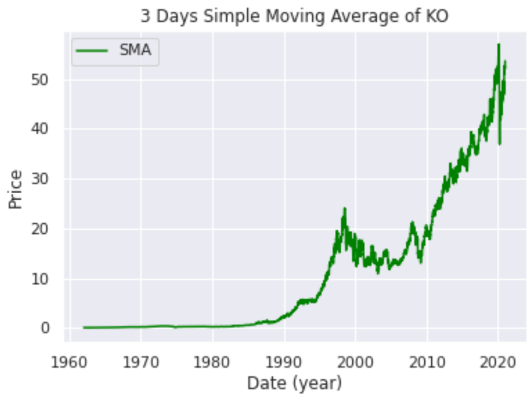


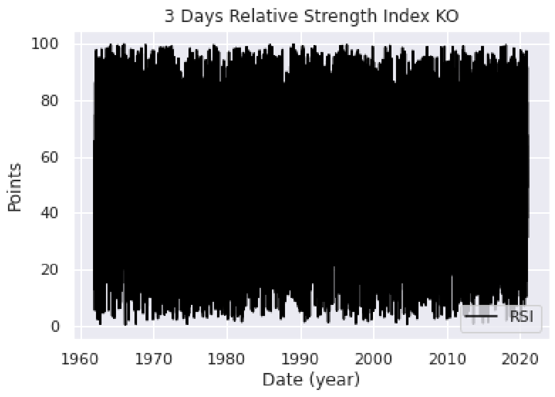

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.33      0.50         3
           1       0.71      1.00      0.83         5

    accuracy                           0.75         8
   macro avg       0.86      0.67      0.67         8
weighted avg       0.82      0.75      0.71         8
```

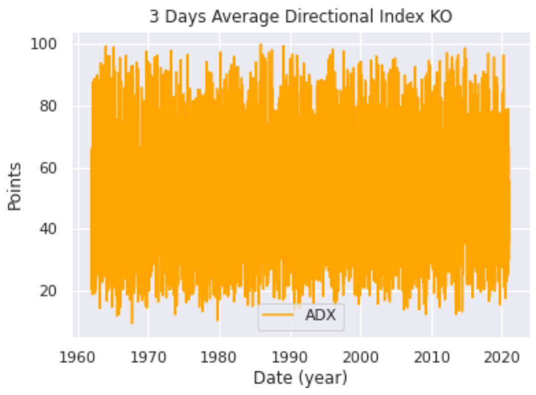
Results of Random forest regressor for stock price prediction using technical analysis:

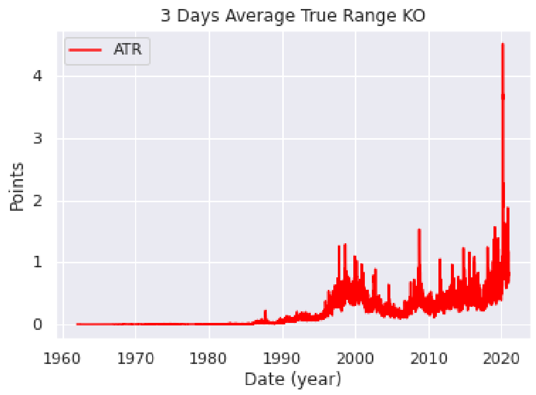
```
Training Score when trained on KO: 0.9999914569520642
Validation Score when trained on KO: 0.9795575377776621
Score after predicting the Close Price for AMC using Technical
Analysis : 0.9908845229884435
```

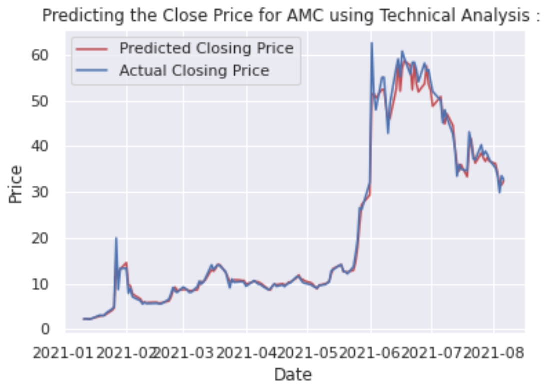






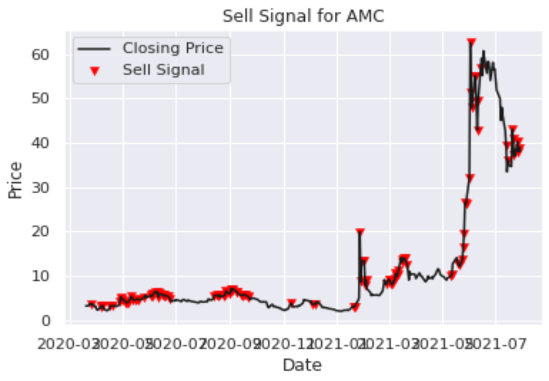


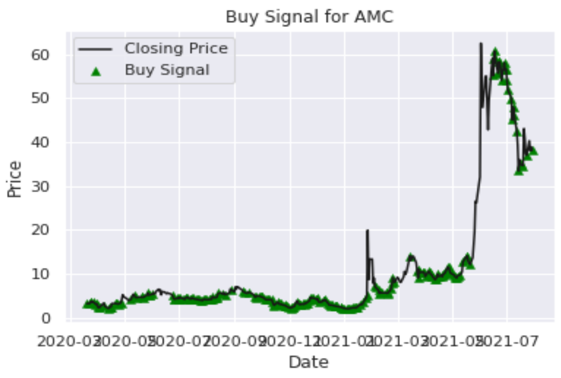


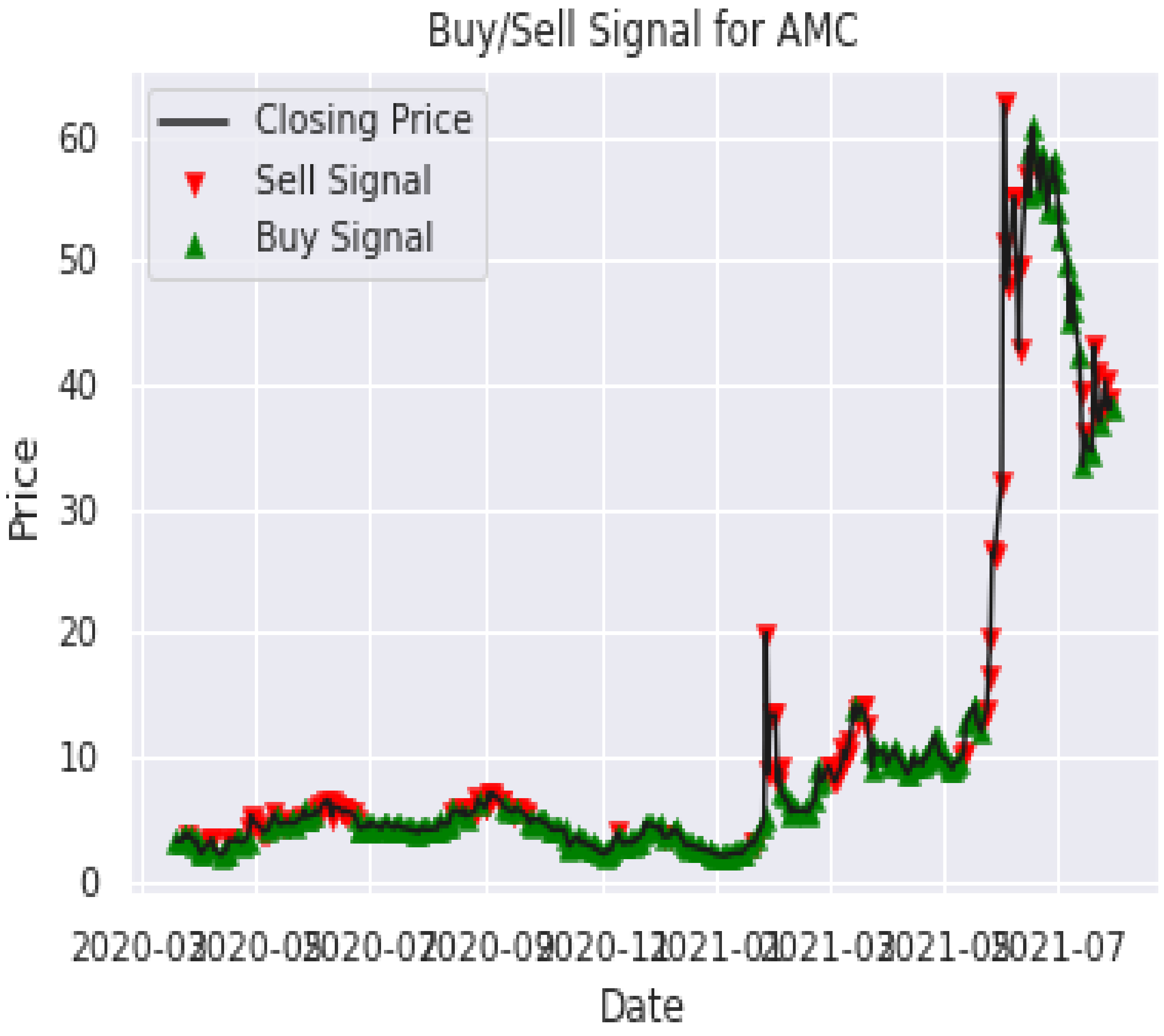


Random forest classifier for buy/sell recommendation using technical analysis:

```
Training Score When trained on Stable stock (KO): 1.0
Prediction Score with hyped Stock (AMC) : 0.5422740524781341
```

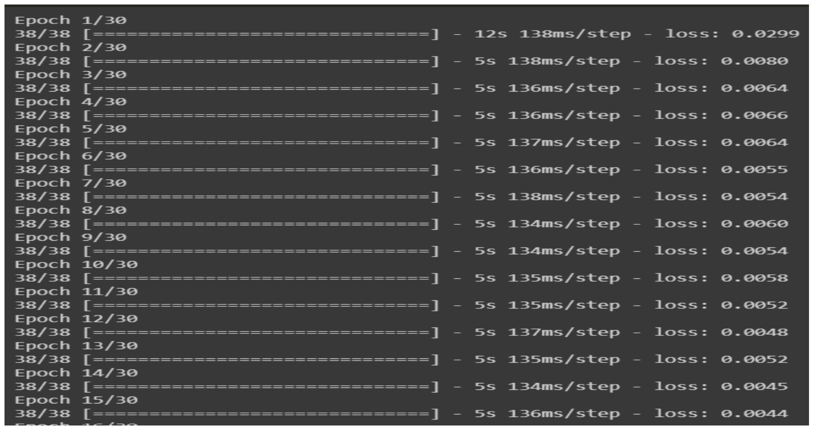




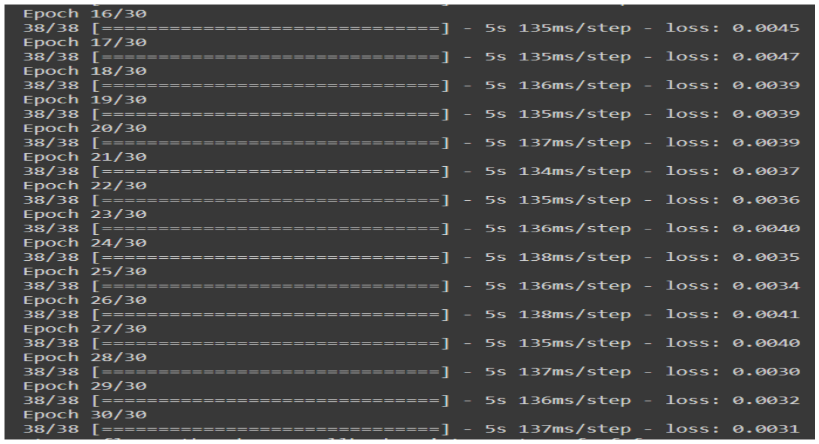


Recurrent Neural Network (RNN) for price prediction:

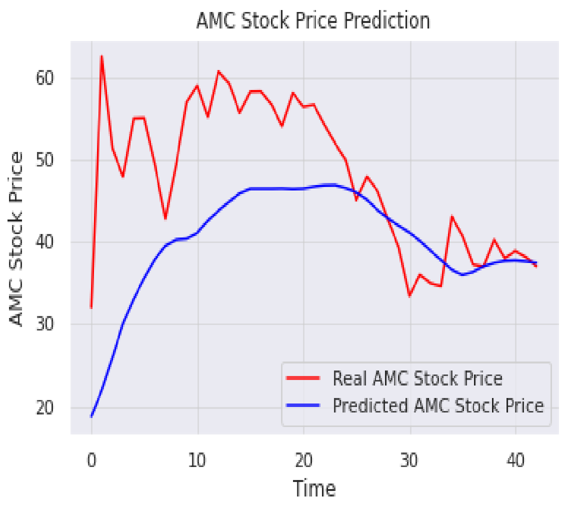
We are showing graphs of both AMC (the hyped stock here) and TSLA (in hype since 2017). Tesla shows better results because we trained the RNN on 5 year data while AMC was only trained on 1 year as it was a recent hype.

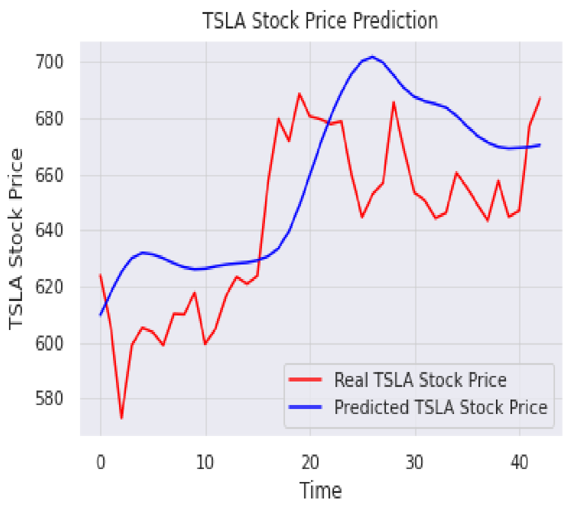


```
Epoch 1/30
38/38 [==============================] - 12s 138ms/step - loss: 0.0299
Epoch 2/30
38/38 [==============================] - 5s 138ms/step - loss: 0.0080
Epoch 3/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0064
Epoch 4/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0066
Epoch 5/30
38/38 [==============================] - 5s 137ms/step - loss: 0.0064
Epoch 6/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0055
Epoch 7/30
38/38 [==============================] - 5s 138ms/step - loss: 0.0054
Epoch 8/30
38/38 [==============================] - 5s 134ms/step - loss: 0.0060
Epoch 9/30
38/38 [==============================] - 5s 134ms/step - loss: 0.0054
Epoch 10/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0058
Epoch 11/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0052
Epoch 12/30
38/38 [==============================] - 5s 137ms/step - loss: 0.0048
Epoch 13/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0052
Epoch 14/30
38/38 [==============================] - 5s 134ms/step - loss: 0.0045
Epoch 15/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0044
```



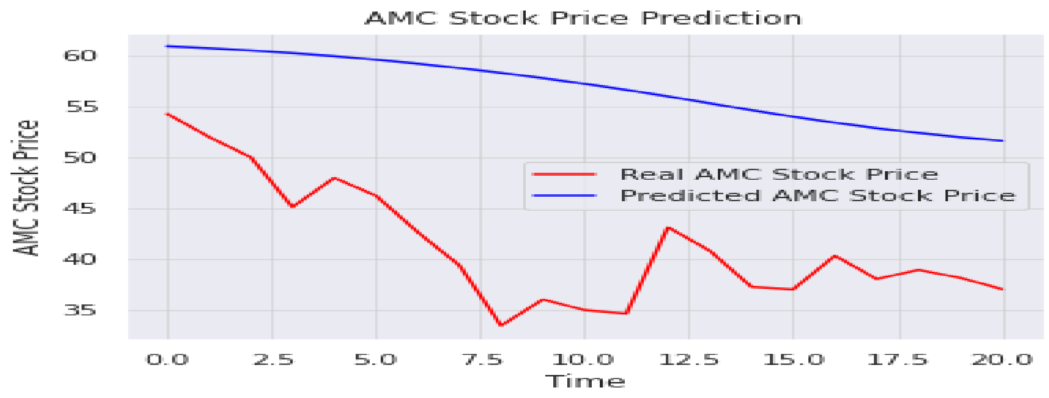
```
Epoch 16/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0045
Epoch 17/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0047
Epoch 18/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0039
Epoch 19/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0039
Epoch 20/30
38/38 [==============================] - 5s 137ms/step - loss: 0.0039
Epoch 21/30
38/38 [==============================] - 5s 134ms/step - loss: 0.0037
Epoch 22/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0036
Epoch 23/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0040
Epoch 24/30
38/38 [==============================] - 5s 138ms/step - loss: 0.0035
Epoch 25/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0034
Epoch 26/30
38/38 [==============================] - 5s 138ms/step - loss: 0.0041
Epoch 27/30
38/38 [==============================] - 5s 135ms/step - loss: 0.0040
Epoch 28/30
38/38 [==============================] - 5s 137ms/step - loss: 0.0030
Epoch 29/30
38/38 [==============================] - 5s 136ms/step - loss: 0.0032
Epoch 30/30
38/38 [==============================] - 5s 137ms/step - loss: 0.0031
```





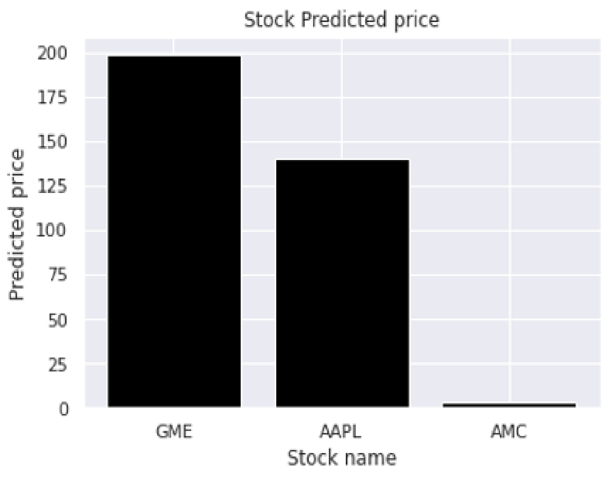
Recurrent Neural Network (RNN) for price prediction using sentiment analysis:

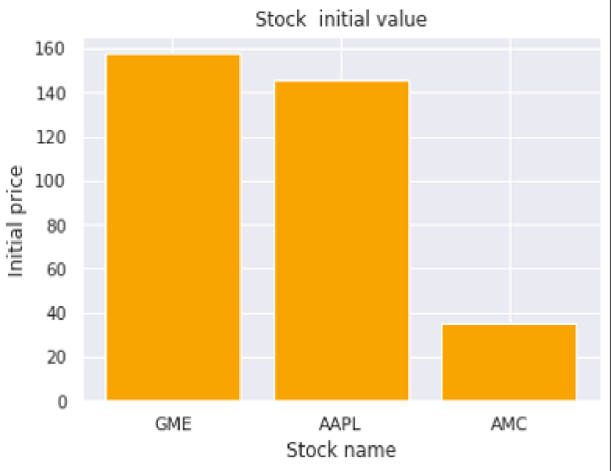
Due to the limit of 200 requests per minute on push shift api, we were not able to take bigger time frames into factor which would have made the results significantly better.

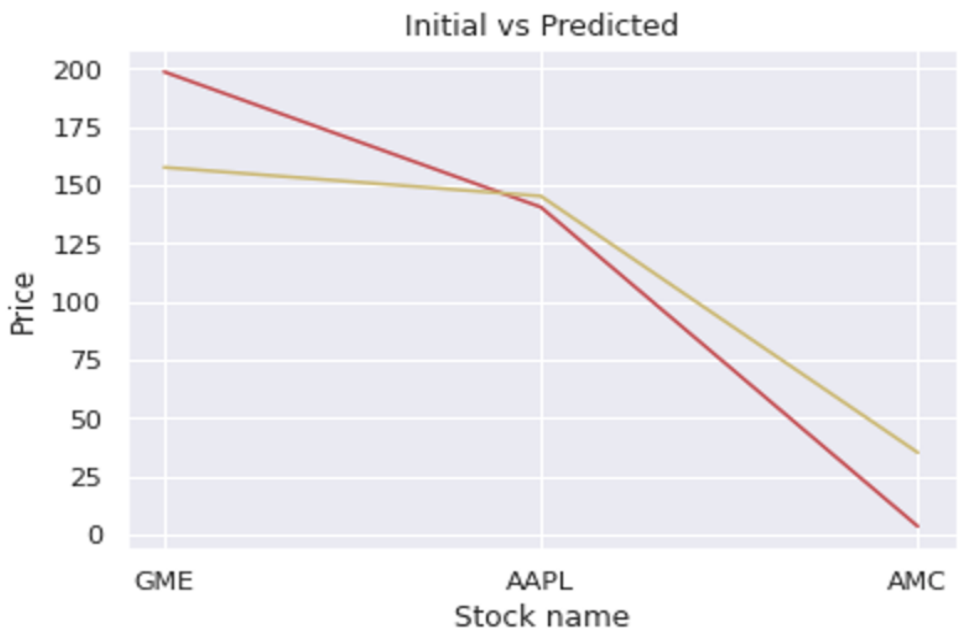


Prediction on Linear Regression model based on count:

Getting the required data for prediction was equally long work as we had to go through the same procedure of extracting the data but in a different way. Our model was trained for 365 days where an increased percentage was calculated till the 364th day but the final price was calculated for the 365th day. We did this because we were going to predict data based on the increase percent before and we trained our model in this way resulting in our prediction calculated for 6 +1 days where that extra plus 1 is the prediction for that day. This way our machine learning model was making good predictions for the data. These predictions change every day because our model is based on real time rates and recent comments of the people. Below is an example of a sample of Predicted price of 3 stock tickers and their initial price.







Above diagram shows the **initial vs predicted price** of this model.

Score for Linear Regression Model:

Our data was split using a test train split to test the validity score of our data. We split out data into 80-20, 80% of which was training data on which our model was trained upon and 20% of which was validation data and our model scored 0.7009005856676388

Disclaimer: Score could vary everyday because our model is based upon real time prices and posts. Plots could also be different every time the code is run for statistical analysis.

Models Which we tried but didn't work Out:

Polynomial Transformer: We tried using polynomial transformer for Linear Regressor model but it didn't work out because it was giving false values which were too far from true, and it provided a model score of 0.634 of our model so we decided to go with Linear Regression as was suggested by the professor.

Limitations

Project Limitations: Although the project was very interesting, there were some limitations:

1. Stock analysis using RNN did not work out as expected due to its nature of requiring huge amounts of data and the sudden volatility in hyped stocks never experienced before. The RNN shows much better results when tested on stock such as Tesla with a price history of 5 years (uncomment tickersymbol = 'TSLA' to see those results) as it has been in a hype for 5 years now.
2. Various API limits are set, including number of calls per minute, the rate at which data is obtained, and optional content available only after subscription.

Time/Resource limitations:

1. We were highly dependent on PushShift API to gather the reddit comments regarding the stock market. PushShift API has a drawback that we can make 200 requests per minute. Considering the huge data requirement of this project, the 200 requests per minute were not enough.
2. Unavailability of hourly data for stock price which could have made our RNN more robust and reliable.

Knowledge limitations:

1. We learned about RNNs while researching stock price prediction. Since we had no knowledge of RNNs, we had to conduct research by reading articles, books, and watching videos. A little prior knowledge of RNN's would have resulted in more accurate results.

Project Experience Summary

Jyotiraditya Mayor:

- Learned about the hyped stocks and what can be done to predict the next GME.
- Improved the data quality and project's reliability by integrating real time API data instead of kaggle files.
- Learned about technical analysis and how stock prices are affected. Implemented machine learning models using technical analysis to predict stock market prices and also recommend buy or sell ratings.
- Implemented deep learning (RNN) models to predict stock market prices.

- Managed the workflows for the project and coordinated with team members for proper execution of the project.
- Helped the team to gain knowledge about the stock market.

Ritika Goyal:

- Integrated real-time API data instead of Kaggle files to improve data quality and project reliability
- Used ETL approach that is Extract, Transform, Load to acquire, filter, clean, and analyse the data.
- Worked on creating classification models used for analysis
- The effects of technical analysis on stock prices were observed. A machine learning model was implemented using technical analysis to predict stock market prices and implemented buy or sell ratings.
- Implemented deep learning (RNN) models to predict stock market prices.
- Made various plots to visualize the data and obtained predictions.
- Provided code assistance to other members of my team.

Navjot Kaur:

- Collected datasets from Kaggle and cleaned the reddit dataset to get the desired stocks.
- Implemented various API's for sentiment and technical analysis.
- Worked and learned about statistical and machine learning models (RNN) by reading articles and helped in making required plots for the models.
- Used Random Regressor to predict the stock prices using technical analysis.
- Helped other teammates with their code whenever needed.

Dhairya Kalra:

- Learned how high stock volume and people's hype about a stock can actually drive the price of stock up and down by performing statistical tests.
- Worked on a linear regression model to predict the price of stocks based upon count and volatility.
- Used Yahoo finance and reddit Api to get real time results.
- Provided code assistance to clean and analyze data.
- Helped the team to gain knowledge about the stock market.