ROUGH SUMMARY OF THE PROJECT

This project was done with the aim to investigate the following :

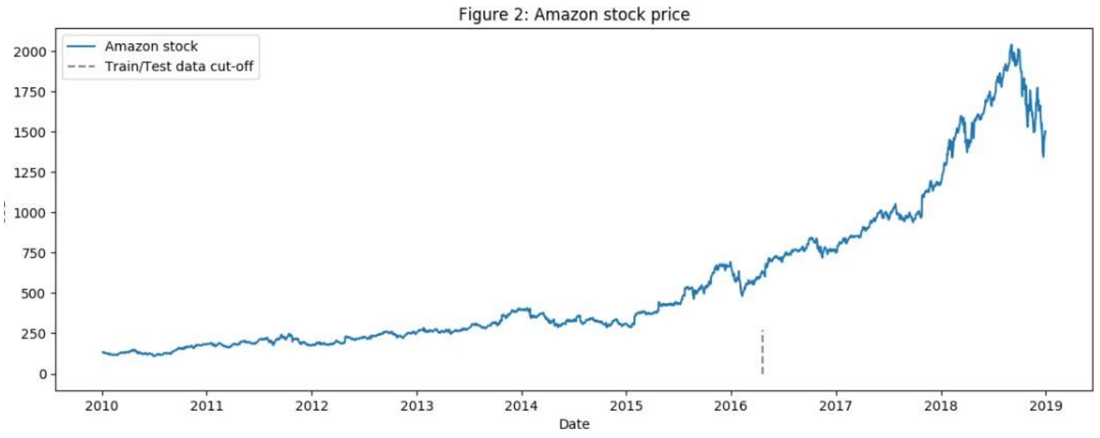
How many different techniques can we use to be able to predict prices of stock? Which techniques are the best? Is it better to try to predict Stock prices or Stock price movement and use trading strategies? Can we tell when we can buy or sell the stock? What metrics can we use for different models?

WORK DONE AND OBSERVATIONS:

Predicted different models of the time series OHLCV data, performed feature extraction, hyper parameter tuning and trained on ARIMA, Fourier, LSTM, LSTM with sentimental analysis and GAN models. Then focused on stock price movement instead of stock prices because it was found that it's more accurate to predict them.

Took historical Amazon data from Yahoo.Api and then performed feature generation on it, ARIMA model, Fourier model. Then performed LSTM on said model.

Put up the data in time series form and split between train and test seen below.



Amazon data peaks around 2015 and after. Most of this data is in the testing set and training set does not have peak value data.

Surmised that this could be a problem while dealing with conventional LSTM.

FEATURE GENERATION:

Following technical indicators were generated other than OHLCV data:

Bollinger bands: Bollinger Bands is used to define the prevailing high and low prices in a market to characterize the trading band of a financial instrument or commodity. Bollinger Bands are a volatility indicator. Bands consists of Moving Average (MA) line, a upper band and lower band. The upper and lower bands are simply MA adding and subtracting standard deviation.

2. EMA: Exponential moving average is a better version of a simple moving average that doesnt have SMAs lag. Moving averages just average out the data for a given time so we know how the company's closing price are trending for a given number of days. example for 4 days is price was 22,23 ,45,1.

(The company crashed on 4th day) the average would be 23. Now 23 is a below average value so it gives us an idea that 45 was indeed just a fluke and that in fact the company was always making losses

EMA is calculated as:

EMA(t)EMA(t0)=(1−α)EMA(t−1)+α p(t)=p(t0)

where α=1L+1 and length of window is α=2M

I used the ewm(exponential weighted mean) function to calculate ema.

3. Momentum: Momentum is perhaps the simplest and easiest oscillator (financial analysis tool) to understand and use. It is the measurement of the speed or velocity of price changes, or the rate of change in price movement for a particular asset.

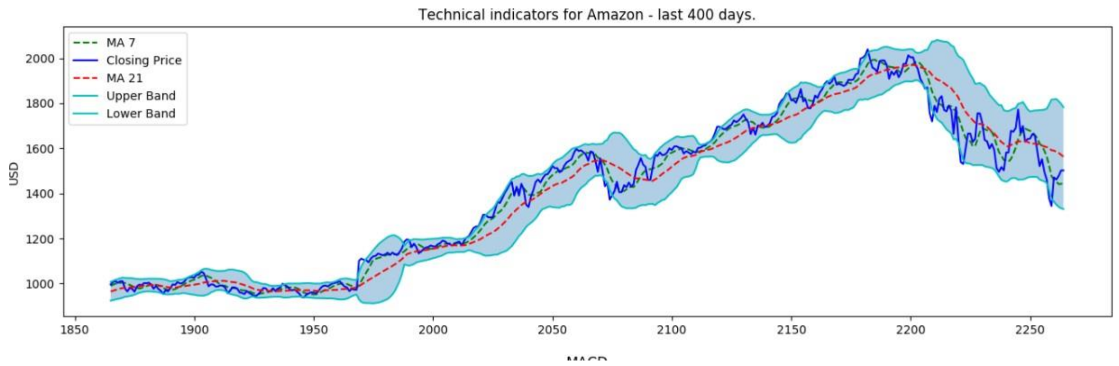
The formula for momentum is: Momentum=V−Vx where:

V=Latest price

Vx=Closing price

x=Number of days ago

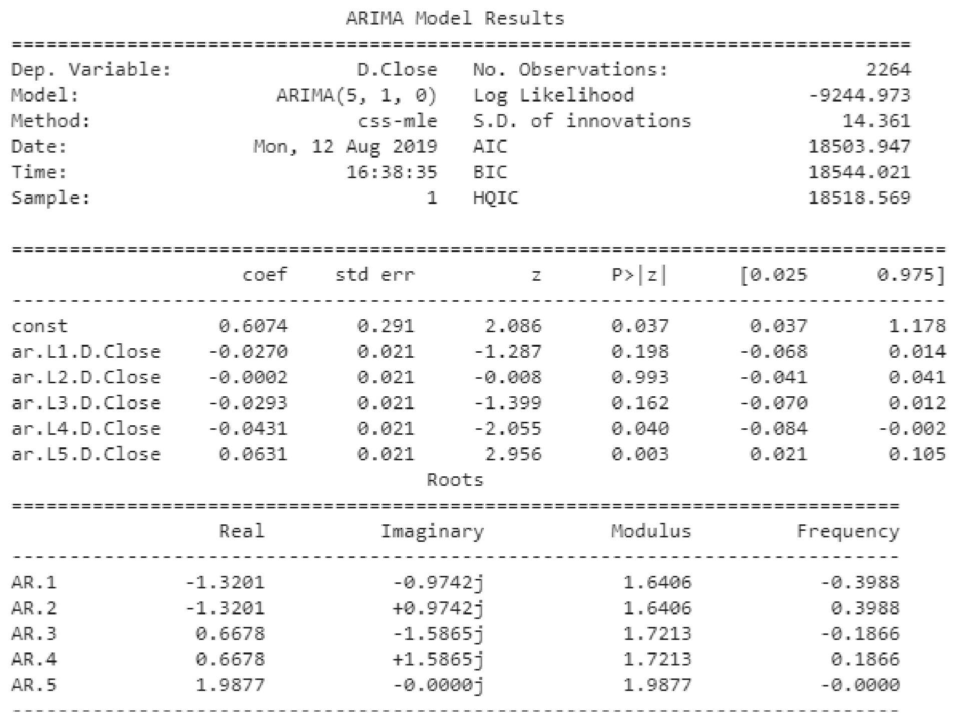
while generating them, other features that got generated were: 20SD(Standard Deviation of 20days) , Upper Band, Lower band, moving average of 7 days and 21 days and exponential moving average of 26 and 12 days.



Plot of technical indictaors over days where they started peaking. They also peak around the days where Amazon saw a huge growth.

Then generated ARIMA and Fourier models and decided to see if they can be used as features.

ARIMA:



```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                D.Close   No. Observations:                 2264
Model:                 ARIMA(5, 1, 0)   Log Likelihood               -9244.973
Method:                       css-mle   S.D. of innovations             14.361
Date:                Mon, 12 Aug 2019   AIC                          18503.947
Time:                        16:38:35   BIC                          18544.021
Sample:                             1   HQIC                         18518.569

=================================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
---------------------------------------------------------------------------------
const             0.6074      0.291      2.086      0.037       0.037       1.178
ar.L1.D.Close    -0.0270      0.021     -1.287      0.198      -0.068       0.014
ar.L2.D.Close    -0.0002      0.021     -0.008      0.993      -0.041       0.041
ar.L3.D.Close    -0.0293      0.021     -1.399      0.162      -0.070       0.012
ar.L4.D.Close    -0.0431      0.021     -2.055      0.040      -0.084      -0.002
ar.L5.D.Close     0.0631      0.021      2.956      0.003       0.021       0.105
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -1.3201           -0.9742j            1.6406           -0.3988
AR.2           -1.3201           +0.9742j            1.6406            0.3988
AR.3            0.6678           -1.5865j            1.7213           -0.1866
AR.4            0.6678           +1.5865j            1.7213            0.1866
AR.5            1.9877           -0.0000j            1.9877           -0.0000
-----------------------------------------------------------------------------
```

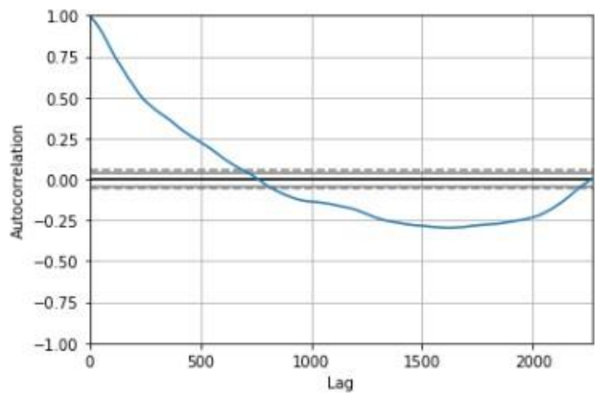
- **SUMMARY OF THE ARIMA MODEL**

1. A good starting point for the AR parameter of the model may be 5 which we did.

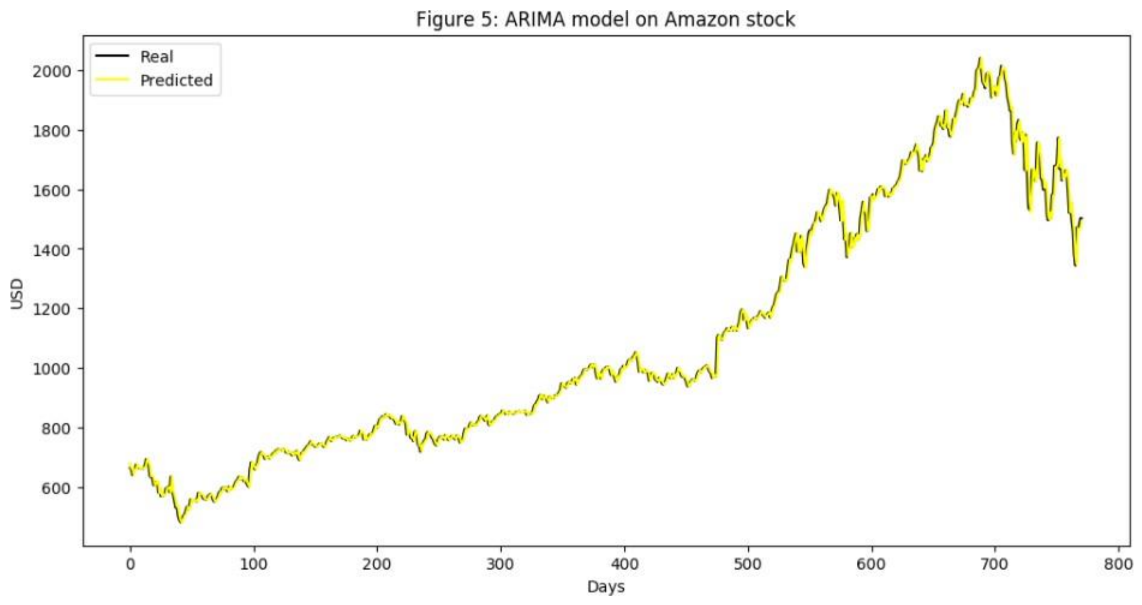
2. From the summary of the ARIMA we can see that most P-values are greater than 0.05 other than the last two. The model should be great!

3. The difference between AIC and BIC is low so this indicates this is a good model.

4. Running the example, we can see that there is a positive correlation with the first 0-to-500 lags that is perhaps significant for the first 250 lags in the autocorrelation below.

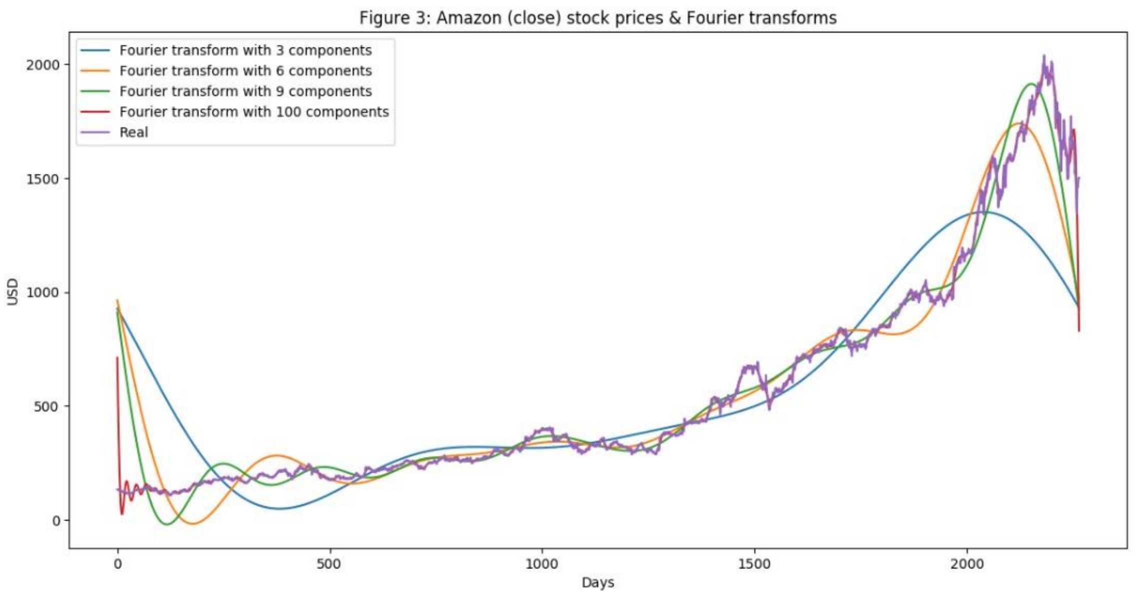


ARIMA prediction plot is pretty good:



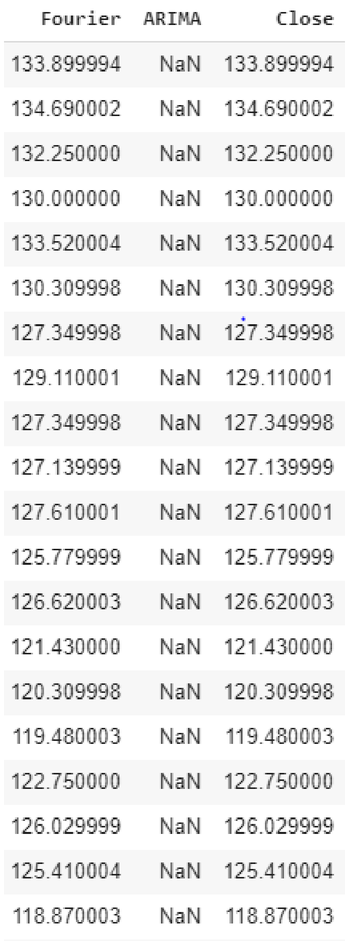
FOURIER MODEL:

Use Fourier Transform in the spectral domain and reconvert it into time domain and plot with multiple components. The component which is closest to real values can be plotted.



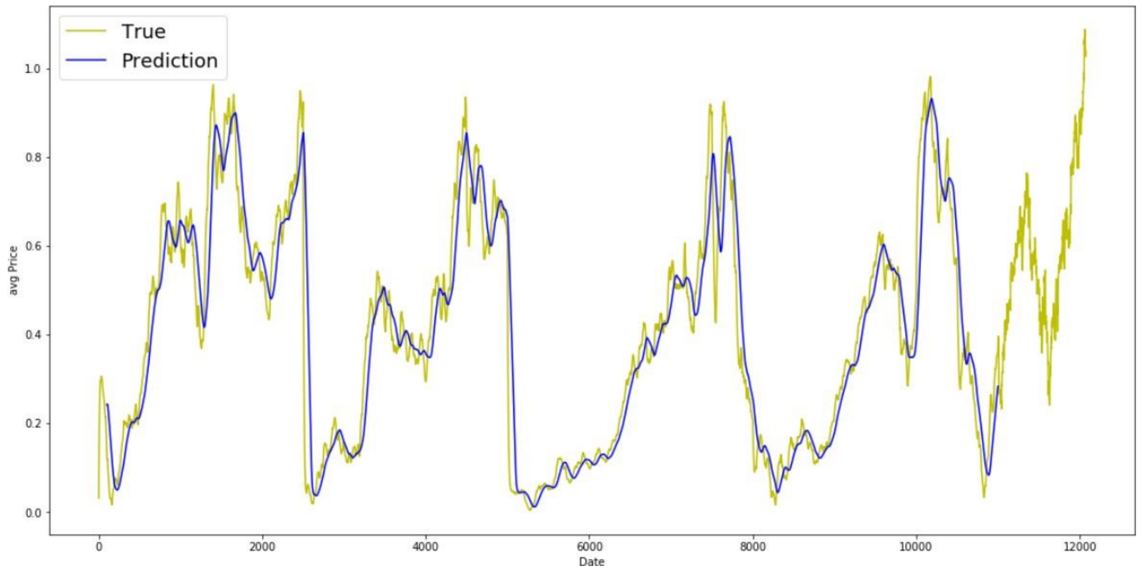
In our case it is 100 components.

Normalize the values i.e., do not keep spectral component values and generate the prediction data. Fourier gives results very close to Closing price data as seen in plot:



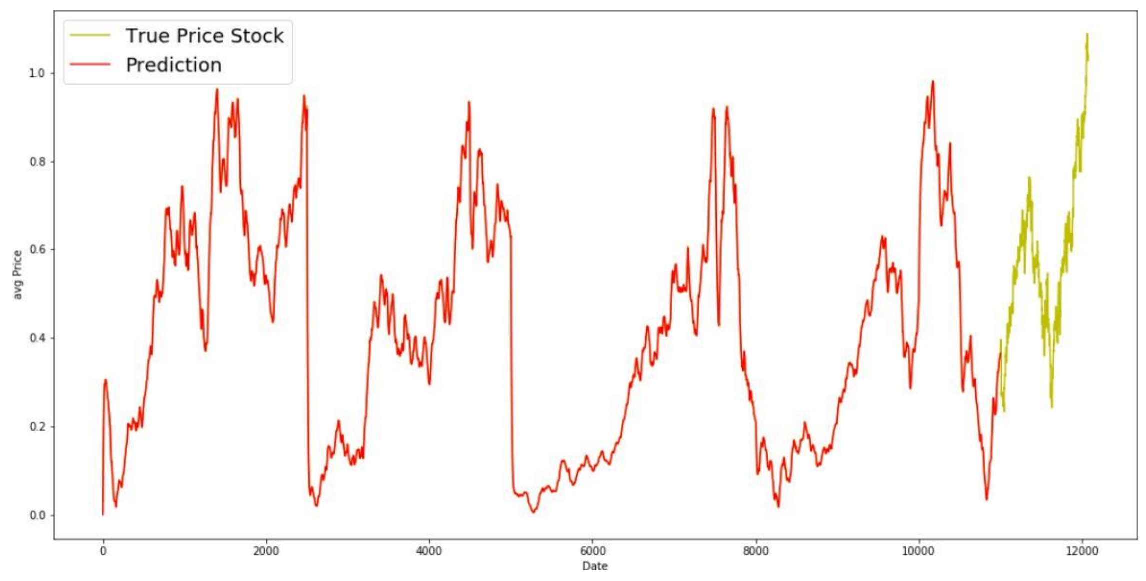
Fourier	ARIMA	Close
133.899994	NaN	133.899994
134.690002	NaN	134.690002
132.250000	NaN	132.250000
130.000000	NaN	130.000000
133.520004	NaN	133.520004
130.309998	NaN	130.309998
127.349998	NaN	127.349998
129.110001	NaN	129.110001
127.349998	NaN	127.349998
127.139999	NaN	127.139999
127.610001	NaN	127.610001
125.779999	NaN	125.779999
126.620003	NaN	126.620003
121.430000	NaN	121.430000
120.309998	NaN	120.309998
119.480003	NaN	119.480003
122.750000	NaN	122.750000
126.029999	NaN	126.029999
125.410004	NaN	125.410004
118.870003	NaN	118.870003

SIMPLE MOVING AVERAGE:



We used the simple moving average by creating a lookback window and then ran it on the data. Thus we were able to get a good model just as expected from SMA.

EXPONENTIAL MOVING AVERAGE:



EMA is a great model for this dataset. Ideally the pattern of the True data should have been followed in the prediction model. Coded from range 1 to N-1 and put all the averaged values in the running mean. Used dense as 0.5 and then multiply it with the running average.

It predicts the predicted values after performing EMA on the dataset formula of which has been given above.

FEATURE IMPORTANCE USING XGBOOST:

Using XGBoost we found which features would make be the best for prediction. These are plotted below.

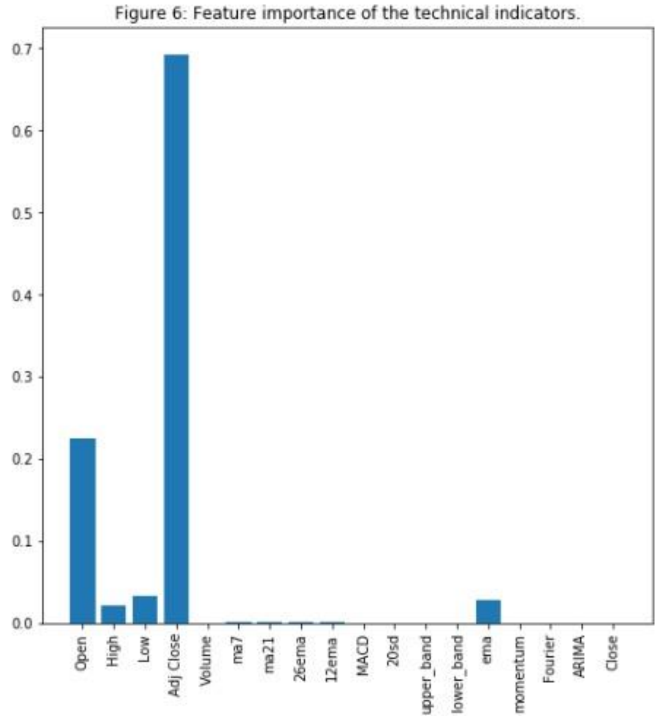


Figure 6: Feature importance of the technical indicators.

As seen above, Open Adj Close, EMA and high and Low are great indicators. Others are ma7, ma21 and 12ema.

LSTM model to predict stock prices using 1 feature.

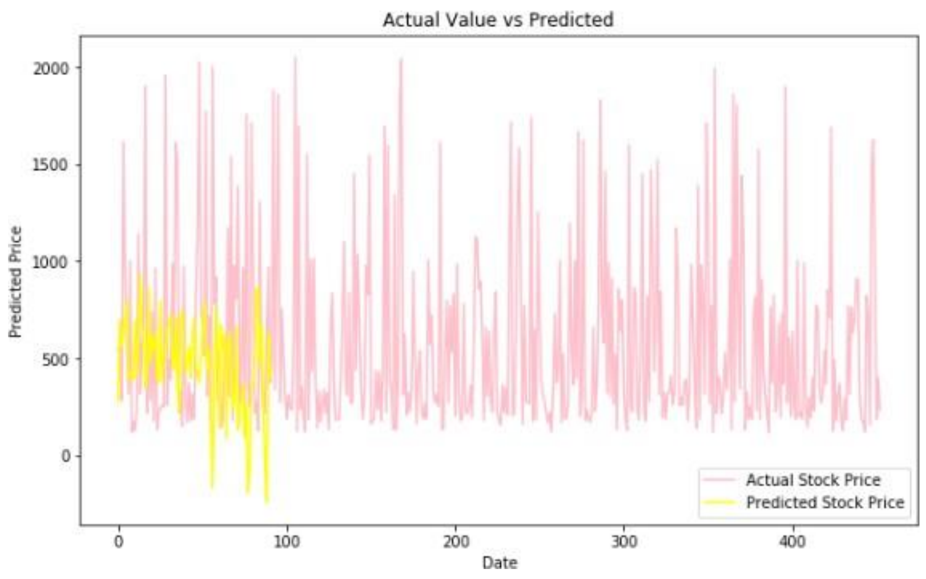
Ran Open training data for 100 epochs and tried to predict Open with it. This is more like a regression problem so the metrics used were MSE and mean absolute error (not accuracy). The results weren't all that great since there was some overfitting and normalization of the data. We also did not perform hyperparameter tuning.

MAE was: 0.167

This means the average difference between input and output for all 2300 datapoints is 0.167.

However the value is for the days here so the MAE here is pretty bad. (2350 length of dataset will be denominator. Difference between actual and prediced values should be so small that such a large denominator dividing the difference should put MAE in rage of 10^-3 i.e. 0.00then digits. Since MAE is 167.something*10^-3(0.167) difference is high.

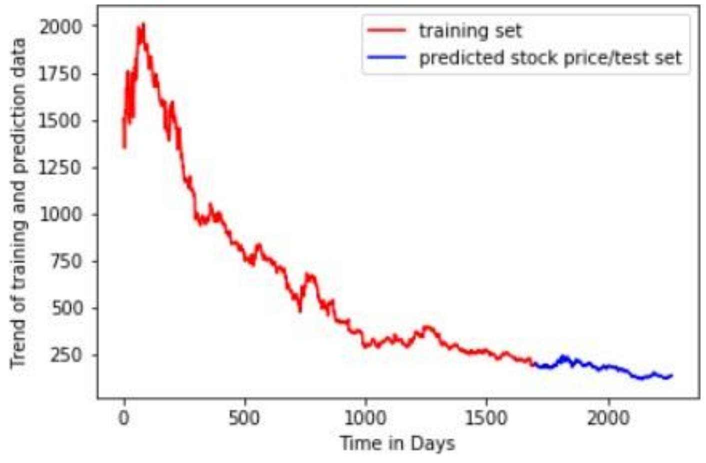
The prediction plot looked like this:



Not great.

LSTM MODEL USING MULTIPLE FEATURES:

So we tried more features (5 features) and tried to predict closing prices with them. We encoded and normalized the 5 features and the prediction plot looked something like the one below. Again, this was a regression problem so the metrics used where MSE, MAE and not accuracy.



MAE here was 0.016 during training and the plot continues the trend of the training data but not satisfactorily.

MAE calculation is sum(Y-X)/total data points = (2000-1980)/2350

here 2000 is the day with highest closing price. 1980 is the day with highest closing price in training set and 2350 will be the length of the data,

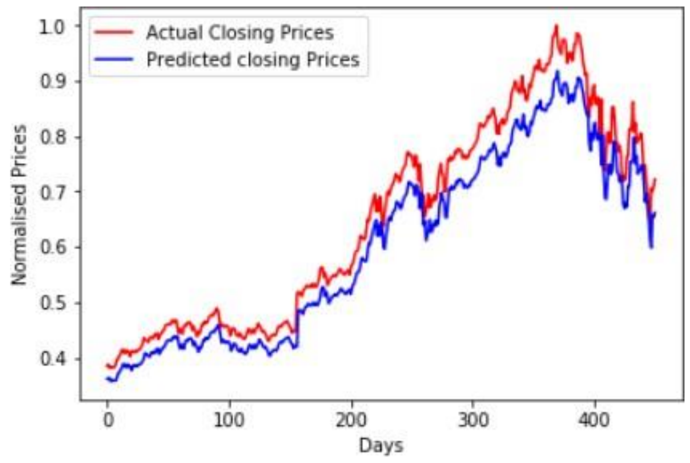
The mistakes with the 1st LSTM models were:

- Encoding and normalization lost a lot of precious data. We have only 2000 days of data and 1500 days of training data where every data point matters. So, no extensive encoding and normalization.
- Tried to predict stock prices as a regression problem but did not do any hyperparameter tuning. Adding linearity in a neural network will produce flat results.
- Instead of feeding direct stock prices we should feed in stock price movements by creating a window that keeps all similar stock price values in one window and feeds in each window asa datapoint.

LSTM PREDICTING STOCK PRICE MOVEMENT

Created a next 'Vanila' LSTM model to predict only Closing stock prices and got a good prediction model:

- Ran a window over close data to convert closing prices into closing price movements.
- Normalized the data and split into train test and validation.
- Ran the model and saw to that it does not overfit.
- Got a MAE of 0.0076 lowest yet in any model
- Plotted a good prediction model:



Next LSTM models will focus on hyperparameter tuning.

Since AMAZON dataset has only 2000 days 1500 days of which are kept for training and 500 for testing in which 500 testing days are the days where Amazon saw its exponential growth, thus the models especially for LSTM will not show further improvement than this.

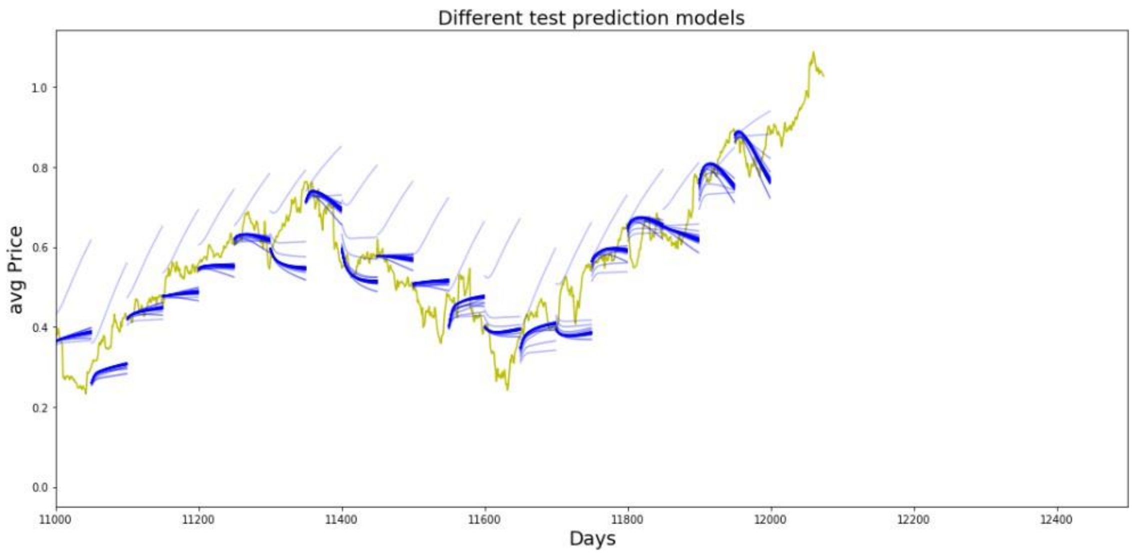
So, I decided to move to a GE dataset that spans from 1970 to 2010s and gives more data to play around with.

- Cleaned up the data.

- Normalized, ran a window to get price movements and split into training and testing datasets.

- Performed hyperparameter tuning

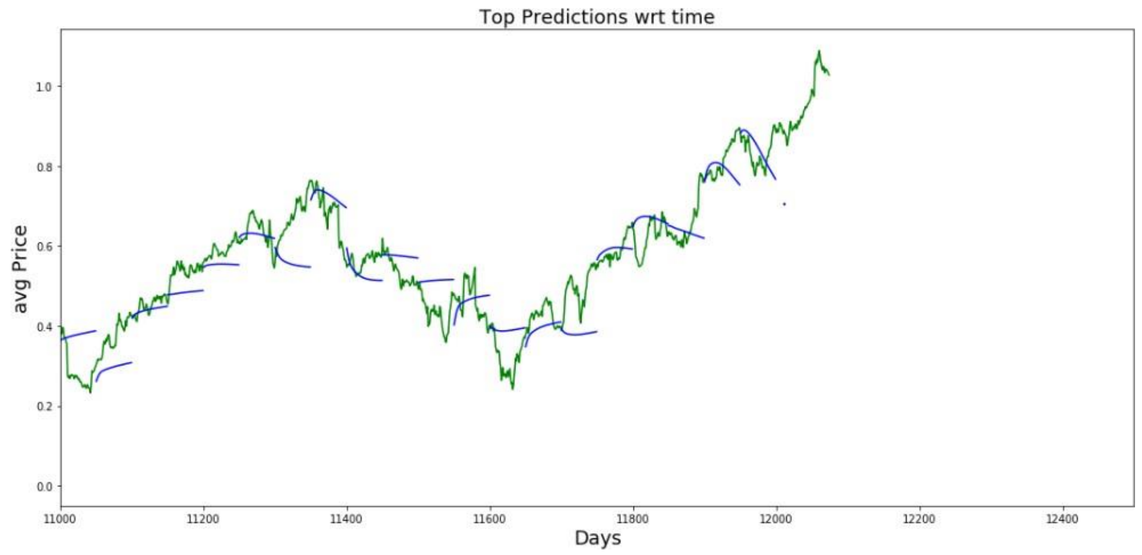
- Plot:



The blue ticks indicate when the price movement changes. As you can see it's mostly making the right prediction for the price movement.

If you are a trader this gives a good idea when to short your stock or invest more.

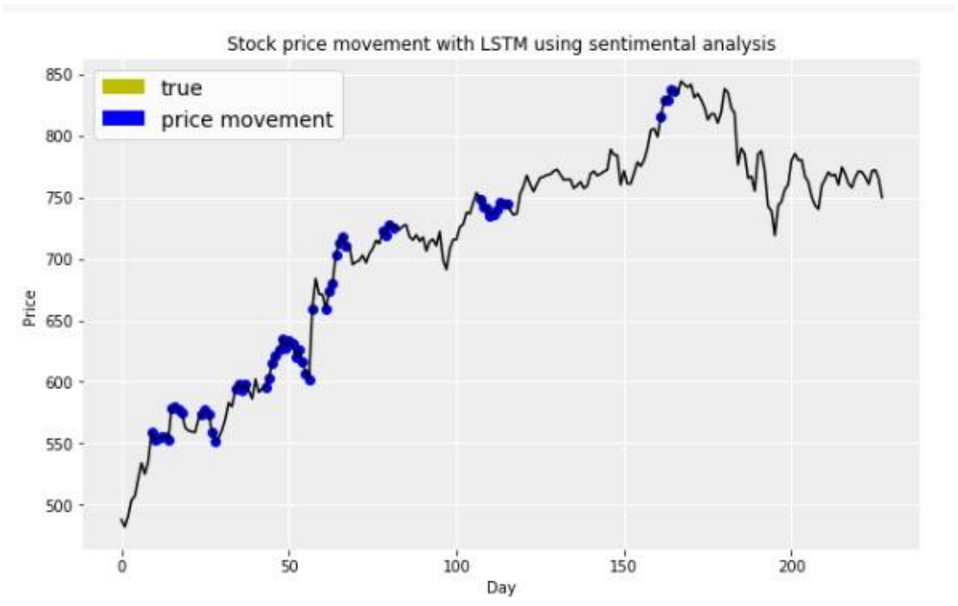
The next plot gives a less confusing version:



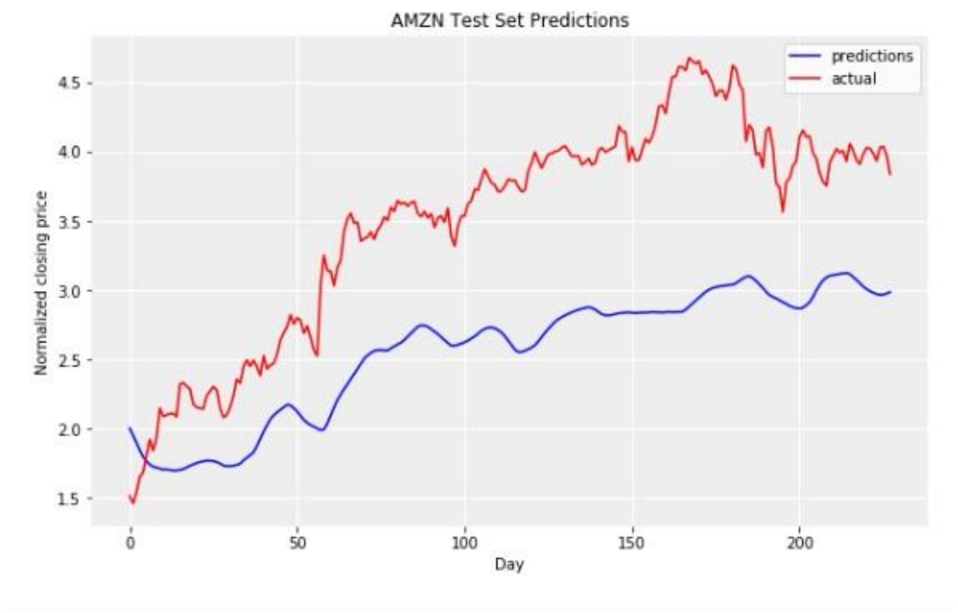
LSTM Sentimental Analysis models:

There are tons of papers talking about LSTM and Twitter sentiments. We went with Reddit sentiments, news and Amazon data and drew in polarity between them and stock prices.

Using this I ran a LSTM model which used the polarity of the sentiment and prices (run a code to encode both of them) and then I predicted the test outputs.



After hyperparameter tuning we got this model. The blue dots indicate the areas where price movement is present and where you can buy/sell the stock. (Sell when lower and buy when higher).



The prediction vs Actual plot. The predicted price follows the same structure as true but with less magnitude.

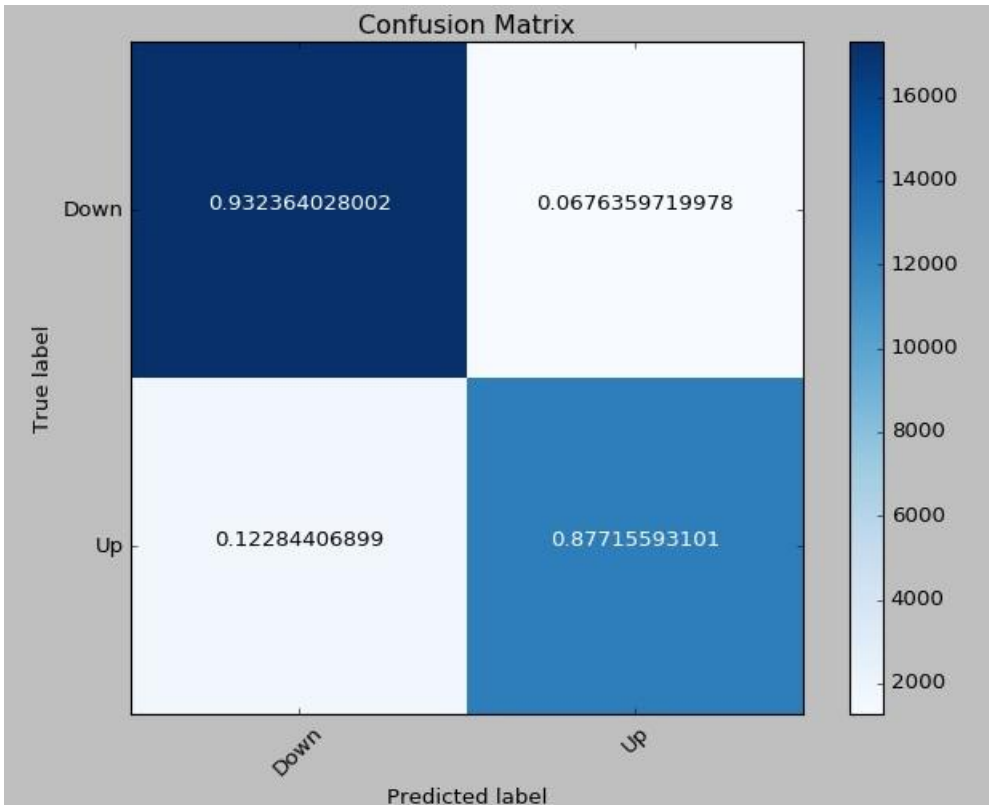
All in all, the sentimental analysis gave a great model for the limited Amazon dataset and it could be inferred about when to buy or sell stock.

GAN MODEL:

The GAN model does not work as the same regression problem.

We used the time series data as generator and a CNN as a discriminator.

Made a confusion matrix that says how close I am to detecting if the price is going up or down.

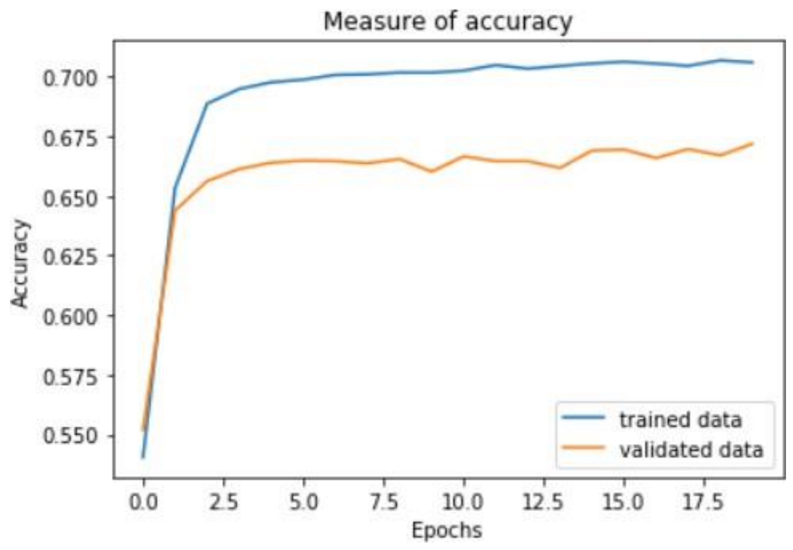


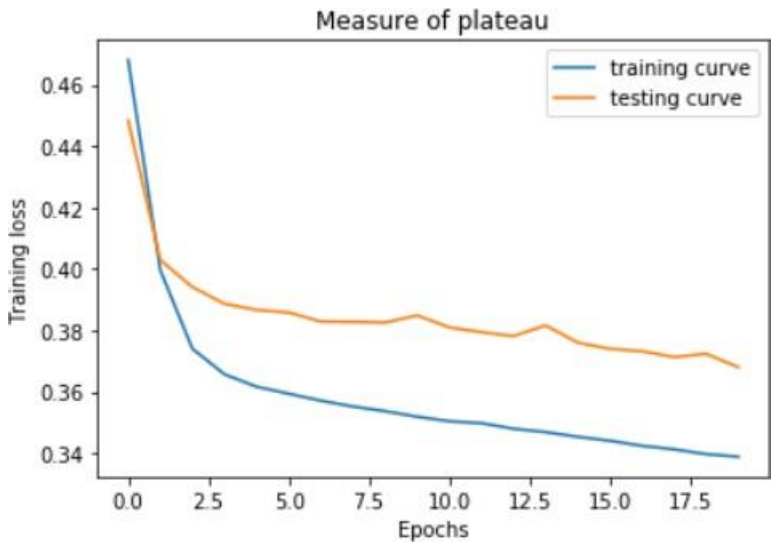
Chances of Down price being predicted as Down is 93% and Up ad Up is 87% which is quite good.

LSTM, GRU USING ACCURACY AS A METRIC:

- Used MAE as a metric of measure. However, I then changed to using accuracy as a measure by changing the regression problem into classification where I found how I could predict datapoints of the closing price by creating EMA for the given dataset.
- Then used this dataset to split into test, train and validate sets.
- Stacked the OHLCV data and used that as X and closing prices as Y hence the classification problem.
- Then predicted the accuracy.
- Used hyperparameter tuning for the LSTM model and ran a GRU cell with some hyperparameter too.

Results of Best LSTM model:





This model gave an accuracy of 68.9% i.e. it was able to predict the right closing prices 68.9% of the time.

References: All references are listed in notebooks for each code block.