2020

Stock Price Predictor

Machine Learning Engineer Nanodegree

Capstone Project

VIVEK PANDEY

Definition:

1. Project Overview:

A stock market is the aggregation of buyers and sellers of stocks, when people talks stocks, they are usually talking about companies listed on major stock exchanges like the New York Stock Exchange (NYSE) , the Nasdaq, the Bombay Stock Exchange (BSE) and many others where stockbrokers and traders can buy and sell shares of stock which represents ownership/share claims on businesses. When we buy a stock of a company and if that company does good then it's more likely that stock price goes up and we gain the same percentage of profit in the investment of that stock and vice versa . Stock market prediction has been one of the challenges for researchers and financial investors, over the years they have been trying to better understand stock price behavior and make profitable investments and trades. However, predicting stock market movements is extremely challenging because according to Zhong and Enke (2017), stock markets are affected by many highly interrelated factors that include economic, political, psychological, and company-specific variables.

Stock market price prediction is a tricky thing. Several theories regarding stock markets have been conceptualized over the years. One is Efficient Market Hypothesis (EMH) and another one is Random Walk Theory.

Efficient Market Hypothesis (EMH): It states that at any point of time, the market price of a stock incorporates all information about that stock. Hence it is not possible to outperform the stock market.

Random walk theory: Random walk theory assumes that it is impossible to predict stock prices as stock prices don't depend on past stock. It also considers that stock price has great fluctuations, so it is infeasible to predict future stock prices.

Many widely accepted empirical studies show that financial markets are to some extent predictable (Chong et al. 2017). Criticism of EMH has given rise to an increasing number of studies that question the validity of EMH and introduce new and successful approaches that combine technical analysis indicators and chart patterns with methodologies from econometrics, statistics, data mining, and artificial intelligence (Arévalo et al. 2017).

I have been doing stock trading for last couple of years and analyzing stock market behavior fascinates me a lot. Being an active stock trader motivates me to utilize all my technical and functional skills to build the stock market predictor, so that I can utilize this predictor for my stock trading. I believe it will be a perfect project to evaluate my final model with real time stock data which can help me to optimize this predictor over period.

2. Problem Statement:

The Market researchers and financial investors have been facing the problem of not being able to predict stock market price movement, hence they are struggling to make profitable investments and trades.

In this project, I will be building a stock price predictor that takes daily trading data for a list of ticker symbols (e.g. TSLA, FB) over a last 5 years span as input, and outputs the predicted stock prices for each of those stocks on the given dates.

3. Metrics:

I will use Root Mean Square Error (RMSE) as the evaluation metric which is very popular evaluation metrics for regression problems, it is a frequently used measure of the differences between values (sample or population values) predicted by a model or an estimator and the values observed.

I will compare RMSE value for Amazon SageMaker DeepAR forecasting algorithm against my benchmark model, also I will implement and validate Long short term memory (LSTM).

In this comparison whichever model will have lower RMSE value that will have the best accuracy among them, hence I will choose the model with lower RMSE as my best model.

Analysis:

4. Data Exploration:

For this project, I will use historical stock data of top 10 stocks holdings of ARK Next Generation Internet ETF (ARKW) ETF from Yahoo! Finance using rapidapi for last 5 years (Jan 10, 2015 - Aug 04, 2020).
I have chosen this very popular ETF because I have invested on this ETF and a lot of top Market analysts have been suggesting buying this, so I am curious to know more hidden insights about this ETF.

The ARK Next Generation Internet ETF is an actively-managed fund, therefore it does not track a particular index. ARKW aims to identify companies that will profit from developments in cloud computing, artificial intelligence (AI), financial technology, and similar innovations. Its largest holdings are Tesla Inc. (TSLA), the electric car company; Square Inc. (SQ), the mobile payments company; and Roku Inc. (ROKU), the digital streaming equipment maker.

The list of top 10 stocks of ARKW ETF that will use in this project.
1- Tesla Inc : (TSLA)
2- Square Inc A : (SQ)
3- Roku Inc Class A : (ROKU)
4- 2U Inc : (TWOU)
5- Zillow Group Inc C : (Z)

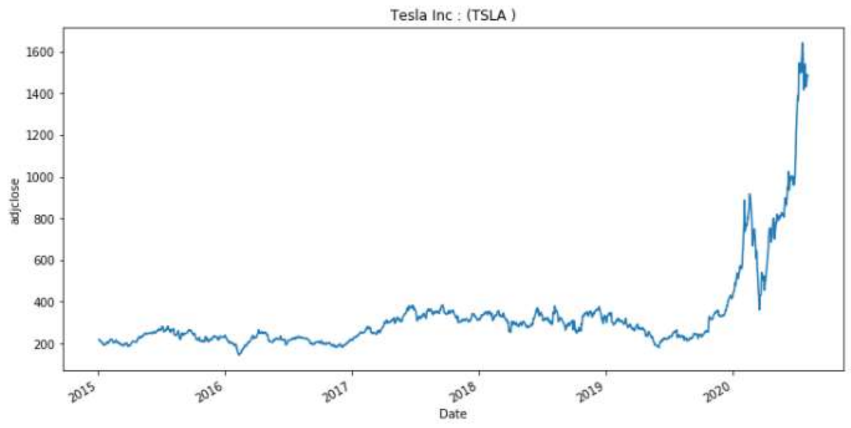
6- Splunk Inc : (SPLK)
7- LendingTree Inc : (TREE)
8- Pinterest Inc : (PINS)
9- Xilinx Inc : (XLNX)
10- Facebook Inc A : (FB)

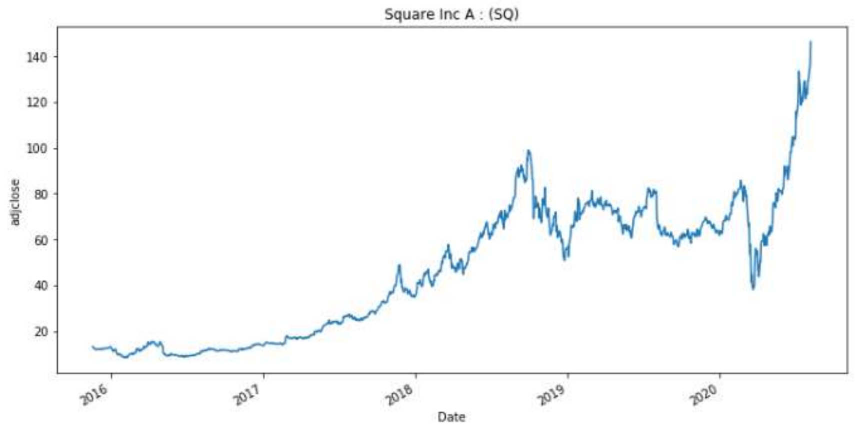
The historical data for last 5 years (Jan 10, 2015 - Aug 04, 2020) fetch from Yahoo finance has seven columns such as Date, Open (opening price), High (highest price the stock traded at), Low (lowest price the stock traded at), Close, Adjusted Close (closing price adjusted for stock splits and dividends) and Volume (how many stocks were traded).

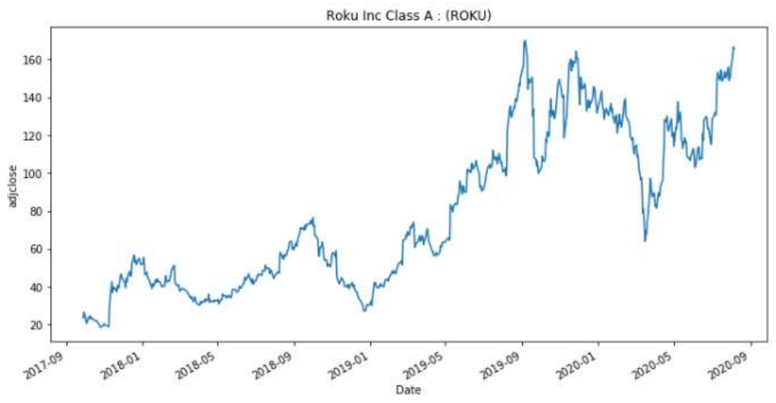
5. Exploratory Visualization:

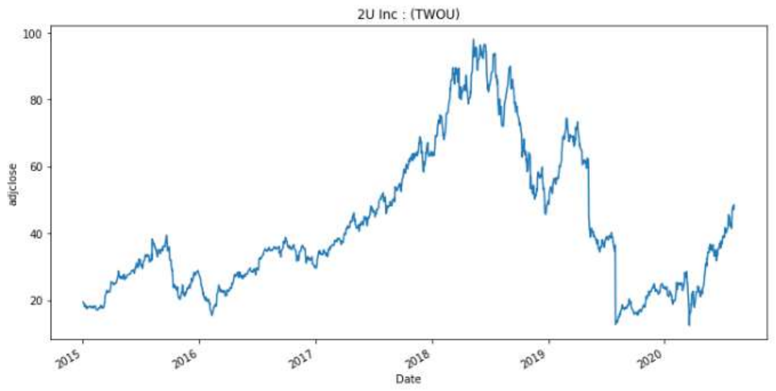
After loading the data and converting it to timeseries format I have visualized all 10 stocks and their patterns.

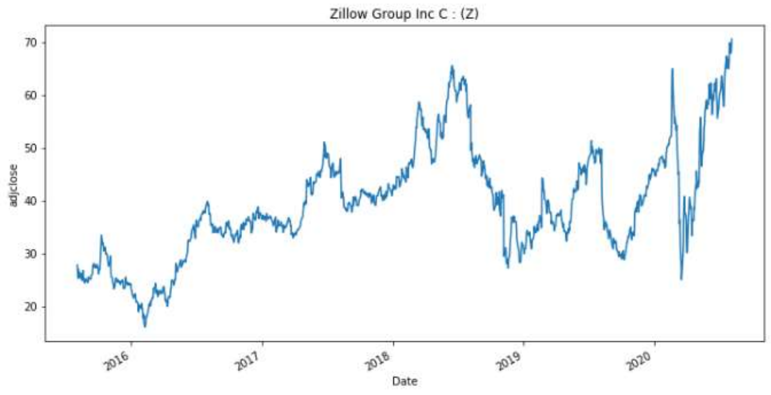
```
Title= ['Tesla Inc : (TSLA )' ,'Square Inc A : (SQ)','Roku Inc Class A : (ROKU)','2U Inc : (TWOU)','Zillow Group Inc C : (Z)','Splunk Inc
: (SPLK)', 'LendingTree Inc : (TREE)','Pinterest Inc : (PINS)','Xilinx Inc : (XLNX)','Facebook Inc A : (FB)']

for k in range(len(time_series)):
    plt.figure(figsize=(12,6))
    time_series[k].plot()
    plt.xlabel('Date')
    plt.ylabel('adjclose')
    plt.title(Title[k])
    plt.show()
```

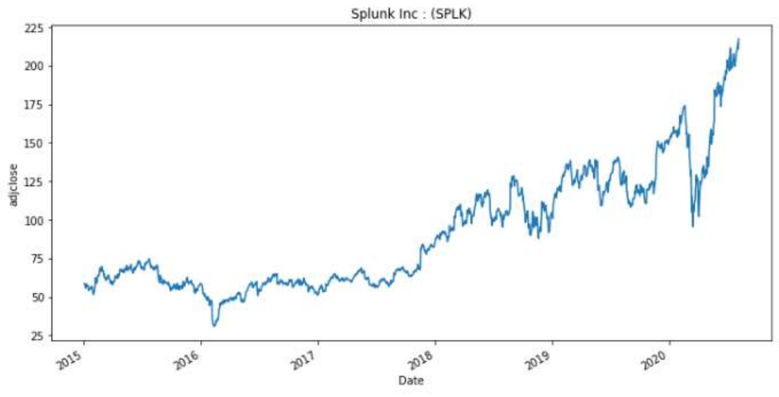


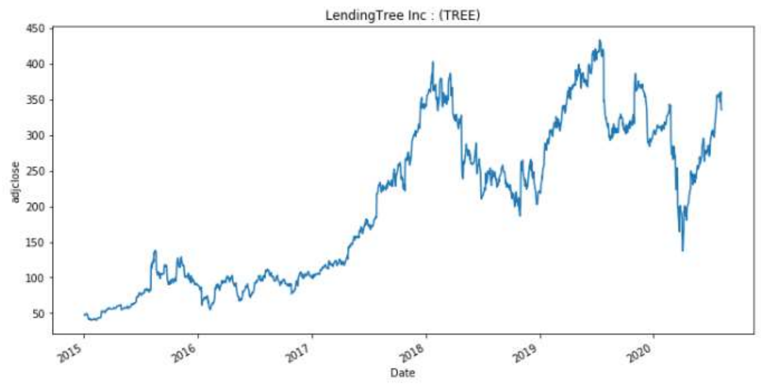


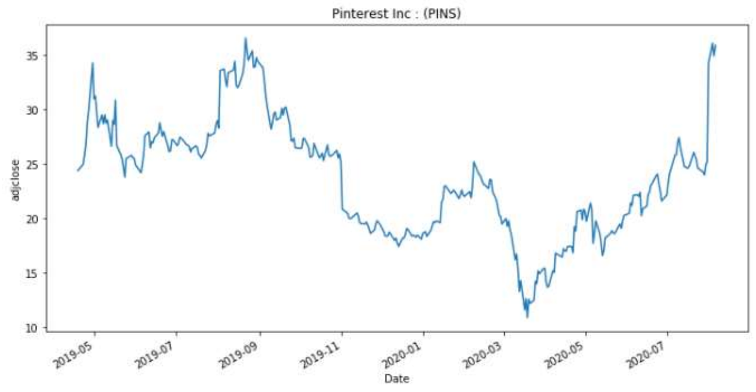

Roku Inc Class A : (ROKU)
adjclose
Date
20
40
60
80
100
120
140
160
2017-09
2018-01
2018-05
2018-09
2019-01
2019-05
2019-09
2020-01
2020-05
2020-09

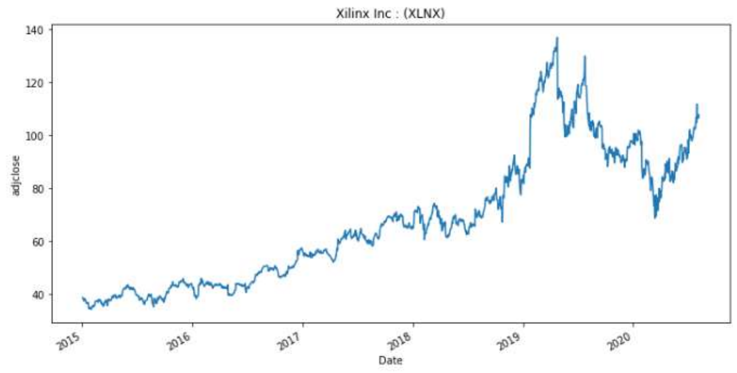

2U Inc : (TWOU)
adjclose
Date
20
40
60
80
100
2015
2016
2017
2018
2019
2020

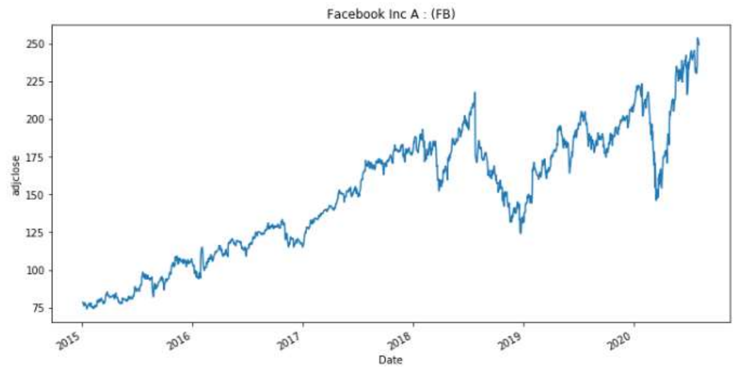

Zillow Group Inc C : (Z)
adjclose
Date
20
30
40
50
60
70
2016
2017
2018
2019
2020


Splunk Inc : (SPLK)
adjclose
Date
25
50
75
100
125
150
175
200
225
2015
2016
2017
2018
2019
2020









By analyzing the above graph of all 10 stocks it seems that all these stocks became very volatile in 2020 specially after March-2020, and the reason is due to pandemic stock market was crash very badly in March, however after that it started recovering and seen a significant rise in most of the stock prices.

6. Algorithms and Techniques:

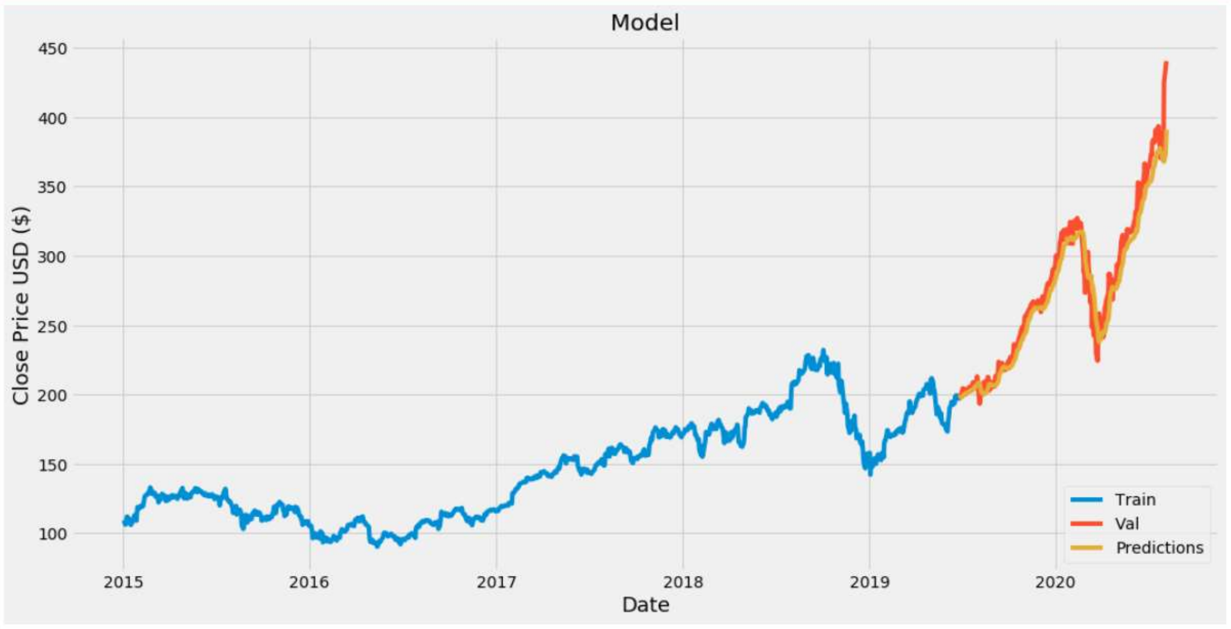
As this is a regression problem where we have to predict stock adjusted price, so I will be using regression algorithms such as Amazon SageMaker DeepAR forecasting algorithm and Long Short-Term Memory (LSTM) network.

As per existing study Amazon SageMaker DeepAR forecasting algorithm is very well suited for financial time series forecasting, so I will consider this algorithm as my first preference for this stock prediction, however I will use it after comparing it's performance against LSTM algorithms which is a type of recurrent neural network used in deep learning because very large architectures can be successfully trained.

First step will be to get data from Yahoo Finance and perform the require preprocessing into the data, then feed this input data to the model, and the model will return the forecast stock Adjusted Close value for next 30 days.

7. Benchmark:

As a Benchmark, I used Long Short-Term Memory (LSTM) a type of RNN.
I compared my final model's results with my benchmark model which is LSTM which is also very well known for time-series prediction. I used the Apple stock for this benchmark model.

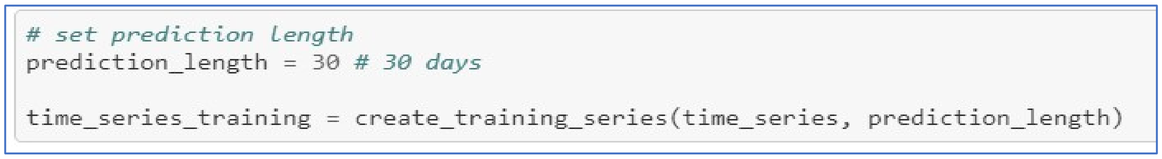


It seems LSTM is predicting data more accurately and RMSE value 12.75 is also significant less than the tuned deepAR model. Also from the above graph it looks like LSTM model is able to predict upward and downward trend very effectively, and stock predicted value is very close to actual stock value.

Methodology:

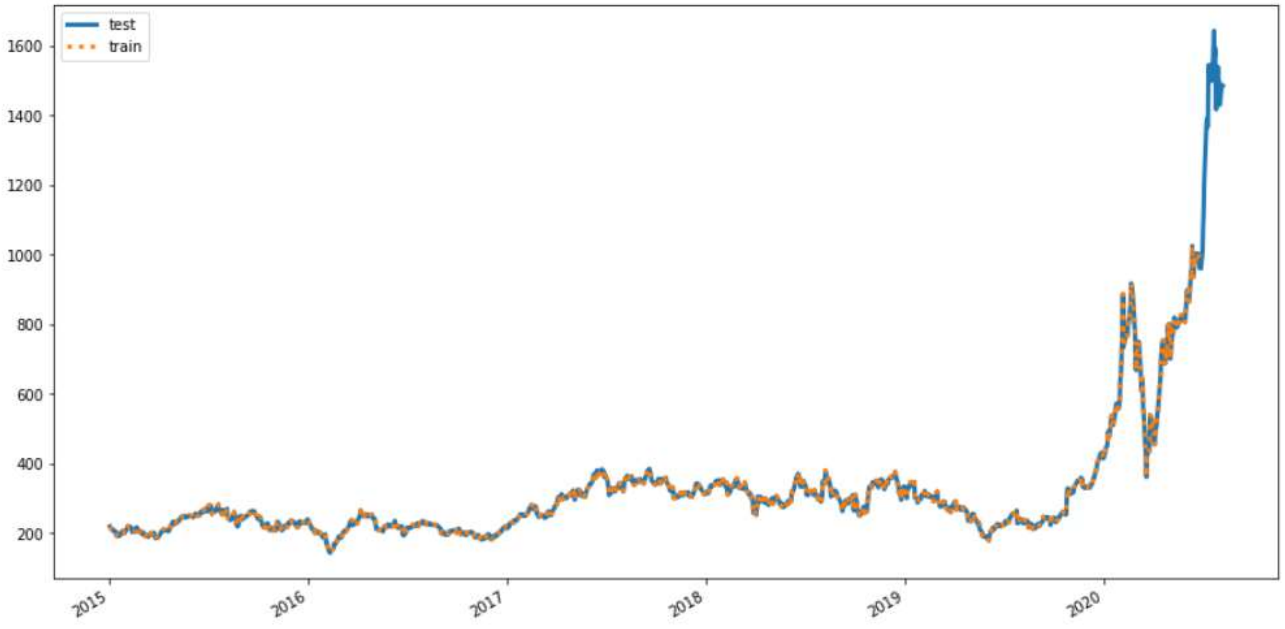
8. Data Preprocessing:

The data received from **rapidapi** api needed to convert to timeseries since DeepAR algorithm takes timeseries as input.
For this prediction I have considered only "adjclose" and "date" data for each timeseries (each stocks) hence extracted only these two fields for each timeseries.
For machine learning tasks like classification, we typically create train/test data by randomly splitting examples into different sets. For forecasting it's important to do this train/test split in time rather than by individual data points.
To create a training set of data, I have left out the last 30 points of each of the time series I had generated, and used only the first part as training data. The test set contains the complete range of each time series.



```
# set prediction length
prediction_length = 30 # 30 days

time_series_training = create_training_series(time_series, prediction_length)
```

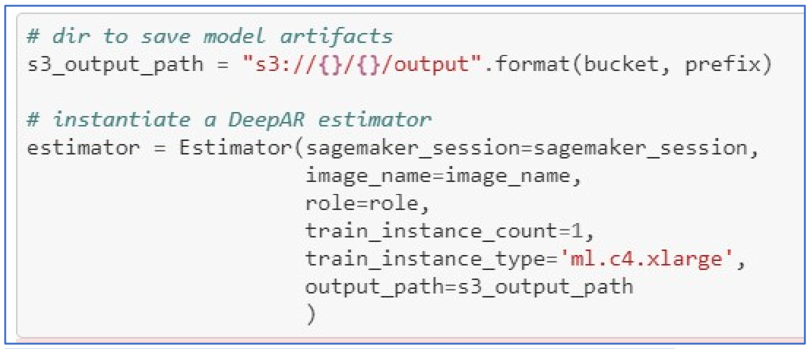
To visualize what these series look like, I have plotted the train/test series on the same axis. We can see that the test series contains all of our data in a year, and a training series contains all but the last prediction_length points.



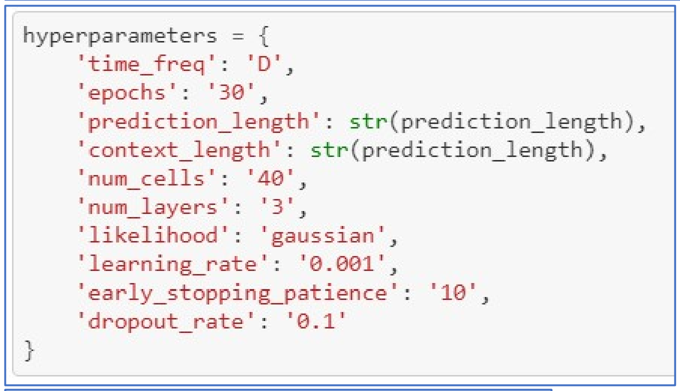
According to the DeepAR documentation, DeepAR expects to see input training data in a JSON format, hence converted timeseries into JSON format, after that saved these files locally and uploaded it to S3.

9. Implementation:

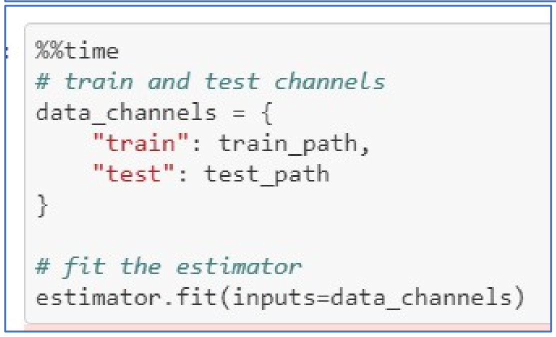
In order to implement this DeepAR model, first I have instantiated estimator ,set hyperparameters and launched the training job then SageMaker started an EC2 instance, downloaded the data from S3, started training the model and saved the trained model.



```
# dir to save model artifacts
s3_output_path = "s3://{}/{}/output".format(bucket, prefix)

# instantiate a DeepAR estimator
estimator = Estimator(sagemaker_session=sagemaker_session,
                      image_name=image_name,
                      role=role,
                      train_instance_count=1,
                      train_instance_type='ml.c4.xlarge',
                      output_path=s3_output_path
                      )
```



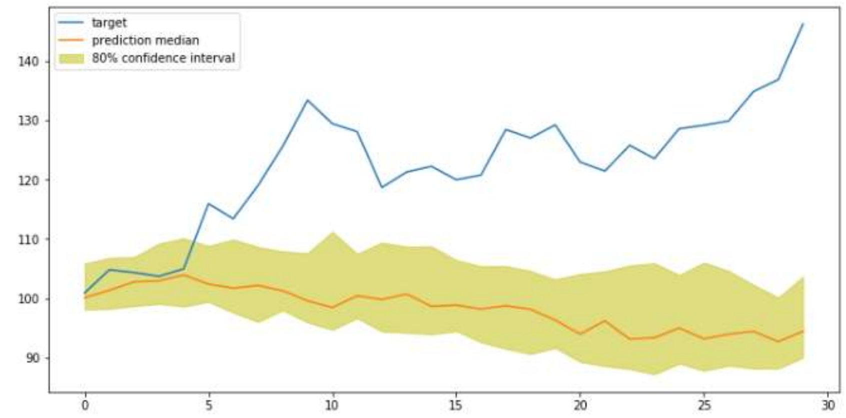
```
hyperparameters = {
    'time_freq': 'D',
    'epochs': '30',
    'prediction_length': str(prediction_length),
    'context_length': str(prediction_length),
    'num_cells': '40',
    'num_layers': '3',
    'likelihood': 'gaussian',
    'learning_rate': '0.001',
    'early_stopping_patience': '10',
    'dropout_rate': '0.1'
}
```

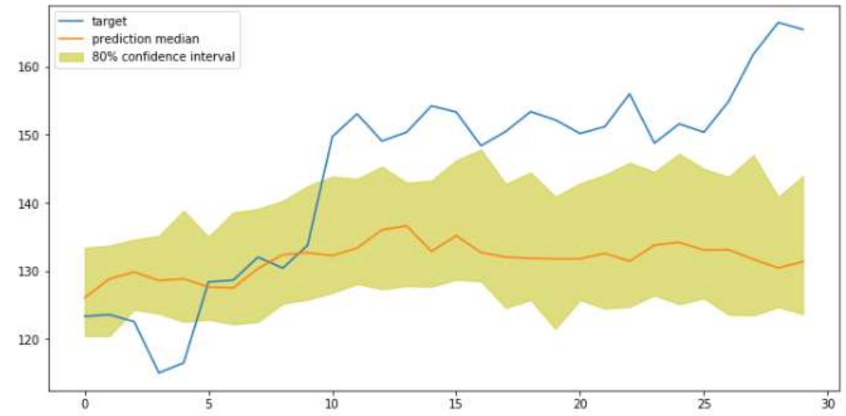


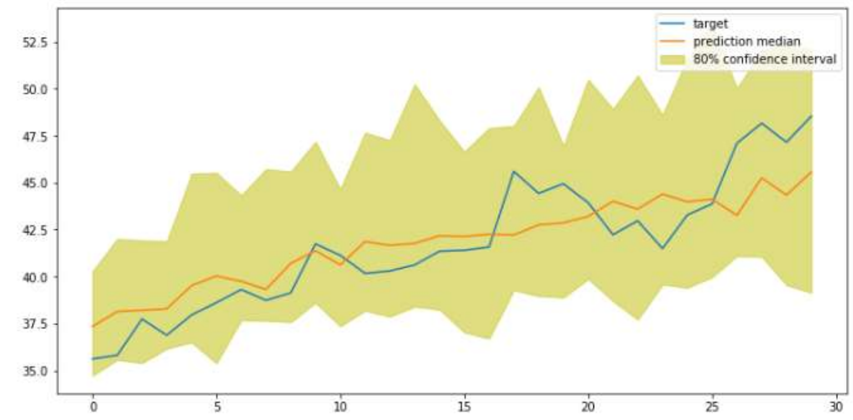
```
%%time
# train and test channels
data_channels = {
    "train": train_path,
    "test": test_path
}

# fit the estimator
estimator.fit(inputs=data_channels)
```

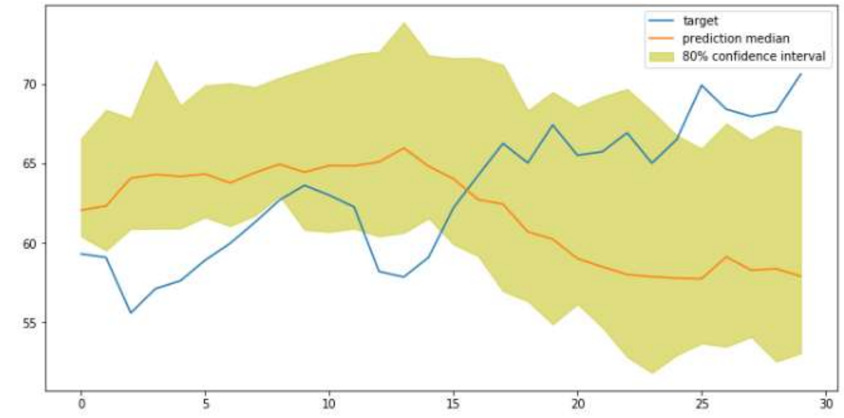
After that I have deployed the trained model to perform predictions by creating a predictor endpoint. The deployed predictor expects to see input data in a JSON format with some configuration parameters, hence have used json_predictor_input function to get a prediction for a formatted time series.

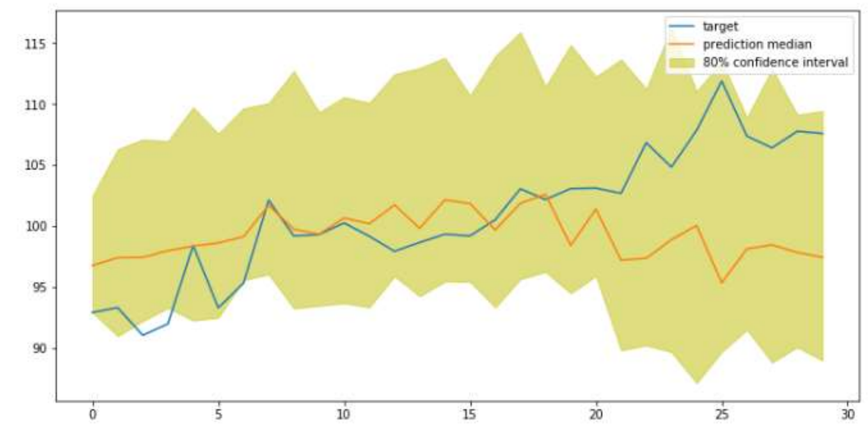
The predictor returned JSON-formatted prediction, and I needed to extract the predictions and quantile data that I wanted for visualizing the result. The function decode_prediction used to reads in a JSON-formatted prediction and produces a list of predictions in each quantile.
In order to visualize the prediction I have used the below graphs with Quantiles 0.2 and 0.9 represent lower and higher bounds for the predicted values and Quantile 0.5 represents the median of all sample predictions.

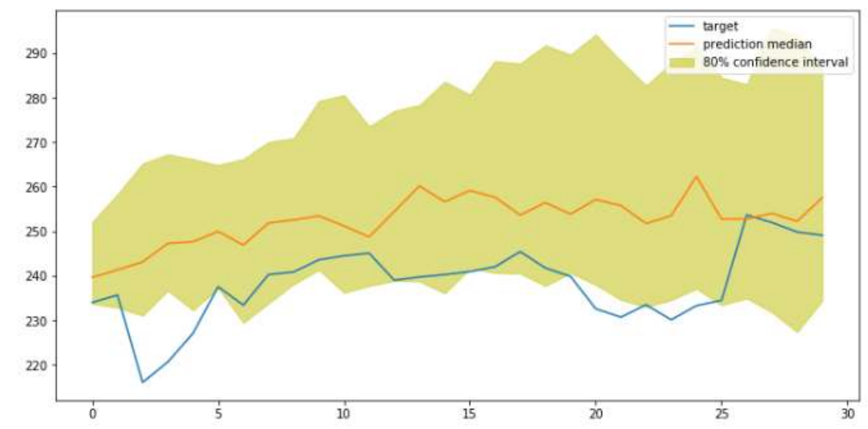

target
prediction median
80% confidence interval
140
130
120
110
100
90
0
5
10
15
20
25
30


target
prediction median
80% confidence interval
160
150
140
130
120
0
5
10
15
20
25
30


target
prediction median
80% confidence interval
52.5
50.0
47.5
45.0
42.5
40.0
37.5
35.0
0
5
10
15
20
25
30


target
prediction median
80% confidence interval
70
65
60
55
0
5
10
15
20
25
30

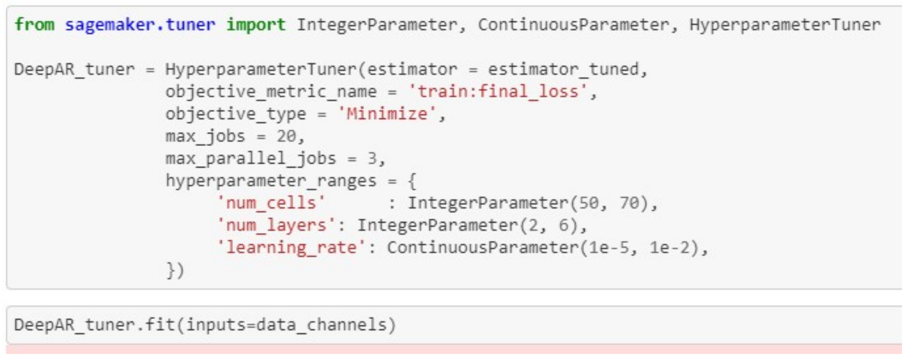




From the above prediction graph of each stocks it seems our predictor did significant well for less volatile stocks like Xilinx Inc(XLNX), 2U Inc(TWOU), Splunk Inc(SPLK) and somewhat good for Facebook(FB) as well, however highly volatile stocks like Tesla Inc(TSLA),Roku Inc Class A(ROKU) and Square Inc A(SQ) did insane market rally in terms of it's stock price in last 4 months and no one had anticipated this much price movement but people's sentiment made it insane which I feel is not possible to predict by using historical stock data and I feel social media data related to these companies would help to consider the market sentiment to make better prediction.

10. Refinement:

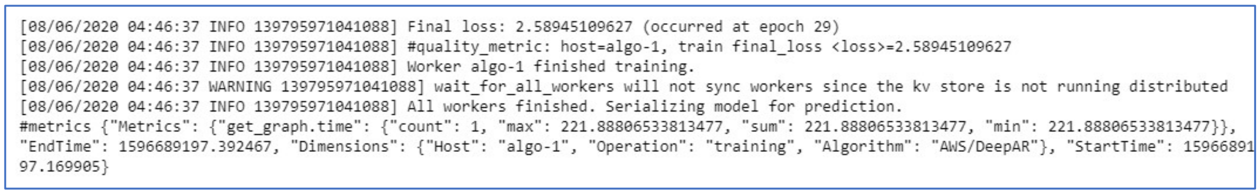
The model which we have trained did not give good prediction for some of the stocks, Hence I have used the below range of DeepAR hyperparameters to tune the model, so that I could get better prediction results.



```
from sagemaker.tuner import IntegerParameter, ContinuousParameter, HyperparameterTuner

DeepAR_tuner = HyperparameterTuner(estimator = estimator_tuned,
               objective_metric_name = 'train:final_loss',
               objective_type = 'Minimize',
               max_jobs = 20,
               max_parallel_jobs = 3,
               hyperparameter_ranges = {
                    'num_cells'      : IntegerParameter(50, 70),
                    'num_layers': IntegerParameter(2, 6),
                    'learning_rate': ContinuousParameter(1e-5, 1e-2),
               })
```

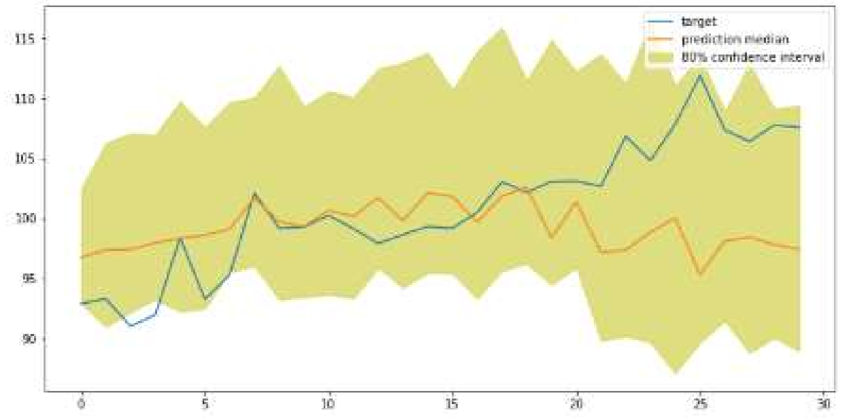
```
DeepAR_tuner.fit(inputs=data_channels)
```

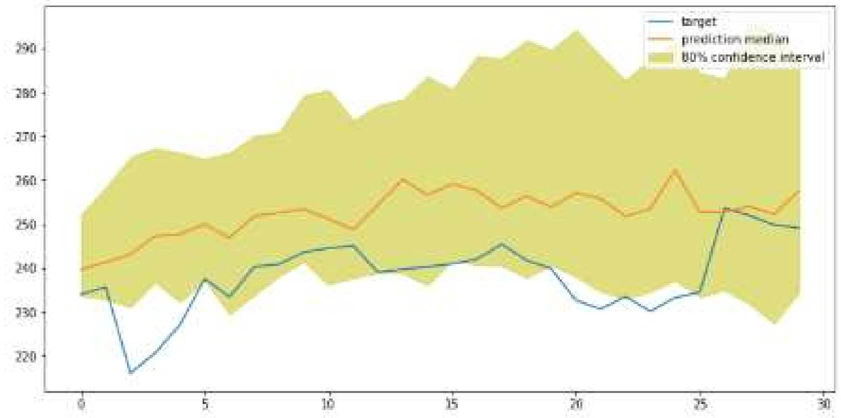
After this hyperparameter tuning the model achieved the highest performance during epoch 29 with Final loss: 2.58945109627and test RMSE score 120.08 which is much more improved from the default hyperparameter model result where test RMSE score was 184.59



```
[08/06/2020 04:46:37 INFO 139795971041088] Final loss: 2.58945109627 (occurred at epoch 29)
[08/06/2020 04:46:37 INFO 139795971041088] #quality_metric: host=algo-1, train final_loss <loss>=2.58945109627
[08/06/2020 04:46:37 INFO 139795971041088] Worker algo-1 finished training.
[08/06/2020 04:46:37 WARNING 139795971041088] wait_for_all_workers will not sync workers since the kv store is not running distributed
[08/06/2020 04:46:37 INFO 139795971041088] All workers finished. Serializing model for prediction.
#metrics {"Metrics": {"get_graph.time": {"count": 1, "max": 221.88806533813477, "sum": 221.88806533813477, "min": 221.88806533813477}},
"EndTime": 1596689197.392467, "Dimensions": {"Host": "algo-1", "Operation": "training", "Algorithm": "AWS/DeepAR"}, "StartTime": 15966891
97.169905}
```

I have created a predictor endpoint to validate the model again using tuned hyperparameters and visualize the same stock's predicted value against the actual value, below are some of the visualization.

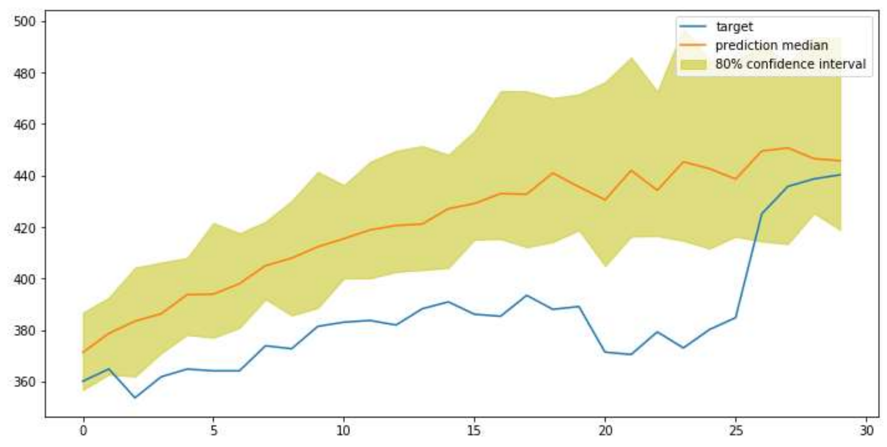




Results:

11. Model Evaluation and Validation:

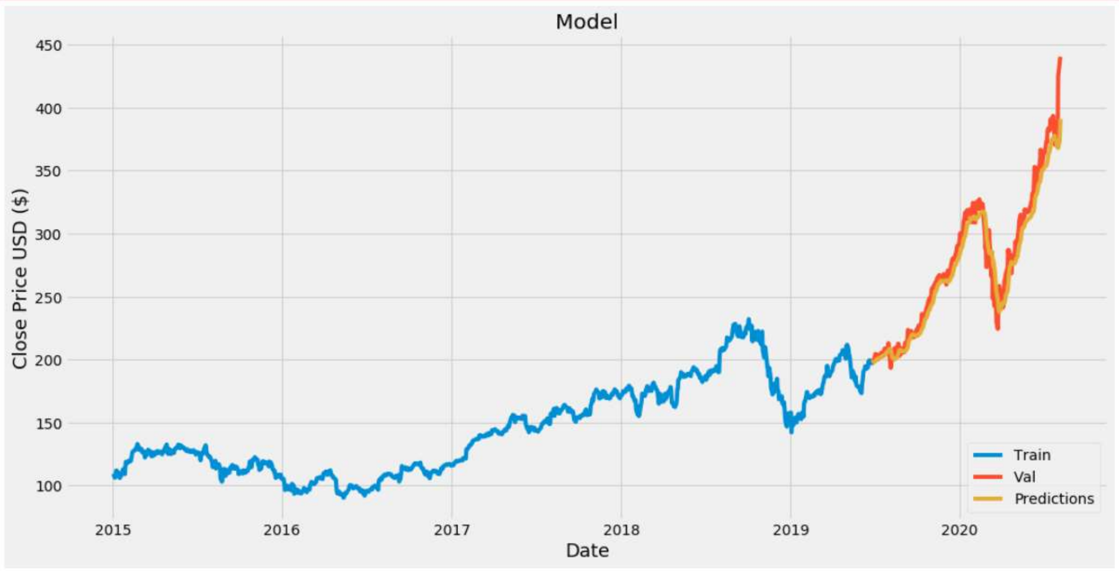
In order to validate my tuned model for it's robustness, I took altogether a new stock Apple(AAPL) which model has not seen before and used the tuned predictor to predict it's price.



The tuned model has worked well for an unseen Apple stock data to predict the trend for last 30 days and it is able to predict upward and downward price movement, also RMSE is a lot better in this which is 40.50. However predicted price is going outside the confidence interval and the reason is apple had quarterly result on 30-July hence for last 30 days it was so volatile and market sentiment was driving it more, another reason was that Apple's 4-for-1 stock split news fuels bets and that also made this stock more volatile.
I believe still this model did well even in this sentiment driven market.

12. Justification:

I have compared my final model's results with my benchmark model which is Long Short-Term Memory network(LSTM), LSTM is also very well known for time-series prediction. I used the same Apple stock for this benchmark model.



It seems LSTM is predicting data more accurately and RMSE value 12.75 is also significant less than the tuned DeepAR model. Also from the above graph it looks like LSTM model is able to predict upward and downward trend very effectively, and stock predicted value is very close to actual stock value.

Conclusion:

In this project, I built a stock price predictor to predict a stock price for next 30 days.
In order to create this model I have taken the historical data from Yahoo Finance and performed the require preprocessing into the data, then fed this input data to the DeepAR model using default hyperparameters which have not given the very good prediction result (Test RMSE: 184.59), So in order to optimize this model I have used DeepAR hyperparameter tuning and evaluated the tuned model which has achieved the highest performance during epoch 29 with Final loss: 2.58945109627and test RMSE score 120.08 which is much more improved from the model with default hyperparameters.

While working on this Capstone project I thoroughly studied Long Short-Term Memory network(LSTM) which has suggested by a lot of researchers for timeseries based prediction problems, So out of curiosity I have also implemented LSTM for Apple stock data and found the surprising results as it has predicted data more accurately and RMSE value 12.75 is also significant less than the tuned DeepAR model.

Improvement:

One thing which I really want to improve on this predictor is by feeding more attributes like company-specific variables (P/E Ratio, Market Cap, Volume, Avg Volume) and company's historical/future events like earnings, upcoming major launches which actually drives their stock price so much.

In addition to this I will also feed social media sentiment data to the model for better prediction, As I have noticed in the current very volatile market that people/Market sentiment is driving the market in this pandemic situation, as a observation I found that in June/July there were many companies who were not doing great fundamentally however still their stock price had surge which was purely because of positive sentiments, and the model was not able to predict well.

Hence I believe we can make much better stock price predictor if we will feed social media sentiment's data along with historical stock data to the model.

Reference:

- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
- https://en.wikipedia.org/wiki/Stock_market
- https://www.investopedia.com/articles/07/stock-exchange-history.asp
- Chong, Eunsuk, Chulwoo Han, and Frank C. Park. 2017. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications 83: 187–205. [CrossRef]
- Arévalo, Rubén, Jorge García, Francisco Guijarro, and Alfred Peris. 2017. A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. Expert Systems with Applications 81: 177–92. [CrossRef]

- https://www.researchgate.net/publication/328832048_Quantitative_Analysis_of_Stock_Market_Prediction_for_Accurate_Investment_Decisions_in_Future
- Zhong, Xiao, and David Enke. 2017. Forecasting daily stock market return using dimensionality reduction. Expert Systems with Applications 67: 126–39. [CrossRef]
- https://rapidapi.com/apidojo/api/yahoo-finance1/endpoints