Machine Learning Engineer Nanodegree

Capstone Project

“Machine Learning Approach to

Stock Price Prediction using LSTM”

Omar Ali

January 11, 2019

I.Definition

Project Overview

Stock price prediction is a popular topic throughout the world. The stock market has always been a goldmine of adventures for mathematicians and statisticians. They keep trying to find patterns such that the behavior of stock market and how it can be predicted, however the huge amount of data and buy/sell decisions carried out every day makes it almost impossible to be analyzed manually. However, in the recent few years the emergence of new organized data, high computational power and machine learning algorithms has given us the ability to use computer for predicting the behavior of stock market.

Stock market prices are highly unpredictable and volatile. This means that there are no consistent patterns in the data that allow you to model stock prices over time near-perfectly. Don't take it from me, take it from Princeton University economist Burton Malkiel, who argues in his 1973 book, "A Random Walk Down Wall Street," that if the market is truly efficient and a share price reflects all factors immediately as soon as they're made public, a blindfolded monkey throwing darts at a newspaper stock listing should do as well as any investment professional.

However, let's not go all the way believing that this is just a stochastic or random process and that there is no hope for machine learning. Let's see if you can at least model the data, so that the predictions you make correlate with the actual behavior of the data. In other words, you don't need the exact

stock values of the future, but the stock price movements (that is, if it is going to rise of fall in the near future).

It seems like machine learning and even deep learning methods can yield good result in stock price prediction. That is why I would like to try my own methods in this project. In this project, I will use the American Airline Stock from the S&P 500 as inputs and target. For each stock, the data will contain Open, High, Low, Close, and I will calculate the median value from the Low and High value and use it to split the data. I will query the historical prices from Alpha Vantage through its provided API.

Problem Statement

The project aims to create a machine learning model that can predict stock price by using historical information as a time-series data. The task is to build a stock price predictor that takes daily trading data over a certain date range as input, and outputs projected estimates for given query dates. The inputs will contain multiple variables, such as opening price (Open), highest price the stock traded at (High), opening price (Open) and closing price (Close); my system will need to predict the closing price calculated from the (Low) and (High) prices.

Metrics

This is a regression problem, so the metrics of choice would be R-square and root-mean squared- error. R-square can provide how much variation in the dependent variable can be explained by the variation in the independent variables. Root-mean-squared-error can provide what is the average deviation of the prediction from the true value, and it can be compared with the mean of the true value to see whether the deviation is large or small.

II. Analysis

Data Exploration

There are 505 companies in the S&P 500. The list of companies and symbols can be find here. My data will include all these stocks' Open, High, Low, Close. The data will start from 2005-09-27 and end at 2018-12-28. A glimpse of a typical stock's data is shown in table 1.

Date	High	Low	Open	Close
2005 – 09 – 27	21.40	1910	21.05	19.30
2005 – 09 – 28	20.53	19.20	19.30	20.50
2005 – 09 – 29	20.58	20.10	20.40	20.21
2005 – 09 – 30	21.05	20.18	20.26	21.01
2005 – 10 – 03	21.75	20.90	20.90	21.50

Table 1: Head of Historical Prices for AAL

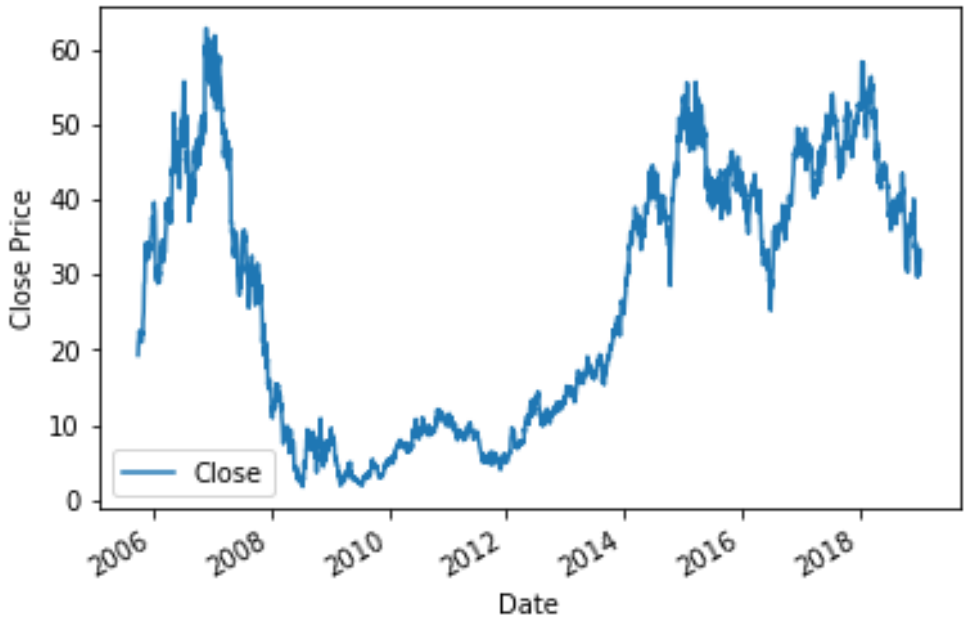
In this project, I will calculate the closing price as target for illustrative purpose. The model can be applied on any of the stocks. The stock I pick is the AAL.

Minimum	Q1	Median	Mean	Q3	Maximum	std
1.76	9.1	29.89	26.877218	42.42	62.95	17.4

Table 2: Close Price Statistical Analysis for AAL

Data Visualization

We can visualize the target series over time. In Figure 1, we can see that there is an upward trend in the series, which is the trend for most of the stock prices in S&P 500. There is also a huge drop in 2008 because of the financial crisis. If we are using statistical analysis, then there will be a lot of statistic test and data preprocessing going on because we have to make sure the dependent and independent variable are stationary, otherwise they have



to be co-integrated. But for neural network, there is no need to do these and do feature engineering (generate rolling window aggregates-like features), we can just use the raw time series.

Algorithms and Techniques

In this project, I would like to use recurrent neural network to solve the problem. Simply put, traditional neural networks take in a stand-alone data vector each time and have no concept of memory to help them on tasks that need memory. This is where the Long Short Term Memory (LSTM) neural network came to the rescue. Like RNN neurons, LSTM neurons kept a context of memory within their pipeline to allow for tackling sequential and temporal problems without the issue of the vanishing gradient affecting their performance. I will use Long-short term memory network as the model, the AAL stock as the target, and top 11 years data as input. I will tune the sequence length which can yield the best result, and set this as the length of the LSTM network. The prediction will be one-step-ahead stock price of the

Figure 1: Close Price of AAL

target stock. Once the model is trained, the model will choose the first 11

years' close prices as input, and the model can predict the last 2 years close price.

Benchmark

The benchmark model for this project would be a linear regression. Including lagged features of the dependent variable and other exogenous features in a linear regression is called autoregressive model with exogenous inputs, which is proved to be very successful in statistical time series

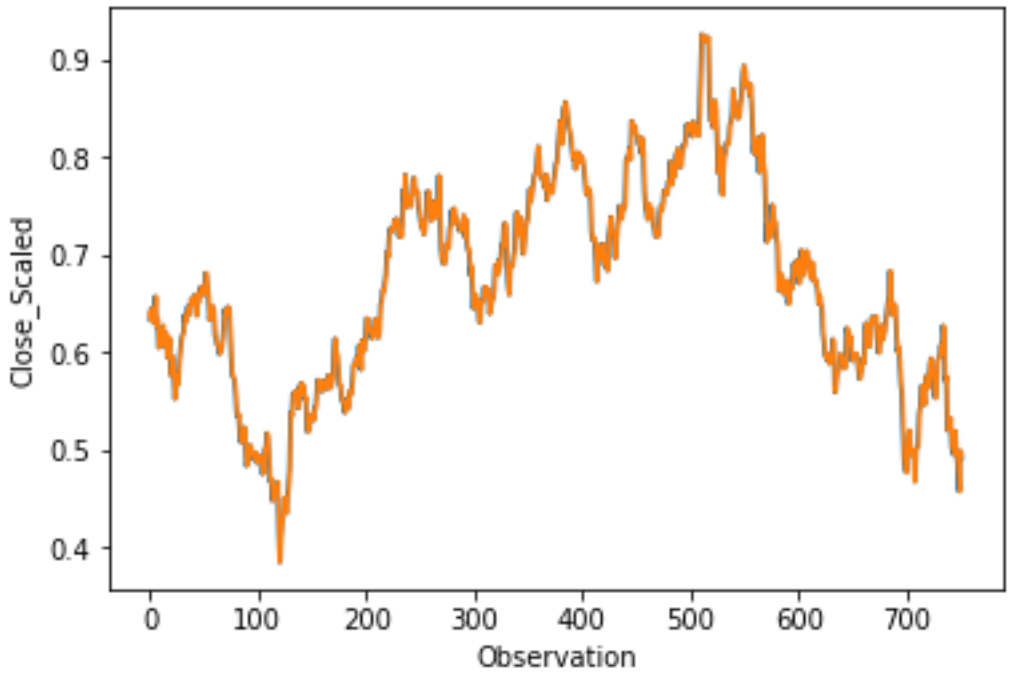


Figure 2:Inear Regression Prediction against Truth for AAL

modeling. The model is trained on 2005-2016, the result on test period (2016-2018) is shown in figure 2. We can see that without doing required statistic test and data preprocessing like stationary check and co-integration check, linear regression on raw time series data can yield unrealistic results in my opinion for the linear regression.

III. Methodology

Data Preprocessing

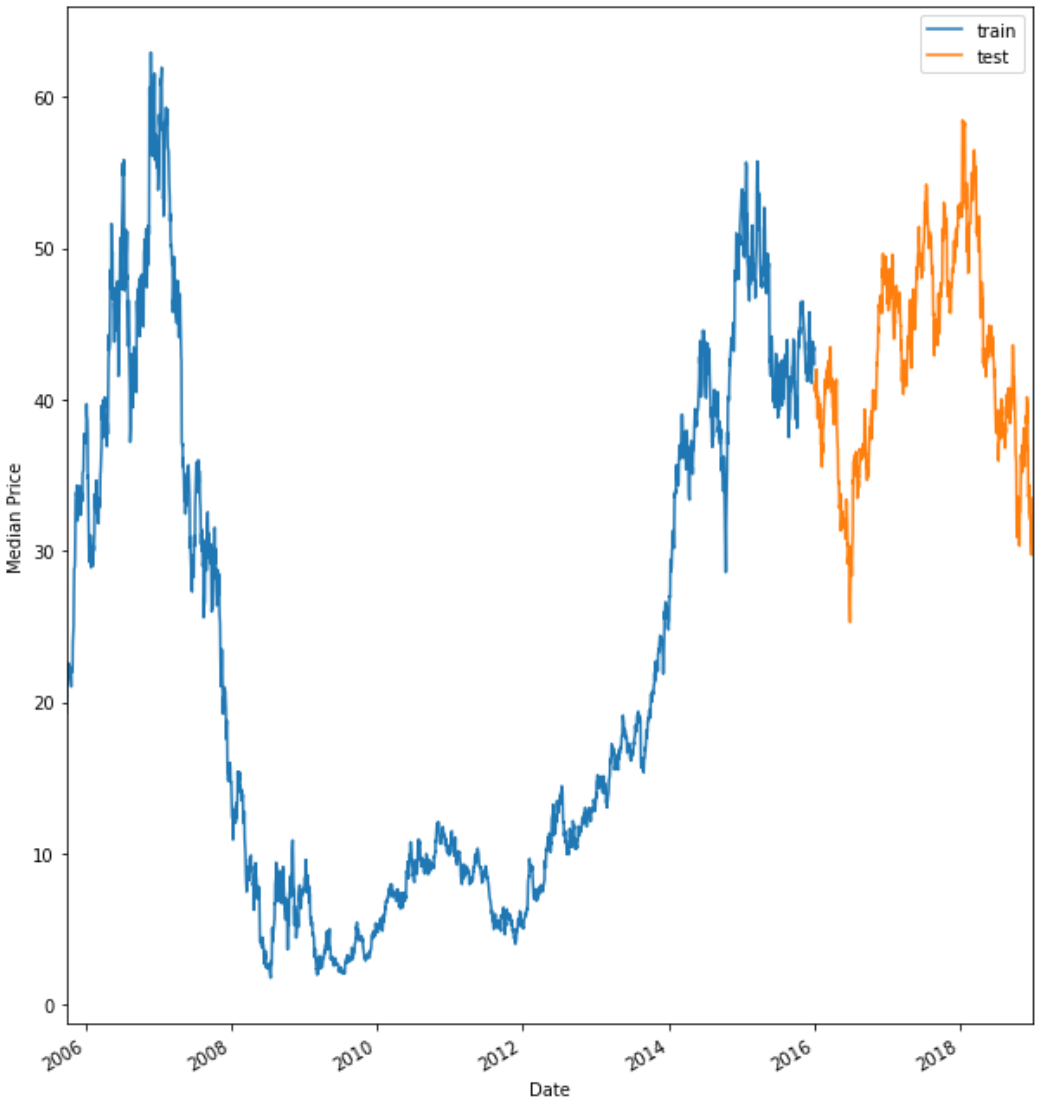


Figure 3: Train-Test Split for AAL

Then we will split the data into training and testing period. The training data contain data from 2005 to 2016, and the test data contain data from 2016 to December 2018. We have prices in different scales. In order to make neural network converge faster, we should scale our inputs. We will use Min-Max scaler to scale the input in the range of 0 to 1 for all data in the training period. Then we will use the scaler which has the min max information of the training period to scale the data in testing period. We split the scaling process because we do not want to have information leakage, which leaks testing period information into training period. Figure 3 will show the split between training data and testing data.

Implementation

A fully Connected Model is a simple neural network model which is built as a simple regression model that will take one input and will spit out one output. This basically takes the price from the previous day and forecasts the price of the next day. As a loss function, we use mean squared error and stochastic gradient descent as an optimizer, which after enough numbers of epochs will try to look for a good local optimum. Below is the summary of

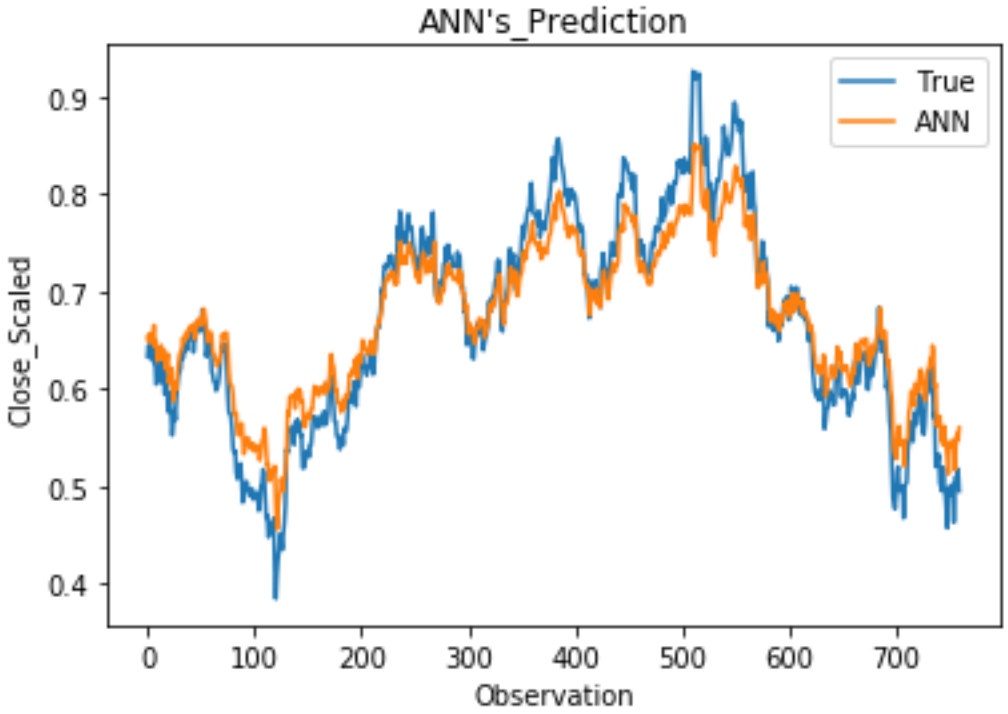


Figure 4: ANN Prediction for AAL

the fully connected layer. After training this model for 200 epochs or early-callbacks (whichever came first), the model tries to learn the pattern and the behavior of the data. Since we split the data into training and testing sets we can now predict the value of testing data and compare them with the ground truth.

As you can see in figure 4, the model is not good. It essentially is repeating the previous values and there is a slight shift. The fully connected

model is not able to predict the future from the single previous value. Let us now try using a recurrent neural network and see how well it does.

Long Short-Term Memory

Long Short-Term Memory models are extremely powerful time-series models. They can predict an arbitrary number of steps into the future. An LSTM module (or cell) has 5 essential components which allows it to model both long-term and short-term data. The recurrent model we have used is a one-layer sequential model. We used 6 LSTM nodes in the layer to which we gave input of shape (1,1), which is one input given to the network with one value. The last layer is a dense layer where the loss is mean squared error with stochastic gradient descent as an optimizer. We train this model for 500 epochs with early-stopping callback. The summary of the model is shown above.

This model has learned to reproduce the yearly shape of the data and doesn't have the lag it used to have with a simple feed forward neural network. It is still underestimating some observations by certain amounts and there is definitely room for improvement in this model.

Refinement

There can be a lot of changes to be made in this model to make it better. One can always try to change the configuration by changing the optimizer. Another important change I see is by using the Sliding Time Window method, which comes from the field of stream data management system.

This approach comes from the idea that only the most recent data are important. One can show the model data from a year and try to make a prediction for the first day of the next year. Sliding time window methods are very useful in terms of fetching important patterns in the dataset that are highly dependent on the past bulk of observations.

There are a few hyper-parameters in the models I have tried to tune. The first is the length of the data set. At first I was hoping to have a larger data set for better variation but unfortunately the only data set I was able to find is from 2005 until the end of 2018. Also I believe that the 2008 crisis

might have an effect on the training. The second hyper-parameter is the optimizer. I tried using RMSprop, Adam and Nesterov Adam, and the result are pretty similar. But I decided to go with Adam. The final hyper-parameter is the number of epochs. According to the learning curve I will show in the next section, the loss function converges for both training and testing set converge after 500 epochs, so I tried using epochs of 500 and 1000, and it turns out 500 gives better result.

IV. Results

Model Evaluation and Validation

Using the model described above to train on training data set, we can use the trained model to predict on testing data set. The result is shown in the figure 5. We can see that the prediction is very good. It performs well in the testing data, it almost perfectly matched without overfitting. The R-square is 0.981 and the RMSE is 0.015. Although R-square of 0.981 is not a very realistic number in common sense, but considering our data available, this result is quite good. The RMSE is way smaller than the mean value of the testing data set, which also indicate this model has a good performance.

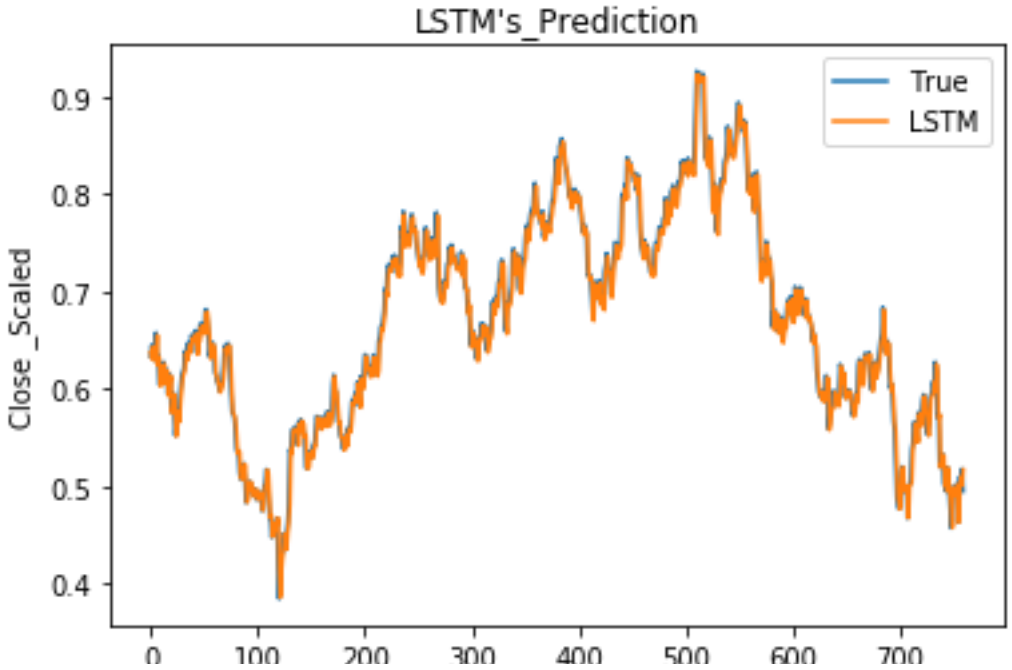


Figure 5: LSTM Prediction for AAL

Justification

Now we can compare the model performance of benchmark model and LSTM model. In table 3 we can clearly see that the LSTM model perform much better than the benchmark model. As I discussed in the benchmark model section, linear regression on time series data should be taken good care of in statistic tests and data preprocessing, otherwise it might fail. However, LSTM model will generate its own feature representation in the hidden units, and it's long-short term memory cell is good at keep useful information and throw away useless information through its gate units, that is why LSTM is simple but useful in dealing with time series data. Although the metrics shows the linear regression might have similar results to the LSTM, but as I mentioned earlier these results from linear regression might be unrealistic.

Model	R-Squared	RMSE
Linear Regression	0.981	0.0149
LSTM	0.981	0.015
ANN	0.946	0.0256

Table 3: Results for AAL

V. Conclusion

Free-Form Visualization

A very important topic for neural networks is its convergence of loss function. Not only the convergence will show whether the model is successfully trained, it will also tell you whether the model is under-fitting or over-fitting. Figure 7 shows the learning curve of the training process of the model. We can see from the curve that both the loss for training and validation set is decreasing as the number of epoch increases, they converge slightly at the end. This indicates the model is successfully training, at least is converging into a local minimum.

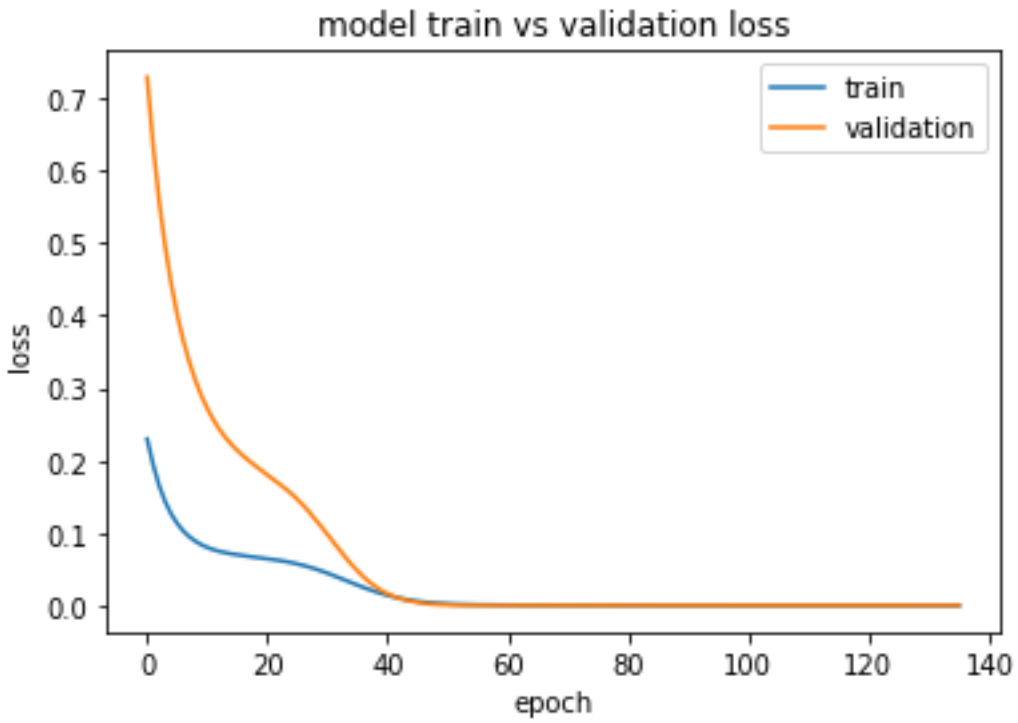


Figure 6: Learning Curve for LSTM

Reflection

This project discusses a machine learning approach to predict the stock price, more specifically, the close price, using lagged terms of itself and other stocks' prices and trading volumes. First we will query historical stock price data from Alpha Vantage. Actually getting data is the most difficult problem in this project. Previously people will use pandas built in method to query historical stock price, and it works well with Yahoo-finance and Google Stock. However, after May 2017, Yahoo-finance closed its developer API, and Google Stock stopped permission for automatic query. Although pandas still works fine with Yahoo-finance, but it will shut down request once you made around 10 consecutive queries. A very well-known finance data provider Quandl provide API to their database, however the data quality is bad, there are a lot of missing values in the database. After we have generated our feature sets, we can input these features into the benchmark linear regression and LSTM networks. I will use scikit-learn's Linear Regression class to do the linear regression, and use Keras's LSTM function to build the neural

networks. I split the data into 2005 to 2016 as training, and the rest as testing data. I use R-square and RMSE to check the model performance, on both benchmark model and LSTM model. The final result turns out that the LSTM model is much better than the benchmark model, and can successfully and robustly predict the close price of multiple stocks. I believe this model can be implemented on all the stocks and can yield good prediction on most of them.

In order to check my model I tried the model with same parameters with Apple stock with the sames conditions. The results are in the below table 4.

Model	R-Squared	RMSE
Linear Regression	0.99	0.0035
LSTM	0.995	0.0037
ANN	0.899	0.017

Table 4: Results for AAPL

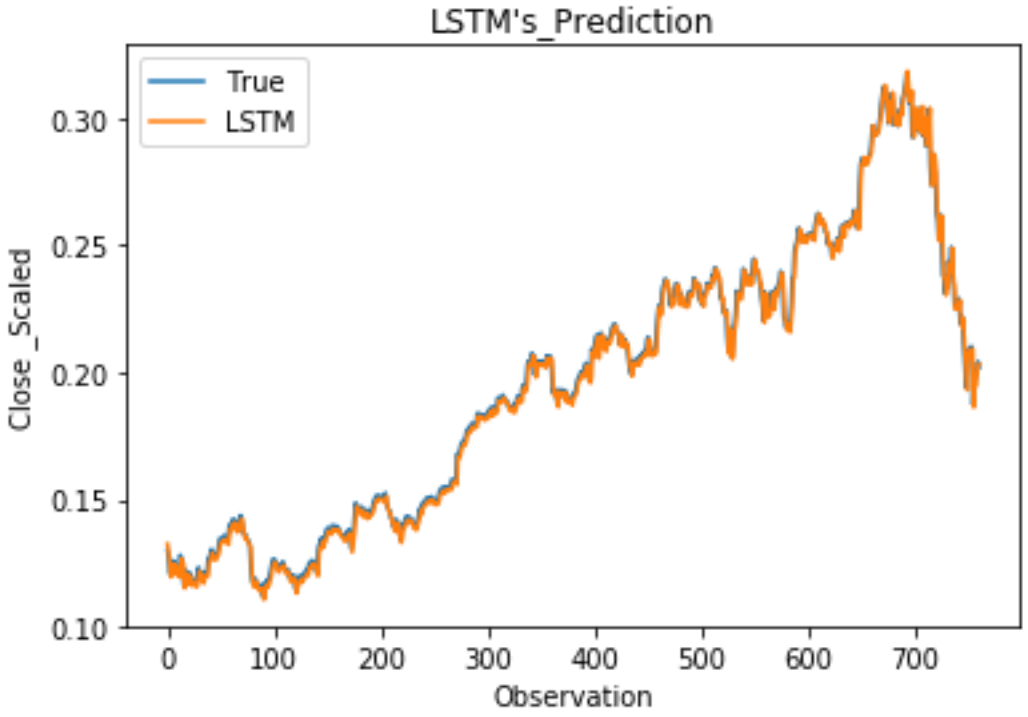


Figure 7: LSTM prediction for AAPL

Improvement

There can be a lot of changes to be made in this model to make it better. One can always try to change the configuration by changing the optimizer. Another important change I see is by using the Sliding Time Window method, which comes from the field of stream data management system.

This approach comes from the idea that only the most recent data are important. One can show the model data from a year and try to make a prediction for the first day of the next year. Sliding time window methods are very useful in terms of fetching important patterns in the dataset that are highly dependent on the past bulk of observations. One thing can try is use dimensional reduction techniques like PCA, and choose top 200 principle components as the features. The reason why I did not use this in this project is PCA generate new dimensions based on the direction of maximal variance, but this variance may not be relevant to the target variable. Doing PCA might totally mess up the information in the feature set, that is why I choose a more conservative method in the project, but it is definitely worthy to try different dimensional reduction techniques and check their performance.

Reference

Sreelekshmy Selvin, Vinayakumar R, Gopalakrishnan E.A, Vijay Krishna Menon, Soman K.P “STOCK PRICE PREDICTION USING LSTM,RNNAND CNN-SLIDING WINDOW MODEL”, 2017 International Conference on Advances in Computing, Communications and informatics (ICACCI)