MECEE4520

DATA SCIENCE FOR MECHANICAL SYSTEM

Final Project

Amazon Stock Daily Close Price Prediction

Dec 20, 2020

Team Member & UNI:
Zhengnan Chen, ZC2544
Jiahao Wu, JW3997

Instructor:
Dr. Josh Browne

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

TABLE OF CONTENTS:

Introduction

Abstract

In the stock market, stock investment's biggest problem is that the information and data in the stock market are complex and excursive. Individual investors are not always able to receive useful and helpful information. Sometimes they even have to pay extra to get this kind of information. Until today, There are still fewer data science tools in the market which are able to provide comprehensive information to individual investors and use machine learning to help their investment decision. The individual investor often replies on some cost-free "Gossip" instead of scientific-technical analysis to decide the stock market's next action. As a result, Individual investors often lose a lot of their investment for the above reasons.

Team plans to aid in such development by designing a machine model that can predict a daily close price for a company which could provide help and suggestions to individuals investors in their stock daily trade.

Project Objective & Goal

After the appearance of machine learning, more and more researchers realized that machine learning can be applied to stock information prediction. There are plenty of thoughts on using a company's stock information such as daily open, daily close, daily high and daily volumes to predict the future stock price of this company. However, many cases approve that only using daily open, high, low, volumes to predict the stock close prices is not workable. In this project, the team develops a new idea to develop machine learning models to predict the daily close price for Amazon.

The goal of this project is to use the knowledge gained from data science courses to collect and analyze stock market data and develop a machine learning model that can predict the daily close price. From Yahoo financial API, the team is able to collect stock data and other technical indicators for stock data analysis and machine learning prediction model training.

The idea of this project is to input more variables such as using the company's stock data which has more relationship with Amazon as one of the variables in machine learning model development. And the team also uses stock indicators such as VIX (volatility) index as variables and into machine learning models for a stock daily close price prediction. Furthermore, the team develops more than one machine learning model by using different methods to predict the amazon daily close price. The team compared each algorithm to find the best machine learning model that has highest accuracy in close price prediction. The team starts with basic machine learning algorithms such as linear regression and quadratic regression, and then develops advanced models like the deep neural network (DNN) and Long short-term memory (LSTM) model to predict the daily close price. As a result, the team expected to develop a machine learning model which was able to predict the amazon daily close price with high accuracy.

To achieve this project's goals, the team decided to use Python as the primary program language. Python is widely used and is one of the favorite tools being a flexible and open-sourced language. Python has massive libraries that are used for data manipulation and analysis. Python also has a lot of useful tools to help the team develop the machine learning model and develop new features for data analysis.

Data Collections & Analysis

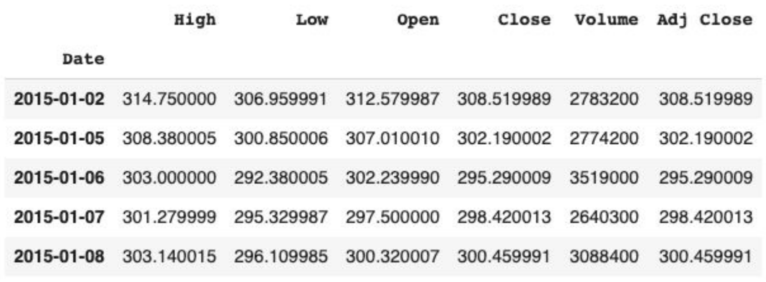
Data Source

By comparing with multiple different data API sources, the team decided to use Yahoo finance API as data collection. Yahoo Finance is one of the reliable data sources for the stock market. It supports python language and is able to provide comprehensive stock market summaries, historical and real-time quotes. The team is able to get detailed information like date, open price, high price, low price, last price, close price, etc. And another advantage of using yahoo finance is that it has easy coding and responds faster than other stock API and supports python program language. It will help the team to make a comprehensive data analysis and develop a more accurate machine learning model for price prediction.

Data Selection

In this project, the team selected six different company stocks, and one stock market indicator index. All six companies either have similar business areas or have a close partnership or business competition with Amazon. The first company is Amazon (Stock symbol: AMZN), which is the company that the team expects to predict its daily close price. The second and third companies are Walmart (Stock symbol: WMT) and eBay (EBAY). Walmart is one of the largest physical sales companies in the world. And eBay is one of the large online sales companies, similar to Amazon. eBay, Walmart, and Amazon are competing between each other but also cooperating in the market. The three companies together represent the US retail industry. The fourth company is Alibaba (Stock symbol: BABA). For Alibaba, Just as most American consumers refer to Amazon as an e-commerce giant, Alibaba leads the Chinese e-commerce market. Apart from online sales, Alibaba and Amazon both have cloud businesses, which will make their relationship even closer in the stock market. The fifth and sixth companies stocks are Facebook and Google; the reason to choose Facebook and Google is because Facebook, Google, and Amazon are the leaders in US technology stocks, and therefore, each company’s behavior affects other stocks. The last variable team selected is one of the essential market indexes: the Volatility index (VIX). compared with normal stock, the VIX index has the opposite trend with market stock. VIX index starts to rise during times of financial stress and lessens as investors become complacent. The VIX index is one of the best options to use for the prediction of near-term market trends.

Data collection

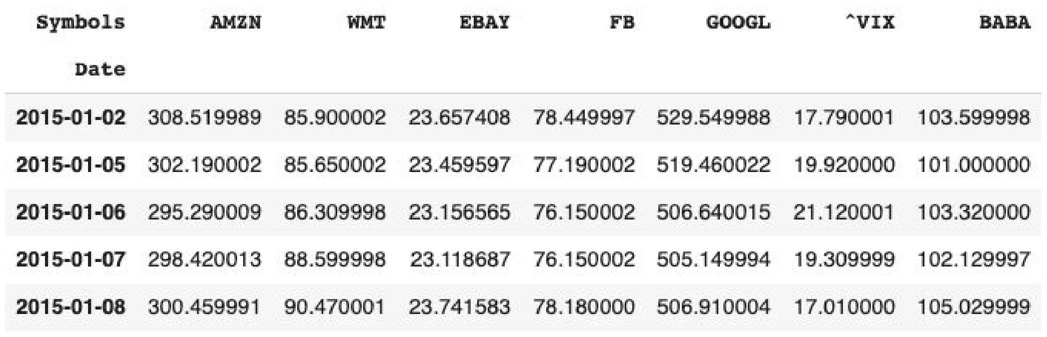


Date	High	Low	Open	Close	Volume	Adj Close
2015-01-02	314.750000	306.959991	312.579987	308.519989	2783200	308.519989
2015-01-05	308.380005	300.850006	307.010010	302.190002	2774200	302.190002
2015-01-06	303.000000	292.380005	302.239990	295.290009	3519000	295.290009
2015-01-07	301.279999	295.329987	297.500000	298.420013	2640300	298.420013
2015-01-08	303.140015	296.109985	300.320007	300.459991	3088400	300.459991

Figure 1. Table for Amazon daily close price

As above introduced, the team used Yahoo finance API to collect the historical data for selected companies and selected indicator index. As figure 1 shows, seven different types of data in each data set: time, high prices, low prices, open prices, close prices, trade volumes, and adj close price. Based on financial knowledge, 5-8 years is one economic cycle. The team determined to collect five years of stock data from 2015 to 2020 as dataJust as most American consumers refer to Amazon as an e-commerce giant, set for the machine learning model development.

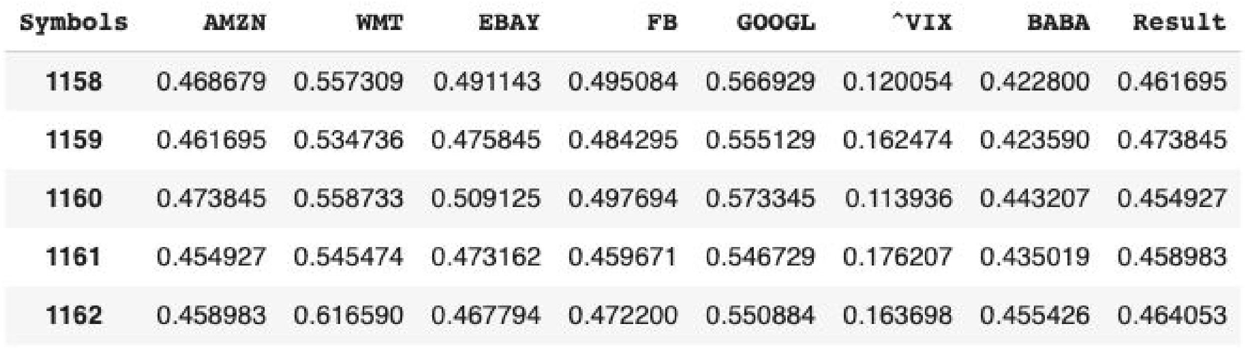
Data Processing



Symbols Date	AMZN	WMT	EBAY	FB	GOOGL	^VIX	BABA
2015-01-02	308.519989	85.900002	23.657408	78.449997	529.549988	17.790001	103.599998
2015-01-05	302.190002	85.650002	23.459597	77.190002	519.460022	19.920000	101.000000
2015-01-06	295.290009	86.309998	23.156565	76.150002	506.640015	21.120001	103.320000
2015-01-07	298.420013	88.599998	23.118687	76.150002	505.149994	19.309999	102.129997
2015-01-08	300.459991	90.470001	23.741583	78.180000	506.910004	17.010000	105.029999

Figure 2. Summary of data set.

The team used daily close data as the input data of machine learning models. The team picked the “close” columns in each stock company’s historical data set and then to trim all the data together. As figure 2 shown. It is the summary of data used to model training and test. In this data set, each column includes 1148 days daily close price.



Symbols	AMZN	WMT	EBAY	FB	GOOGL	^VIX	BABA	Result
1158	0.468679	0.557309	0.491143	0.495084	0.566929	0.120054	0.422800	0.461695
1159	0.461695	0.534736	0.475845	0.484295	0.555129	0.162474	0.423590	0.473845
1160	0.473845	0.558733	0.509125	0.497694	0.573345	0.113936	0.443207	0.454927
1161	0.454927	0.545474	0.473162	0.459671	0.546729	0.176207	0.435019	0.458983
1162	0.458983	0.616590	0.467794	0.472200	0.550884	0.163698	0.455426	0.464053

Figure 3. Normalized Data set

Before inputting the data set into the machine learning model, the time data has to be deleted from the data set and also data needs to be normalized. As above figure 3 shown, the date has been deleted from the data set and all of the data has been rescaled which will help to increase the prediction accuracy of the machine learning model. And also increase the speed of machine learning training time. The team separated the data set into two parts. One part for model training and another for model testing. The team uses 80% of data for training and 20% of data for testing.

Data Analyze

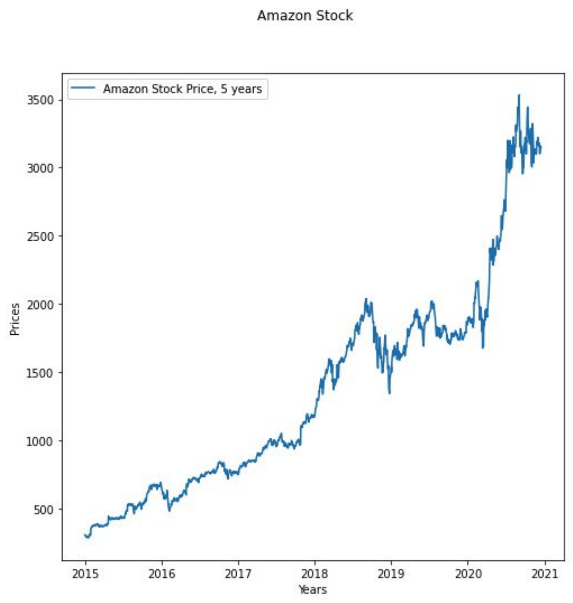


Figure 4. Amazon Stock

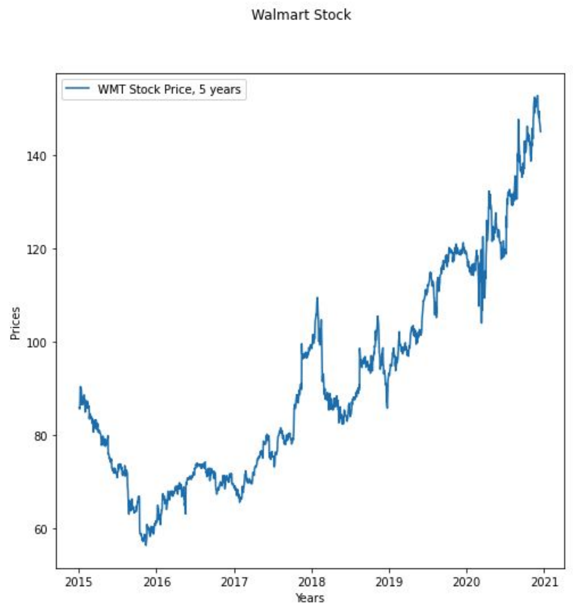


Figure 5. Walmart Stock

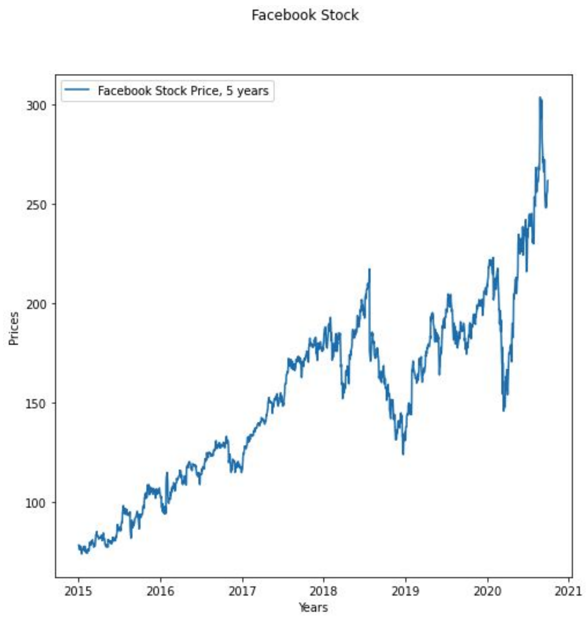


Figure 6. Facebook Stock

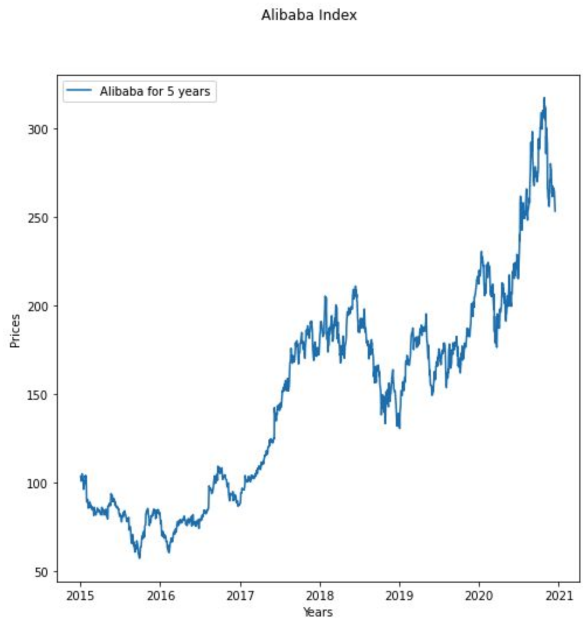


Figure 7. Alibaba Stock

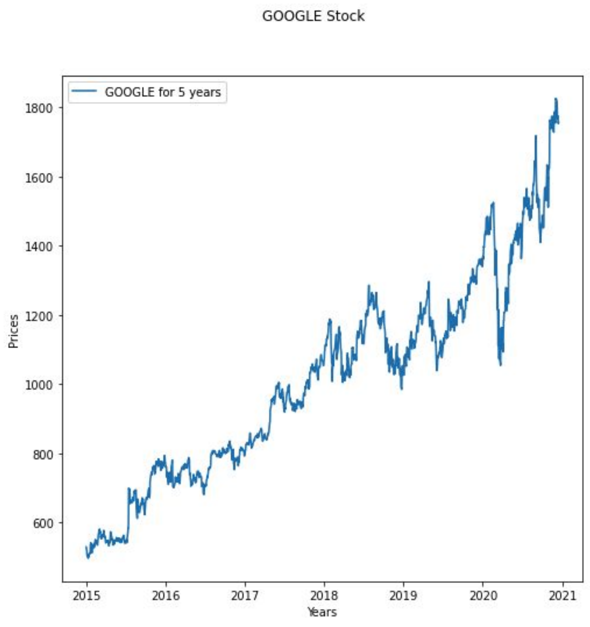


Figure 8. Google Stock

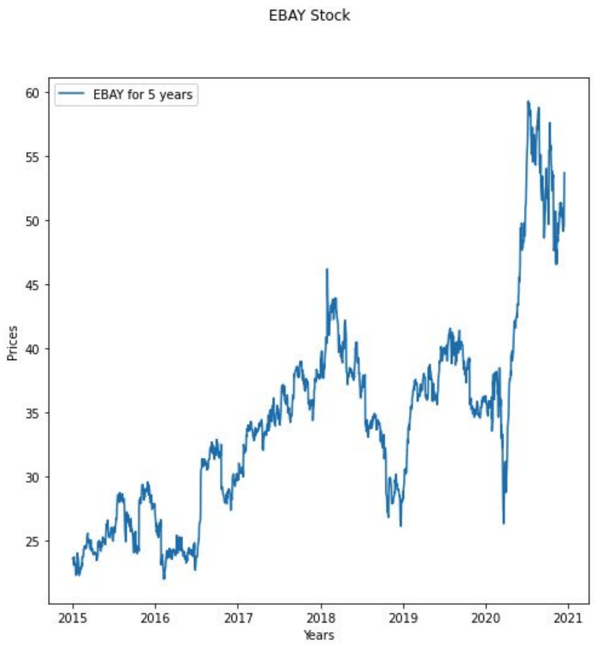


Figure 9. Ebay Stock

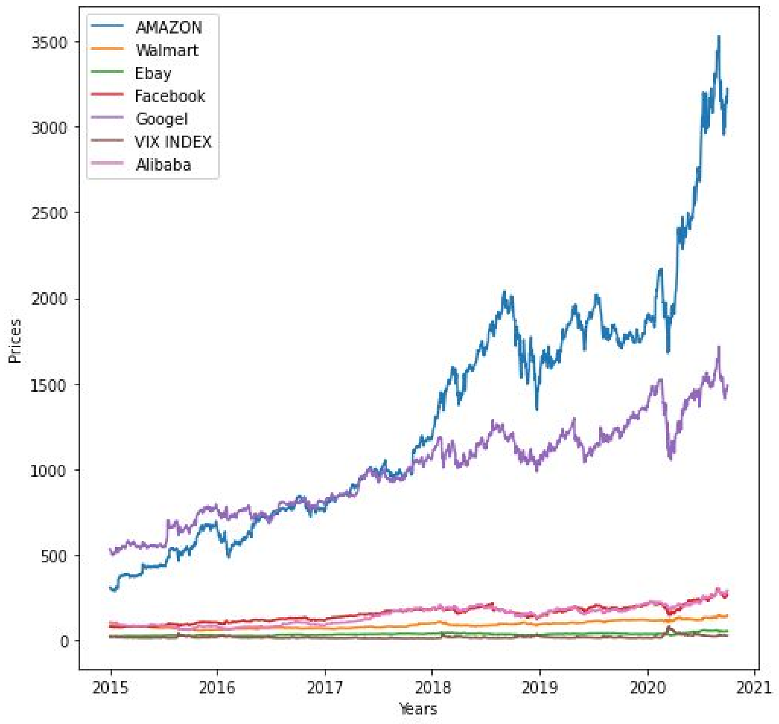


Figure 10. Summary Plot for all selected Stock and Indicator Index

To better understand all of the selected stock company, the team plots the data which is able to see and compare relationships between each stock. Above figure 4-10 shows the daily close price from 2015 to 2020 for all six companies.

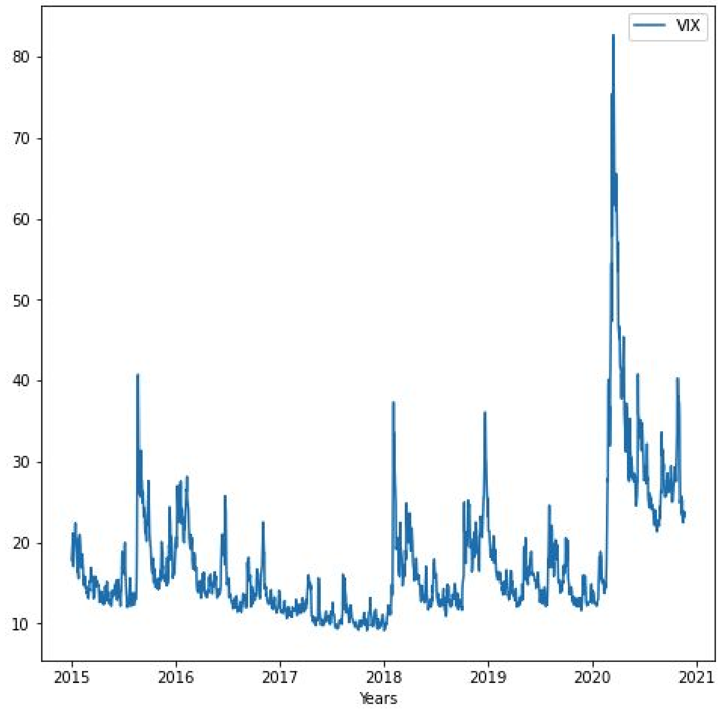


Figure 11. Volatility (VIX) index

The above figure shows the plot of the volatility index from 2015 to today. The VIX is one of the popular measurements to measure the stock market's expectation for volatility based on the S & P 500 index. The VIX represents how bad the stock market is. If the VIX index increases, what happens in the stock market is that people sell out their stocks and stock marketing starts to lose money in the market. In other words, when the stock market goes down, the VIX index goes up. Based on this relationship, the team found that Vix index and company stock are opposite relationships. For example, the stock market crash of 2020 on Monday, March 9, with history's largest point plugs for all stock indexes such as the Dow Jones Industrial Average(DJIA) and Nasdaq index. However, the VIX index has a huge jump at the same time. Using the VIX index as part of training data for machine learning will help the ML model predict the stock more accurately. To compare the above six companies' stocks with the VIX index, it is very easy to observe that VIX increased during the all of six company's stock slipped.

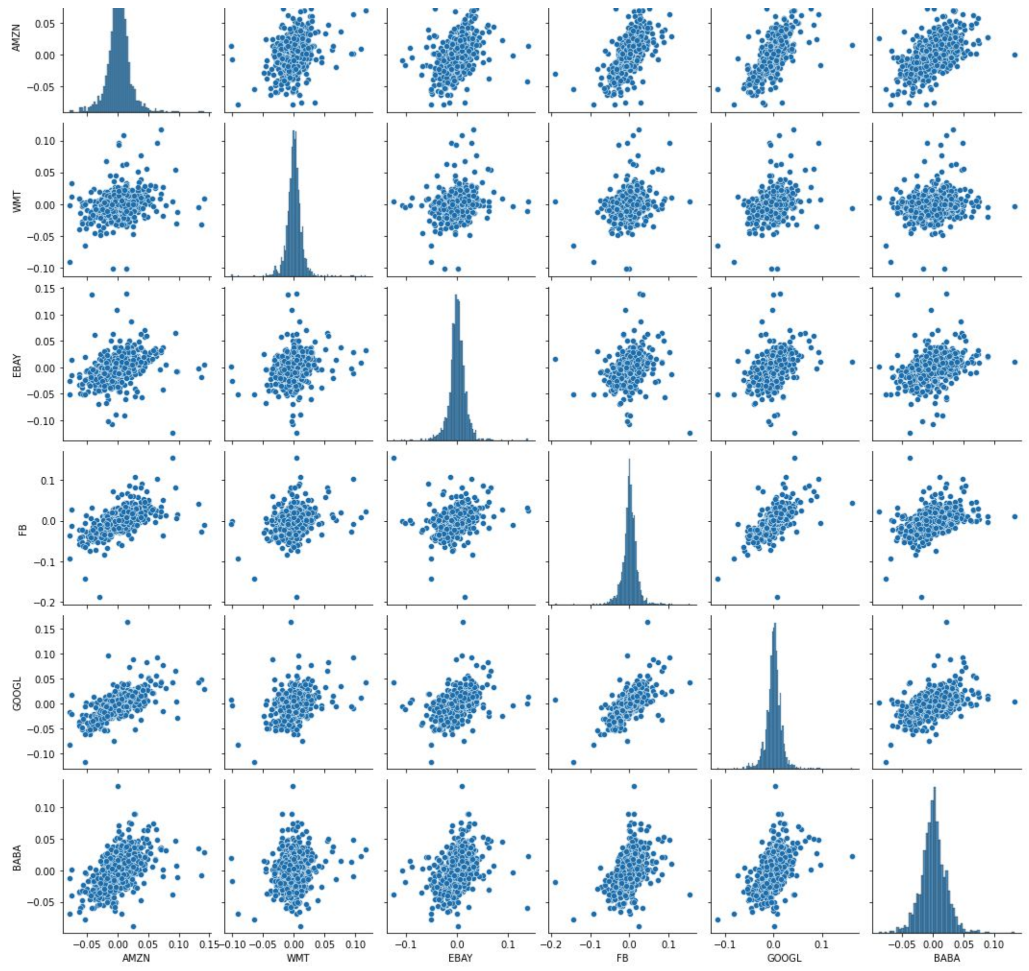
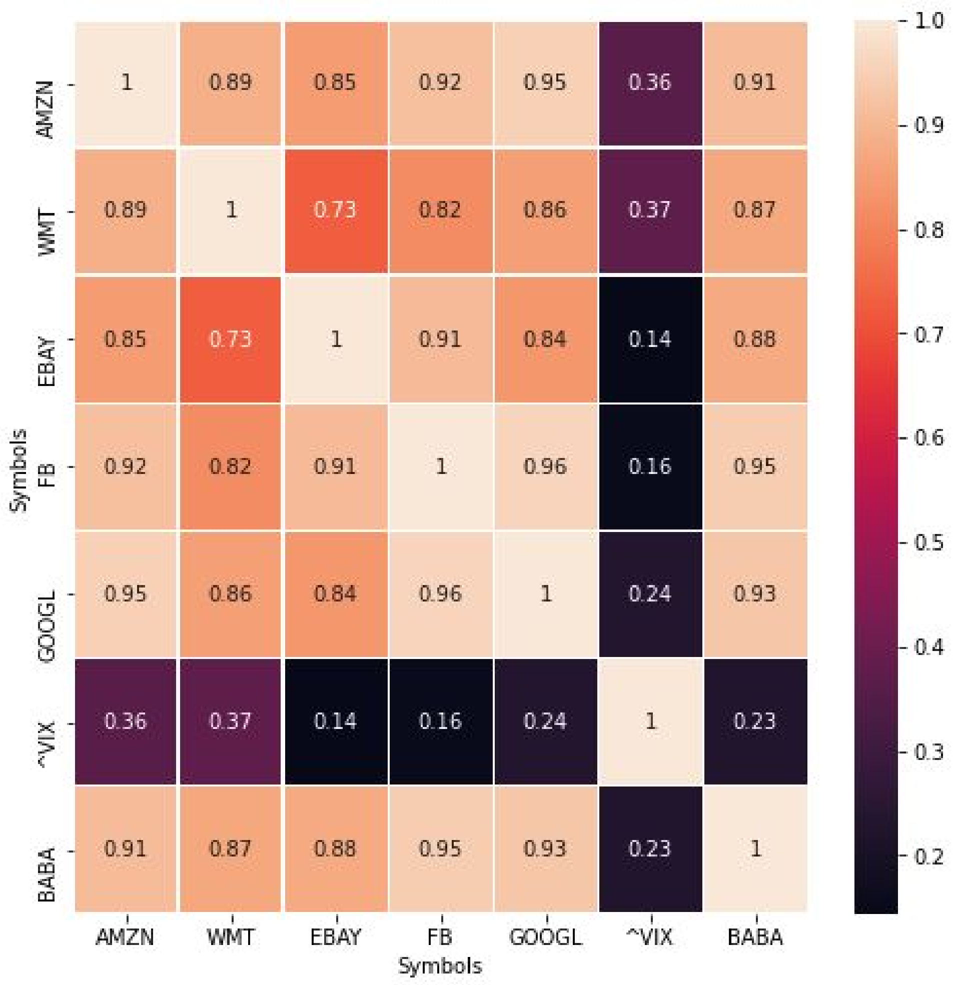


Figure 12.. distribution map

Based on the above figure 12. The team is able to find that all five companies have positive linear relationships with amazon, especially for Facebook and Google. The reason may be because Facebook, Amazon, and Google have more cooperation and less or no competition than the relationship between Amazon with other companies. And all those three companies are giant technology companies in the United States technology field. Walmart has a less positive relationship with amazon based on observation from the above figure. The reason is that Walmart and Amazon have more competition than corporations, and Walmart also does not in the technology field, which means any effective factors in the technology field will not affect the Walmart stock.



	AMZN	WMT	EBAY	FB	GOOGL	^VIX	BABA
AMZN	1	0.89	0.85	0.92	0.95	0.36	0.91
WMT	0.89	1	0.73	0.82	0.86	0.37	0.87
EBAY	0.85	0.73	1	0.91	0.84	0.14	0.88
FB	0.92	0.82	0.91	1	0.96	0.16	0.95
GOOGL	0.95	0.86	0.84	0.96	1	0.24	0.93
^VIX	0.36	0.37	0.14	0.16	0.24	1	0.23
BABA	0.91	0.87	0.88	0.95	0.93	0.23	1

Figure 13. heat map

The heat map shows the same results as the above distribution map, but more clearly than a distribution map. According to the heat map, the team is able to see the relationship between each stock and index more directly. Based on the above figure, the team is able to see all of the companies the team selected has a very tight relationship with amazon. And as expected, the VIX has a negative relationship with amazon. From the above figure. Google has the highest index with Amazon, which is 0.95. Facebook has the second closest positive correlation, and Alibaba is third.

Prediction Models Training

Linear Regression Model

The first model team used is the Sklearn Linear Regression model. In general, the linear regression model is used to solve some problems like "y = B0 + B1 * x". When a problem has more than one variable (Xs) in higher dimensions, the line is called a plane or a hyper-plane. Therefore, the representation in the form of the equation and the specific values used for the coefficients. In linear regression models, there is one important operation team used for this project, it is called "Gradient Descent" and it works by starting with random values for each coefficient. And the explanatory of this theory is When there are one or more inputs, Linear regression uses a process of optimizing the coefficient values by iteratively minimizing the error of the model on the training data. In this project, the sum of the squared errors is calculated for each pair of input and output values. The process is repeated until a minimum sum squared error is achieved, or no further improvement is possible.

As in the data section mentioned, the team uses 80% of total data in training and 20% data in testing and uses the sklearn built-in function to do linear regression. The core code for the linear regression model team used are shown below.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

logreg = LinearRegression()
logreg.fit(x_Train, y_Train)

y_train_predict = logreg.predict(x_Train)
y_test_predict = logreg.predict(x_Test)
```

Figure 14. Core code for Linear Regression Model

QUADRATIC REGRESSION MODEL

The team also used polynomial regression in this project as the second machine learning model to predict the daily close price of Amazon stock. The mechanism of polynomial regression is similar to linear regression. The difference between polynomial and linear regression is the way to expressing the equation as y = B0 + B1*x, the expression is expressed as y = B0 + B1*x + B2*x^2 (2-degree polynomial, x^3 in 3-degree polynomial). This is still considered to be a linear model as the coefficients/weights associated with the features are still linear. x^2 is only a feature. However, the curve that teams are fitting is quadratic in nature.

The team uses the built-in function `polynomialfeature` from sklearn package. The team developed both a 2 degree and 3 degree polynomial regression model. The codes below show the core code on both 2 degree and 3 degree polynomial regression. `PolynomialFeatures(2)` creates a 2 degree polynomial model and `PolynomialFeatures(3)` creates a 3 degree polynomial model. Both models are trained on the same training dataset x_train and y_train.

```
clfpoly2 = make_pipeline(PolynomialFeatures(2), Ridge())
clfpoly2.fit(x_Train, y_Train)

clfpoly3 = make_pipeline(PolynomialFeatures(3), Ridge())
clfpoly3.fit(x_Train, y_Train)
```

Figure 15. Core code for Quadratic Regression Model

Deep Neural Network Model

The third model team developed for this project is the deep neural network prediction model. Compared to the Linear Regression model and quadratic regression model, the deep neural network can solve the problem more globally, and it can also draw conclusions and predictions depending on the information supplied and the desired result.

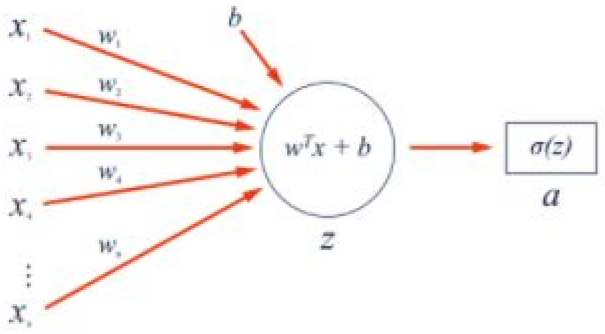


Figure 16. Activation function

A deep neural network consists of the Input Layer, the Hidden Layer(s), and the Output Layer. Connections between neurons are associated with weight, and it dictates the importance of the input value. Neurons apply an activation function on the data to "standardize" the neuron's output. Iterating through the data set and comparing the outputs will produce a Cost Function, indicating how much the Model is off from the real outputs. After each iteration through the data set, the weights between neurons are adjusted using Gradient Descent to reduce the cost function. For this Project, the team uses the Tensorflow package to implement Deep Neural Network in our Project.

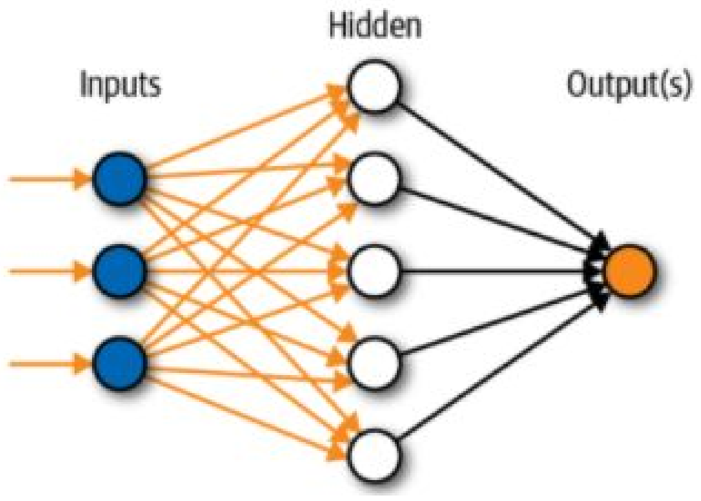


Figure 17. Neural Network

The core code team wrote for the neural network is shown below. In this model development, the team built many deep neural networks and they found the best one based on final test results. The best neural network has 9 layers and each layer has 1 output layer. From layer 1 to layer 4, the number of neurons increased and decreased to 1(output) from the fifth layer to output layers.

```
DNNmodel = Sequential()
DNNmodel.add(Dense(64, activation='relu' , input_shape=(7,)))
DNNmodel.add(Dense(128, activation='relu'))
DNNmodel.add(Dense(256, activation='relu'))
DNNmodel.add(Dense(512, activation='relu'))
DNNmodel.add(Dense(512, activation='relu'))
DNNmodel.add(Dense(256, activation='relu'))
DNNmodel.add(Dense(128, activation='relu'))
DNNmodel.add(Dense(64, activation='relu'))
DNNmodel.add(Dense(1))
DNNmodel.compile(loss='mean_squared_error', optimizer='adam')
```

Figure 18. Core code for deep neural network model

In the deep machine learning model, the team uses Relu as the activation function, mean squared error as loss function and Adam as optimizer. Furthermore, the model is trained for 20 epochs with 32 batch sizes.

As the data section mentioned, in order to improve DNN's accuracy, the team first normalizes the data before input implementing the Neural Network. For the method of data normalization, the team uses the traditional normalization method, which normalizes each number in an array by the maximum and minimum value. I.e. y_normalized = y - min / (y_max - y_min). Thus, each number is normalized between 0.0 and 1.0. After model training, the team used the RMSE, confusion matrix and prediction vs. actual plot to evaluate the model. The result of each model is explained in the Results section.

Long Short Term Memory Model

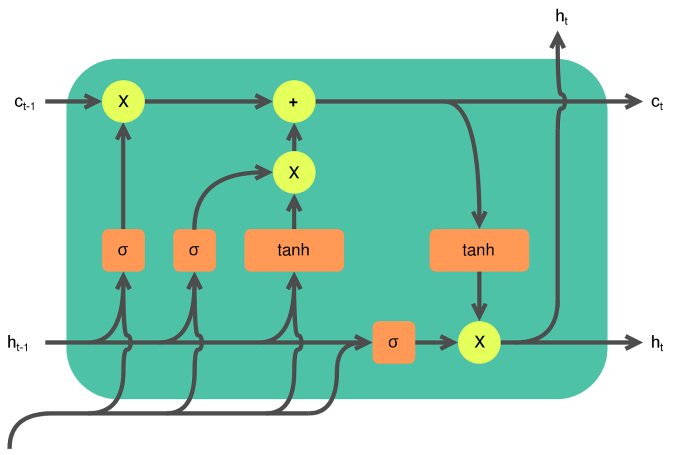


Figure 19, LSTM processing

The last model team developed is the Long Short Term Memory (LSTM) model. The LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior. The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to flow along with it unchanged. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to let information through optionally. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means nothing through, while a value of one means all information through. LSTM Algorithms are very powerful in sequence prediction problems because they're able to store past information. This is important in our case because the previous price of a stock is crucial in predicting its future price. Before implementing LSTM, we normalize our data the same way we did in DNN and preprocessed our data into a 60 time step. Each input is a NumPy array with shape (60, 7) and output with shape(1, 1).

```
X_train = []
y_train = []
for i in range(60, 1185):
       X_train.append(trainning[i-60:i, 0:7])
       y_train.append(trainning[i-1, 7])
X_train, y_train = np.array(X_train), np.array(y_train)
X_train = np.reshape(X_train, (X_train.shape[0],   X_train.shape[1],7))
```

Figure 20. Code for LSTM model

Since the team uses 80 percent of the total data for training, the final input nuympy array size is (1125, 60, 7). The preprocessed data are then input into our LSTM mode.

```
model = Sequential()
model.add(LSTM(units = 256, return_sequences = True, input_shape = (60,7)))
#model.add(Dropout(0.2))
model.add(LSTM(units = 512, return_sequences = True))
#model.add(Dropout(0.2))
model.add(LSTM(units = 512, return_sequences = True))
#model.add(Dropout(0.2))
model.add(LSTM(units = 512))
model.add(Dropout(0.2))
model.add(Dense(units = 1))
model.compile(optimizer = 'adam', loss = 'mean_squared_error')
model.fit(X_train, y_train, epochs = 100, batch_size = 32)
```

Figure 21. Core Code for LSTM model

In the LSTM model, the team chose the Adam optimizer and mean_squared_error as loss function. The team tries to add a few dense layers before the output layer; however, the extra dense layers do not decrease the loss. After training the model, we used the RMSE, confusion matrix and prediction vs. actual plot to evaluate the model. The result of each model is explained in the Results section.

Results and Analyze

Summary of Results

	Linear Regression	Quadratic Regression (2 Degree)	Quadratic Regression (3 Degree)	Deep Neural Network (adjusted)	Long Short Term Memory (adjusted)
RMSE	51.4852	71.7624	746.2417	15.56482	50.096471

Table 1. Summary of Results

Based on the above results table, the best model to predict Amazon's daily close price is deep neural network, it has the smallest RMSE than all other methods. The worst model in this project is the quadratic regression model.

Analyze

In this project the team developed three ways to evaluate our model's accuracy, which are RMSE (Root mean square error), the actual price vs. predicted price plot and a confusion matrix. The Root Mean Square Error is calculated by using the Sklearn package:

```
mse = mean_squared_error(y_true, y_test_predict)
rmse = np.sqrt(mse)
```

Figure 22, Code for RMSE calculation

RMSE measures how much the predicted values vary from the actual values. The predicted vs. actual plot is a visualized tool used to observe the model's performance. The actual values are data models acquired. The predicted values are calculated by inputting testing data into the trained model. The confusion matrix is another way to measure the dataset and prediction values. Instead of measuring how far the predictions deviated from the actual values, to convert the actual values into binary classifications. The logic the team uses is that if the daily close price is lower than the input close price (which is one day before the result), the program marks the result as ‘0’. For example, 0 means Amazon's daily close price on January 2nd (Actual y value) is lower than Amazon’s price on January 1st (input value). On the contrary, if the result close price is higher than the input daily close price (which is one day before the result), the program will mark the result as ‘1’. In conclusion, ‘0’ stands for the closing price drops compared to yesterday’s close price. ‘1’ stands for the opposite situation. The program also labels the predicted values with the same measurement (compared to yesterday’s close price). ‘0’ stands for the predicted close price drops compared to yesterday’s close price. ‘1’ stands for the predicted close price raises compared to yesterday’s close price. Then the team created a confusion matrix to measure how accurate our model performs according to such criteria.

```
testmatrix = []
truematrix = []
for i in range(len(count_table)-1):
        if count_table.iloc[i][0] < count_table.iloc[i][7]:
                truematrix.append(1)
        else:
                truematrix.append(0)

        if count_table.iloc[i][0] < y_test_predict[i]:
                testmatrix.append(1)
        else:
                testmatrix.append(0)
y_actu = pd.Series(truematrix, name='Actual')
y_pred = pd.Series(testmatrix, name='Predicted')
df_confusion = pd.crosstab(y_actu, y_pred)
```

Figure 23. Code for 0/1 function

The team's motivation to use another evaluation method is that RMSE measures the deviation from the actual price; however, the deviation can be either positive or negative. No matter how small the RMSE is, positive means earning and negative means losing money, and RMSE can not express the positive and negative effects. Thus, the team decided to use that confusion matrix and RMSE to evaluate our model.

Linear Regression Model

The linear regression model performs pretty well on the testing dataset. The model achieves an RMSE (Root mean square error) of 21.86 on the training dataset and 51.5 on the testing dataset. Input testing x values calculate the predicted price into the model by using sklearn and keras's model.predict function: `y_test_predict = logreg.predict(x_Test).`

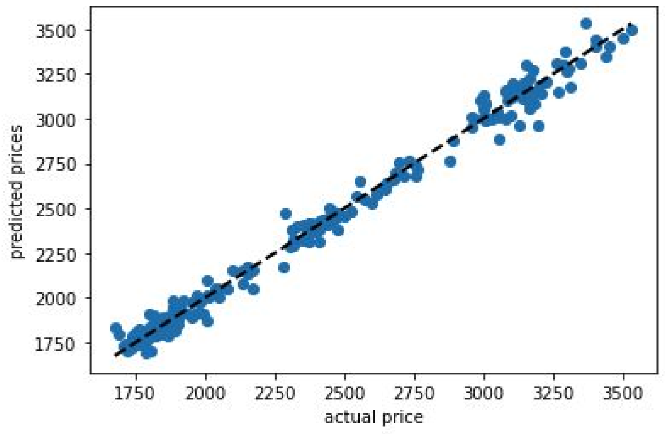


Figure 24. Linear plot for Actual and Predicted results

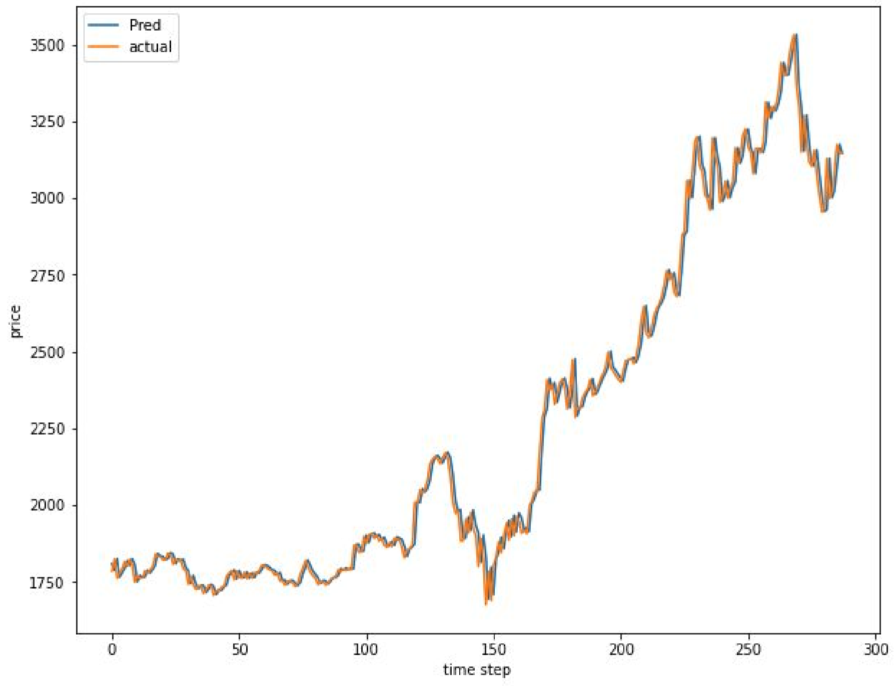
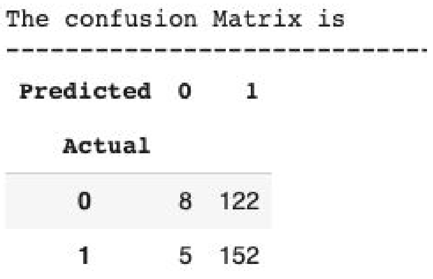


Figure 25. Linear Regression prediction for next 288 days Amazon daily close prices.

The blue line indicates the prediction and orange line indicates the actual data. Based on the graph, our team concludes that linear regression performs very well on the test dataset.



```
The confusion Matrix is
-----------------------------
Predicted  0    1
   Actual
   0       8  122
   1       5  152
```

Figure 26. Confusion matrix for linear regression

Figure 26 shows the Confusion matrix. Although the RMSE is small and the Prediction vs. actual price plot seems perfect, the confusion matrix shows the model is not very accurate in predicting the rise or drop behavior of the Next day's Closing Price.

Polynomial Regression Model

The Polynomial regression model does not perform well on the testing dataset. The two-degree polynomial model achieves an RMSE (Root mean square error) of 71.76 on the test dataset. And the 3-degree polynomial model achieves 746.24 on the test dataset. Input testing x values calculate the predicted price into the model by using sklearn and keras's model. predict function: `forecast_set = clfpoly.predict(x_Test)`. The team concludes that polynomial regression is not workable in stock prediction problems.

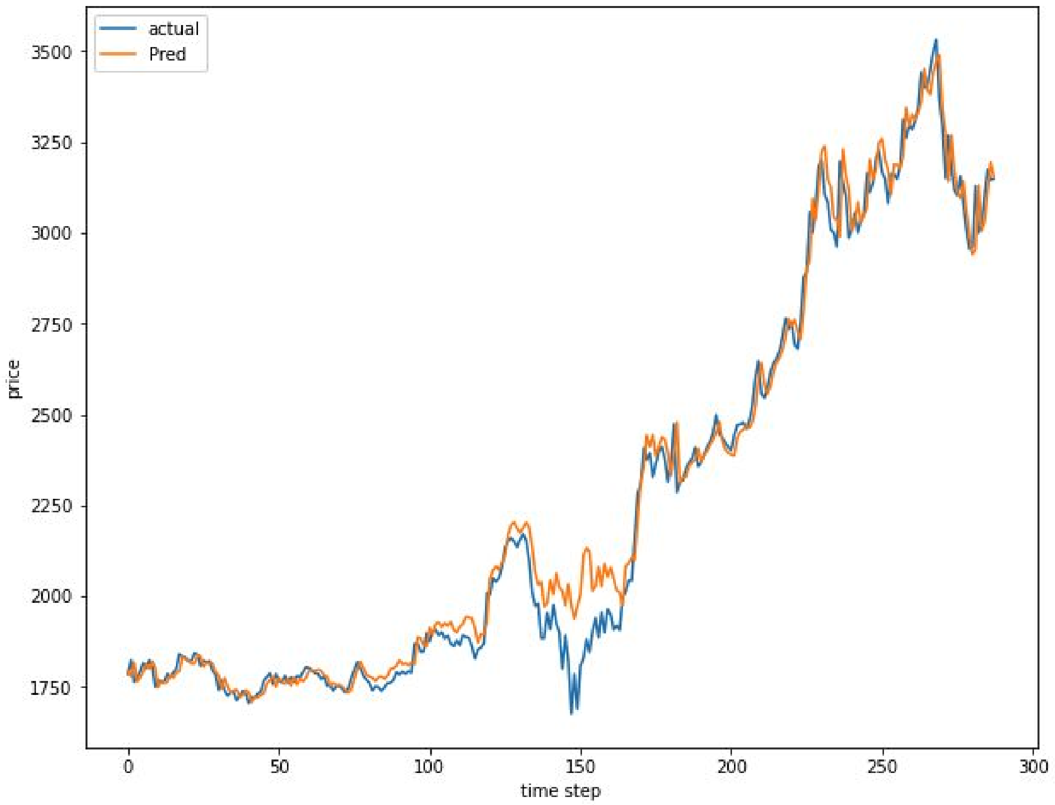


Figure 27. Polynomial Regression prediction for next 288 days Amazon daily close prices.

The orange line shows the prediction and blue line shows the actual data. The 2 degree polynomial only captures the general trend of the test data but performs worse than the linear regression model.

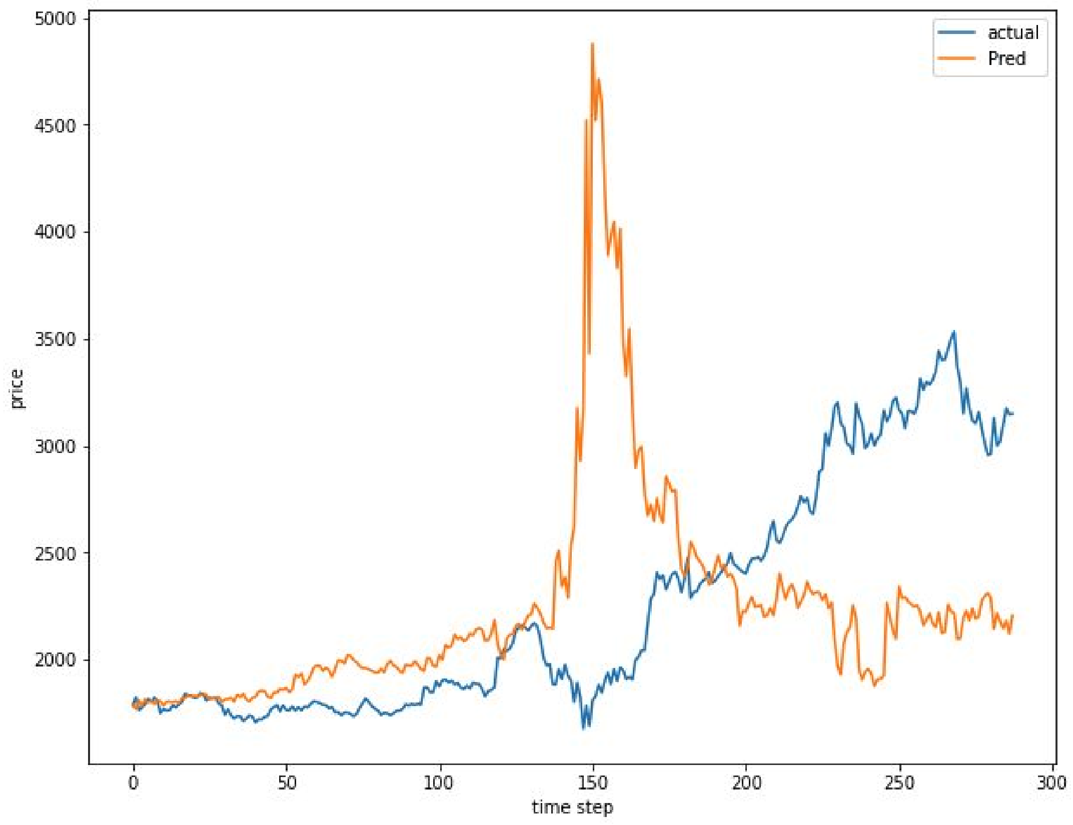


Figure 28. Polynomial Regression prediction for next 288 days Amazon daily close prices.

The orange line shows the prediction, and the blue line shows the actual data. The 3-degree polynomial is the worst prediction among linear regression models. The prediction deviates from the actual dataset. Thus our team concludes that high degree polynomials can not capture the data trend in our scenario.

Deep Neural Network Model

The deep neural network model performs well on the testing dataset. The model achieves a RMSE (Root mean square error) of 6.34 * 10^-5 on the training dataset. Since all the training data for DNN is normalized, the RMSE is small. The RMSE on the testing set is 41.26, which is lower than the Linear Regression model. The predicted price is calculated by input testing x values into the model by using sklearn and keras's model.predict function:

```
predicted_stock_price = DNNmodel.predict(X_test).
```

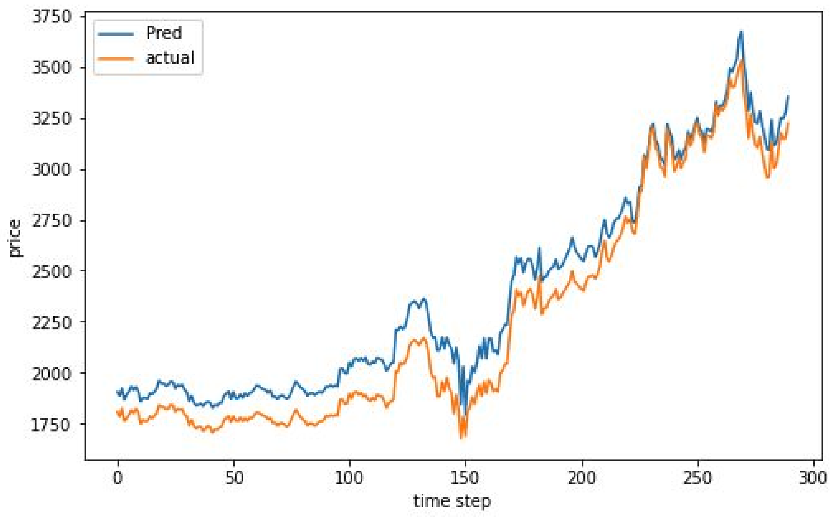
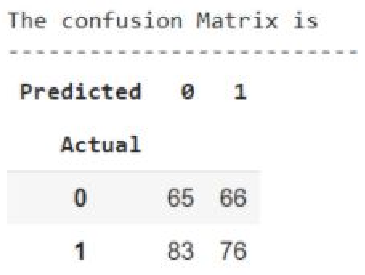


Figure 29. DNN model prediction for next 288 days Amazon daily close prices before adjust



```
The confusion Matrix is
--------------------------
```

Predicted	0	1
Actual		
0	65	66
1	83	76

Figure 30. Confusion matrix for DNN model

Although the prediction vs. actual plot look similar, the model is not accurate in predicting the drop and rise of the stock price.

The orange line shows the prediction, and the blue line shows the actual data. Although the prediction is lower than the actual dataset, our team concludes that the DNN model successfully captures the daily price change. Furthermore, the team developed a revision factor function to modify the model's prediction based on our conclusion.

As the figure 29 shown, Based on the plot of the deep machine learning prediction model, the team found the DNN model does not provide accurate results on daily close prediction. However, deep learning models do make a very good prediction on stock movement. Based on this, the team decided to develop a function that could adjust the prediction which would make better predicted results.

Method of Prediction Improvement Function (revision factor)

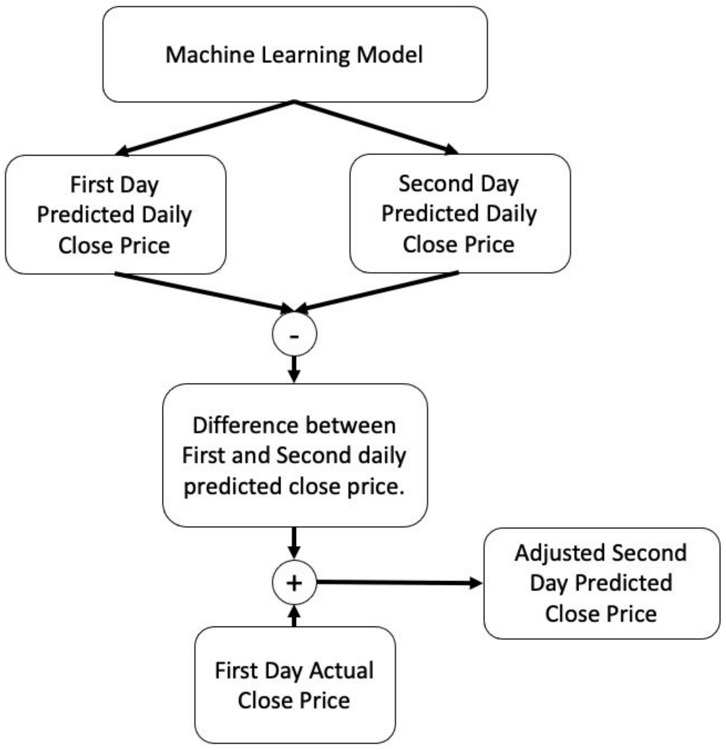

Machine Learning Model
First Day Predicted Daily Close Price
Second Day Predicted Daily Close Price
-
Difference between First and Second daily predicted close price.
+
Adjusted Second Day Predicted Close Price
First Day Actual Close Price

Figure 31. Adjustment Function Process Flow

Since the deep learning model is able to provide an accurate stock movement. To improve the prediction results, the team developed a function that could combine the prediction with actual daily close price which to improve the prediction on next day's daily close price. The principle of the adjustment function is based on predicted stock movement between the first and second day predicted close price, then added the movement into actual first day close price. The output of the function are adjusted second day predicted close price.

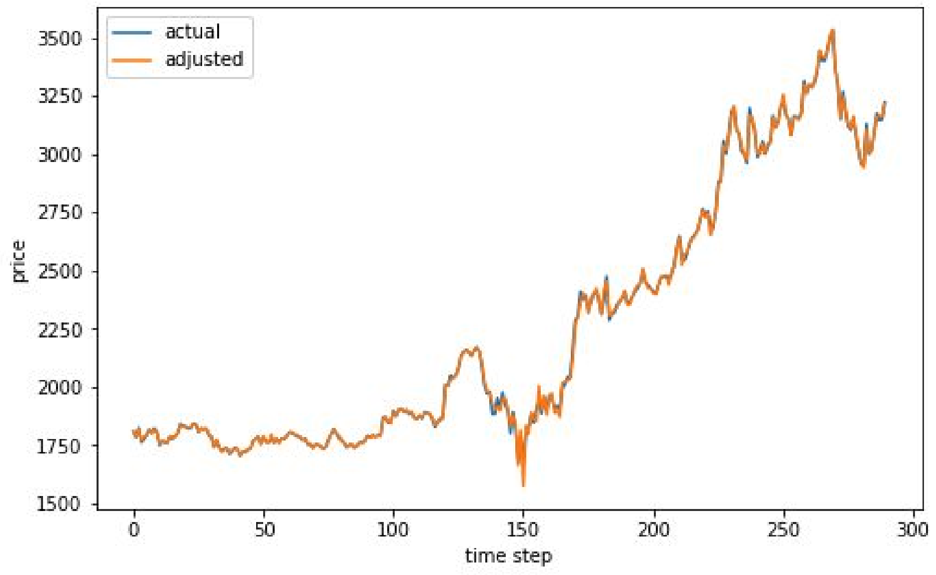


Figure 32. DNN model prediction for next 288 days Amazon daily close prices

Above figure 31 shows the improved prediction for Amazon’s daily close price for next 288 days. The orange plot is an improved prediction of close prices and blue plot is actual close price of Amazon stock. In RMSE, the previous prediction rmse is 137.92. The rmse of improved prediction is 15.56. Compared with other prediction model teams developed, the deep neural network machine learning model with adjustment function could provide the best daily price prediction. The advantage of using this adjustment function is that even sometimes trained a prediction model that has a high rmse, the adjustment could always improve the model and keep the RMSE less than 20.

LSTM (Long Short Term Memory) Model:

The LSTM model performs well on the testing dataset. The model achieves a RMSE (Root mean square error) of 1.76*10^-4 on the training dataset and 104 on the testing dataset.
The predicted price is calculated by input testing x values into the model by using sklearn and keras's model.predict function: `predicted_stock_price = model.predict(X_test)`.

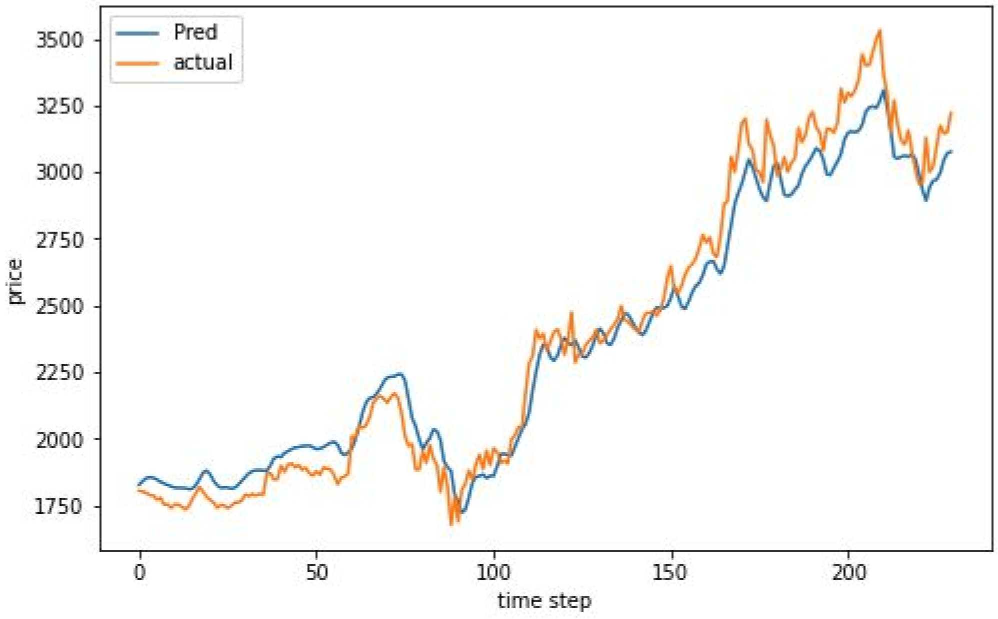


Figure 33. LSTM model prediction for next 288 days Amazon daily close prices before adjust

The orange line shows the prediction, and the blue line shows the actual data. From the plot, the LSTM model captures the trend of the testing dataset, but the actual values deviate from the true value. Our team applies the same revision method to the LSTM result. However, similarly with deep neural networks models, even the accuracy of daily close price prediction is low, the long short-term memory model provides better accuracy for stock movement prediction. Based on the output from the long short term model, the team decided to use an adjustment function and to see if it can improve results of the daily close price prediction.

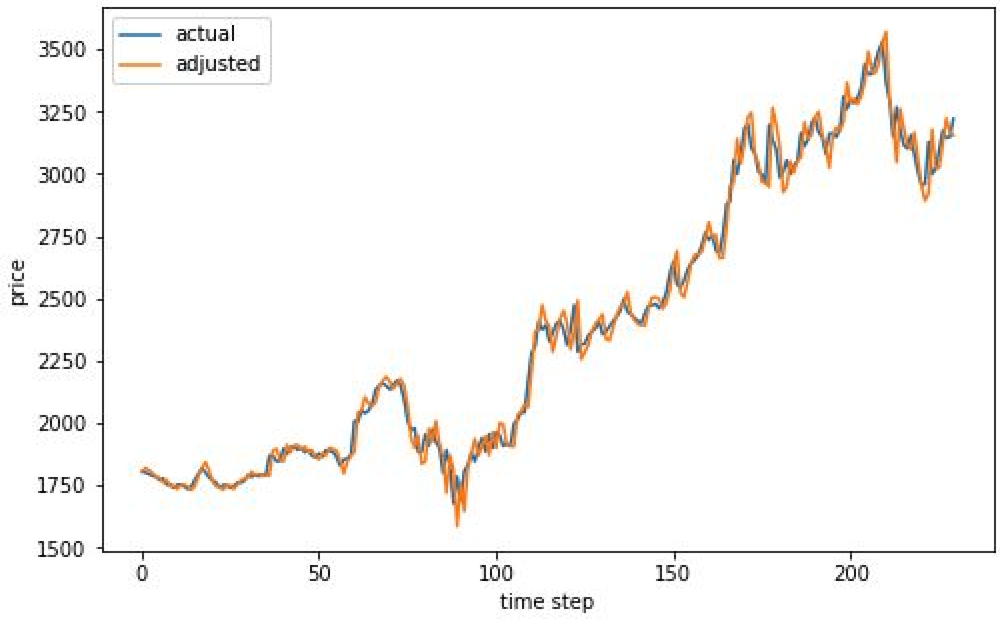
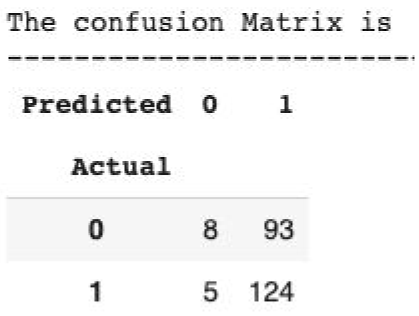


Figure 34, LSTM model prediction for next 288 days Amazon daily close prices after adjust

As figure 34 shown, the adjustment function improved the daily close price prediction. The previous rmse is 104.87 and the rmse of the improved prediction model is 50.096471. The reason the improved prediction model does not have lower rmse than DNN, is because the DNN model provides a better stock movement prediction than LSTM does.



```
The confusion Matrix is
-------------------------
```

Predicted	0	1
Actual		
0	8	93
1	5	124

Figure 35. Confusion matrix for LSTM model.

Although the prediction vs. actual price plot seems worse than that of DNN, the LSTM model is more accurate in predicting the drop and rise behavior.

CONCLUSION & LESSONS LEARNED

This project aims to develop a model that can predict the daily close price of Amazon. Compared to four machine learning models, the deep learning network model with the prediction improvement function is the best method to predict very accurate stock daily price and stock movement. This method can provide high accuracy in Amazon's daily close price prediction. However, the team realized that even a deep neural learning network with an improvement function can not predict 100% correctly for a second daily close price. The reason may be because there are more other factors affecting the stock, such as the influence of market news or natural disaster events like a pandemic disease. But those problems can be the future goal of this team. In the future, the team wants to try to develop more advanced machine learning models which include more functions such as news analysis and it can predict the more accurate stock daily price.

After the successful completion of this project, the team learned a lot. One of the biggest lessons that the team learned is knowledge about how to develop the machine learning model to solve real-life issues. During this project, the group learned how to build machine learning from scratch to final model development by learning from data science for mechanical system courses and self-learning. The team is excited to solve the real-life problem by using learned knowledge in this class and project.

FUTURE IMPROVEMENT:

Integrating News Factor

The volatility of news is one of the important factors that makes stock price prediction difficult for typical machine learning algorithms because news can directly or indirectly influence stock price of a company. In the future, the team will try to develop the function which could integrate the influence of news into our machine learning model. To achieve this goal in future, the idea of this function is, in the first step, the team will build a web scraper that downloads all stock news related to Amazon's price in the past five years from websites such as MarketWatch. Then the team will develop a model that converts each News into a number. The news, which is text, can be converted into an integer list using a word embedding and vectorization. The vectorized text is input to a machine learning model (such as bert) as the x value. The y value of the model is some number that can quantify the influence of news on the stock price. After collecting the x value and y value, the team can train the machine learning model that maps each news into a number. The output numbers can be used as another column in the DNN , Linear Regression or LSTM model above, then retrain the model. We hope that the model can be more sensitive to News's influence by integrating the news factor into the training dataset.

Feature Engineering

Features in the training dataset decide the performance of machine learning models. In this project, the team used stock prices of different companies (google, ebay, Alibaba, Facebook, etc). In the future, the team will try to add more features, for example to use the difference between Amazon's Daily High price and Low price, to use the ratio between High price and Low price, or the team could use the difference of the prices between Amazon and its other competitors.

Furthermore, all the training data can be normalized by different normalization methods. In the project, the team use the simple normalization equation $X = (X - Xmin) / (Xmax - Xmin)$. In the future, the team could try different mathematical normalization theories on the training dataset. We believe the model can be optimized by adding more useful features and also using better encoding techniques.

Reference

1.Linear Regression for Machine Learning, Brownlee

https://machinelearningmastery.com/linear-regression-for-machine-learning/

2.Basic classification: Classify images of clothing : TensorFlow Core
https://www.tensorflow.org/tutorials/keras/classification

3.Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning, Jeff Heaton
ORCID: orcid.org/0000-0003-1496-40491
https://link.springer.com/article/10.1007/s10710-017-9314-z

4.Understanding LSTM Networks, Cloah's Blog, Aug 27, 2015

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

5. Determining Market Direction With VIX, Summa, Aug 19, 2020
https://www.investopedia.com/articles/optioninvestor/03/091003.asp

6. Polynomial Regression, Agarwal, Oct 8, 2018

https://towardsdatascience.com/polynomial-regression-bbe8b9d97491