Investment and Trading Capstone Project

Build a Stock Price Indicator

by

Felipe Martinez

to

Udacity

Nanodegree Program

Machine Learning Engineer

November 2018

CONTENIDO

1 DEFINITION

Project Overview

Investors make their decisions to buy or sell stock of a company relying on their analysis of stock market indicators, in order to make the best investment. Therefore, observing the behavior of the stock price indicator every day will always be in tune with the current trend.

For an investor to look for the path that leads him to predict the behavior of the stock price would give him the knowledge that would place him in advantage to make a satisfactory investment. However Burton Malkiel, in his influential 1973 work, A Random Walk Down Wall Street and Warren Buffett, respectively state that:

“that stock prices could therefore not be accurately predicted by looking at price history. As a result, Malkiel argued, stock prices are best described by a statistical process called a "random walk" meaning each day's deviations from the central value are random and unpredictable” (Burton Malkiel). [1]

Stop trying to predict the direction of the stock market, the economy, interest rates, or elections (Warren Buffett). [2]

In the last two decades, the desire to make predictions about the actions of values has been the subject of study. Prediction methodologies are divided into three broad categories: [1]

1. **Fundamental analysis**
2. **Technical analysis**
3. **Machine learning**

The technology in these categories has an important role because of this depends speed, consistency, timeliness, security and trust, such characteristics can support to observe and analyze thousands of transactions every day.
With the advent of the digital computer, stock market prediction has since moved into the technological realm. For this project the main objective is Implement stock predictor and builds a model of stock behavior using a method called Long short-term memory (LSTM) which are units of a recurrent neural network (RNN). Today is considered the best model to perform this task.

Problem Statement

The challenge of this project is to build a model to predict the adjusted closing price of the stock price, taking the daily operations data in a certain range of dates as input, and generate projected estimates for certain consultation dates.

For this project will be used a
Long Short Term Memory networks – called “LSTMs” to predict the Adjusted Closing price of the S&P 5002 using a historical dataset prices

GOALS

1. Explore stock prices;
    - We will understand our data set to know what features it has and how it would be your preprocessing.
2. Data Preprocessing;
    - We will manipulate the elements of the data set to produce meaningful information.
3. Implement Deep Learning model using LSTM.
    - will perform the task of predicting the stock price.
4. Visualize the results;

- We can see the graphs of the predictions made by the model.

5. Evaluate model;
   - It will be checked if the model has a good performance.
7. Improve model;
   - Various changes will be made to improve the model and make better predictions.

Metrics

The measures of performance for this project will be the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are standards for measuring error for machine learning predictions and will offer consistency across all the studies developed.

MAE is a quantity used to measure how close forecasts or predictions are to the eventual outcomes. MAE is more robust to outliers since it does not make use of square.

Root Mean Squared Error (RMSE) calculated as the difference between predicted and actual values of the target stock ticker at daily market close. The use of the RMSE should be avoided when there is atypical values (large measurement errors). These values will have a strong effect on the as they are squared. Consider using these two metrics important to see that the results were consistent

2 ANALYSIS

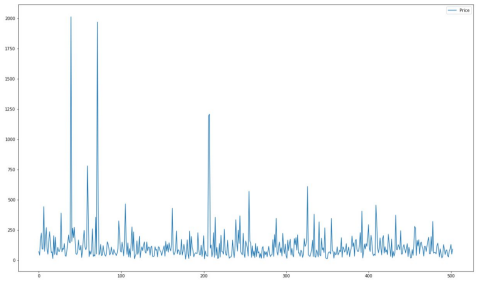
Data Exploration

The S&P500 index is the acronym of Standard & Poor's 500, which includes the 500 most representative companies in the New York Stock Exchange. The importance of this index, together with the Dow Jones, is that they participate in the most important financial market of the planet: Wall Street. It is one of the indexes most used by institutional investors, because it incorporates one of the largest portfolios of companies. [6]

For this step it was considered important to have as antecedent the symbols that form the S&P 500 list and show it to have a reference of the information. The first step is to obtain the list that makes up this index, Table 1, later we extract the symbols directly from the table to assign to each ticker the value of the price of Adj Close we realize a visualization with the information. Figure 2.

	0	1	2	3	4	5	6	7	8
0	Ticker symbol	Security	SEC filings	GICS Sector	GICS Sub Industry	Location	Date first added[3][4]	CIK	Founded
1	MMM	3M Company	reports	Industrials	Industrial Conglomerates	St. Paul, Minnesota	NaN	0000066740	1902
2	ABT	Abbott Laboratories	reports	Health Care	Health Care Equipment	North Chicago, Illinois	1964-03-31	0000001800	1888
3	ABBV	AbbVie Inc.	reports	Health Care	Pharmaceuticals	North Chicago, Illinois	2012-12-31	0001551152	2013 (1888)
4	ABMD	ABIOMED Inc	reports	Health Care	Health Care Equipment	Danvers, Massachusetts	2018-05-31	0000815094	1981

	Symbol	Price
0	A	70.800003
1	AAL	41.500000
2	AAP	169.789993
3	AAPL	224.949997
4	ABBV	94.139999



a) *b)*

Table 1: Image from jupyter notebook file "ML Get list S & P500 and price"

Figure 2: In the left image a) we have a list of the first 5 symbols with their Adjusted Closing Price, In the image on the right b) is the graph of the S & P 500 list at "2018-09-27".

In the graph of figure 2, we see in the first 100 symbols the prices that are with a value close to 2000 that correspond to the AMZN ticker (Amazon), with price of $ 2,012.97, following BKNG (Booking Holdings Inc) with price of $ 1,969.33, for the last we have GOOG and GOOGL (Alphabet Inc) with price $ 1,194.64 and $ 1,207.35 respectively Figure 3.

	Symbol	Price
39	AMZN	2012.97998

a)

	Symbol	Price
71	BKNG	1969.339966

b)

	Symbol	Price
206	GOOG	1194.640015
207	GOOGL	1207.359985

c)

Figure 3, price of the adjusted close of date "2018-09-27, a) Price of the symbol AMZN (Amazon), b) price of the symbol BKNG (Booking Holdings Inc), c) price symbol GOOG and GOOGL (Alphabet Inc).

Now let's see how this data is composed, what we do to obtain a set of historical data of each symbol is to extract the files from Yahoo Finance and save them in a .csv format. Within each file is the data set as shown in Figure 4.

	Date	Open	High	Low	Close	Adj Close	Volume
0	2013-10-01	314.220001	321.000000	313.649994	320.950012	320.950012	2396400
1	2013-10-02	318.040009	321.730011	317.519989	320.510010	320.510010	2217400
2	2013-10-03	320.390015	322.920013	313.019989	314.760010	314.760010	2674800
3	2013-10-04	315.130005	319.200012	312.619995	319.040009	319.040009	1815000
4	2013-10-07	315.239990	315.339996	309.739990	310.029999	310.029999	2083200

Figure 4. Data set with .csv format of the symbol AMZN (Amazon)

For this project we will focus on the Adj Close indicator, below I will make a brief explanation of what this indicator is about.

An adjusted closing price is a stock's closing price on any given day of trading that has been amended to include any distributions and corporate actions that occurred at any time before the next day's open. The adjusted closing price is often used when examining historical returns or performing a detailed analysis of historical returns. [7]

In the following illustration Figure 5, a graph is used as a historical representation of price behavior by symbolo, for this visualization we use twelve randomly selected symbols. Each symbol shows with a color label the date and the value of the adjusted closing price of that day.

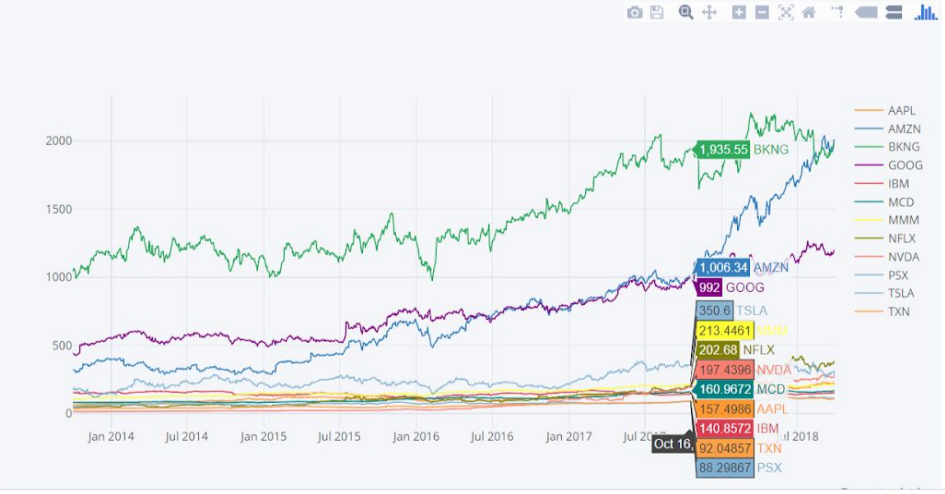


Figure 5, Graph of 12 symbols with the adjusted closing price (Adj Close)

We observe that in the graph the line of the symbol BKNG has a price of 1, 935.55 to Oct 16 of 2017 being the highest value of the other symbols, following below the symbols AMZN and GOOG with 1,006.34 and 993.00 respectively. The three symbols show a growing trend since October 2013, date in which the sample was taken on October 16, 2017, which shows that these symbols appear to be non-stationary, that is, the mean will not be constant for all observations.

Making a description of statistics as shown in the table of figure 6 we see the concept of mean; This statistical indicator can be used to measure the performance of the stock price of a company over a period of days, months or years.

Statistic	Symbols			
	Adj Close			
	AAPL	AMZN	BKNG	GOOG
count	1259.000000	1259.000000	1259.000000	1259.000000
mean	120.831117	757.680930	1469.311152	758.742901
std	37.684647	444.615235	337.092391	213.556093
min	62.609249	286.950012	973.799988	424.076782
25%	93.849827	375.125000	1191.589966	561.269165
50%	110.414680	664.510010	1320.479980	728.580017
75%	149.575882	967.695007	1802.075012	929.220001
max	228.360001	2039.510010	2206.090088	1268.329956

Figure 6. Descriptive statistics del indicador de Cierre Ajustado

Another equally important concept is the standard deviation. This measure is an especially useful tool in investment and negotiation strategies, as it helps to measure market volatility and security, and predicts performance trends. A lower standard deviation is not necessarily preferable. Everything depends on the investments one is making and the willingness to take risks. When dealing with the amount of deviation in their portfolios, investors should consider their personal tolerance to volatility and their overall investment objectives. More aggressive investors may feel comfortable with an investment strategy that chooses vehicles with higher than average volatility, while more conservative investors may not do it. [8] A high standard deviation indicates that stock prices vary greatly over time.

Exploratory Visualization

If you want to make a comparison when we have very variable prices, it becomes a difficult task because the price ranges in the graphs are very long, which means that you can not make a good comparative visualization. So a solution would be that all the values begin at a single point, that is to say that it is at 1.00 from there to start comparing them on an equal basis.

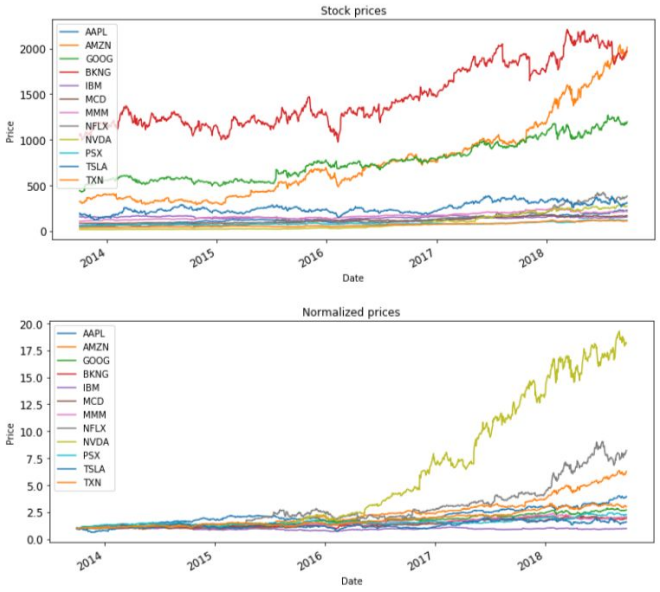


Figure 7, Comparative graph between two conventional graphical visualizations vs normalized graph

In these two comparative graphs we see how there is a behavior of the prices that are around 2.5 without much variation during the first three years, this means that the majority of the prices of each symbol grew around 2.5. While the price of NVDA increased for 2018 up to 20 times its value, this is that the minimum price of 13.77 rose to a maximum of 283.00, which means that this is the company with the highest increase in that period.

Suppose we want to invest some money and we are looking for an opportunity to buy stocks and we have already seen some indicators that give us information about the behavior of the stock price, however, we still do not have much confidence in the best time to buy. For financial operators the key commercial tool is Bollinger bands, created in the early 80's and named in honor of the developer.

Bollinger bands ®; Bollinger Bands represent a key commercial technical tool for financial operators. Bollinger bands are represented by two standard deviations (a measure of volatility) far from the moving average of a price. Bollinger bands allow traders to monitor and take advantage of changes in price volatilities. [9]
John Bollinger suggests using them with two or three non-correlated indicators that provide more direct signals from the market. [10]

In the graph of Figure 9, the stock price is between an upper and lower band along with a simple moving average of 21 days. This means that the standard deviation is a measure of volatility, when the markets become more volatile, the bands widen; During the less volatile periods, the bands contract.

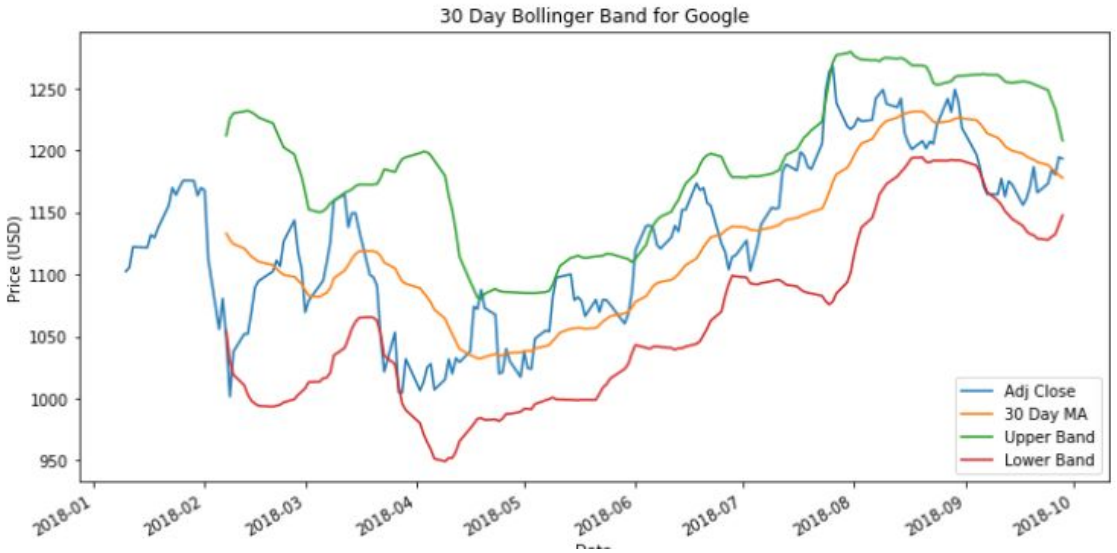


Figure 9, Bollinger Bands chart

Main Components of a Bollinger Bands

- Upper limit: The upper limit is simply two standard deviations above the moving average of a stock's price.
- Middle Band: The middle band is simply the moving average of the stock's price.
- Lower limit: Two standard deviations below the moving average is the lower limit.

Many traders believe that the closer the prices are to the higher band, the more the market overblows and the lower the prices get closer to the lower band, the more the market is oversold. John Bollinger has a set of twenty-two rules to follow), when the bands are used as a trading system (The rules will not be mentioned in this document, for better information see the reference). [10]

To see if our investment has good returns we could use one of the most important statistics used in financial analysis called Daily return. Daily return is simply the daily returns, describes how much went up or how much down on a particular day.

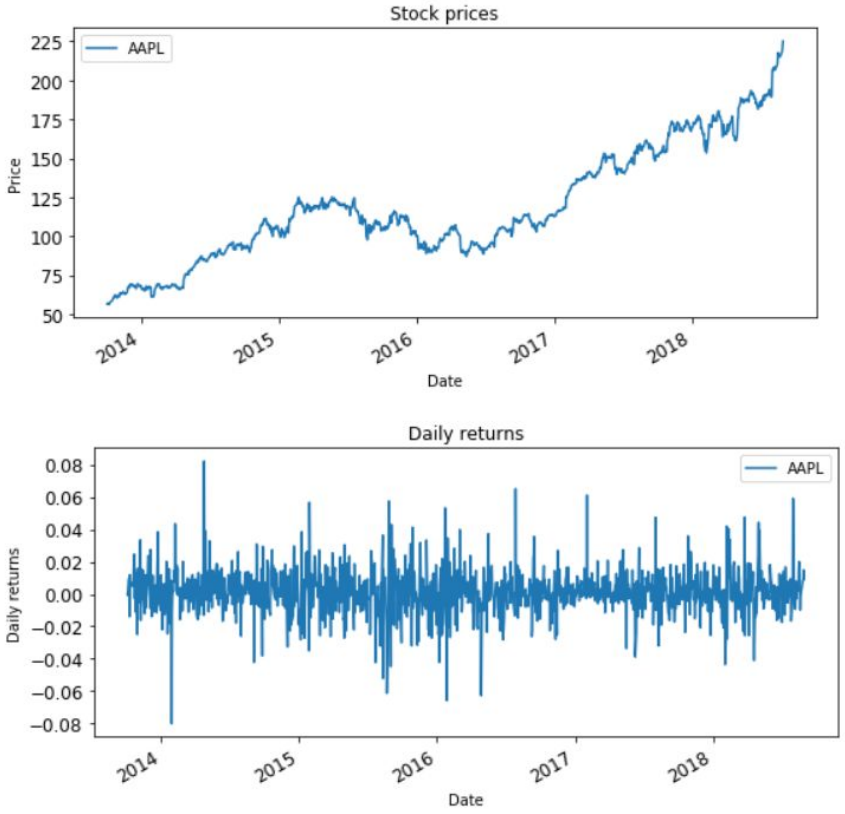


Figura 10, Above graphical representation of Stock price. Below Daily return

A positive result means that the returns exceed the costs. Analysts, therefore, consider investment as a net gain. The opposite type of result, the negative return on investment results, means that the costs outweigh the returns. The analyst, therefore, sees the investment as a net loss.

Another important statistic is called Cumulative return, this statistic shows the total performance in a given period for example, the graph in Figure 11 shows the cumulative return from August to September 2018, which indicates that it had a 12% return in that period.

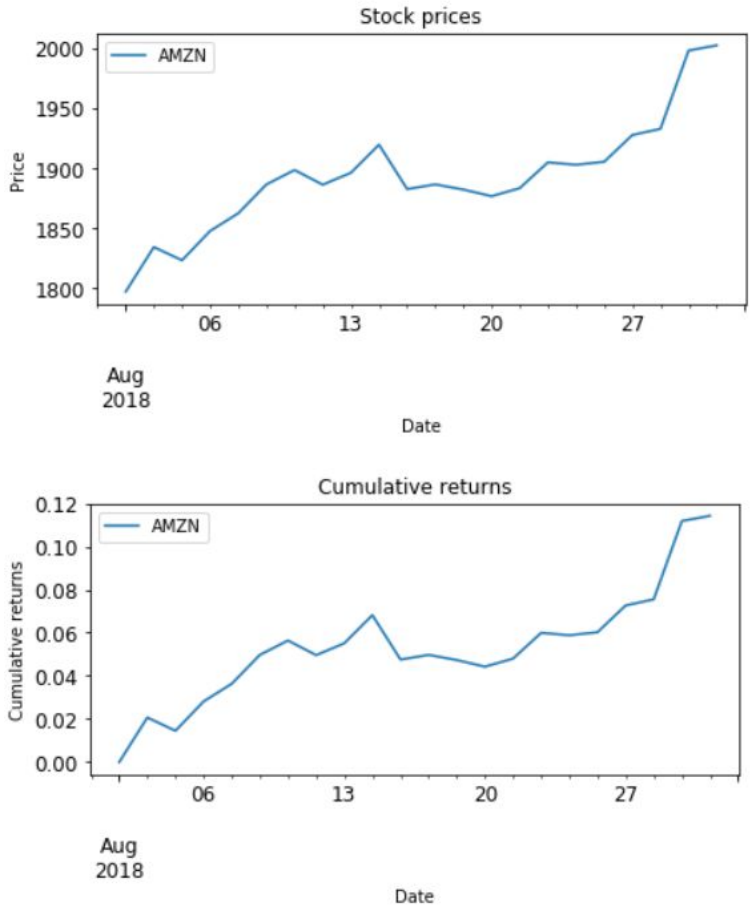


Figure 11, Cumulative return graph from August to September 2018

+

Algorithms and Techniques

The objective of this project is to build a model to predict the adjusted closing price of the stock price, taking the daily operations data in a certain range of data as input, and to generate estimates for certain consultation dates. for this project it was proposed to build the model with a Recurrent Neural Network (RNN).

Recurrent machine learning models can examine the history of a data stream and correctly predict what the future elements of the sequence will be.

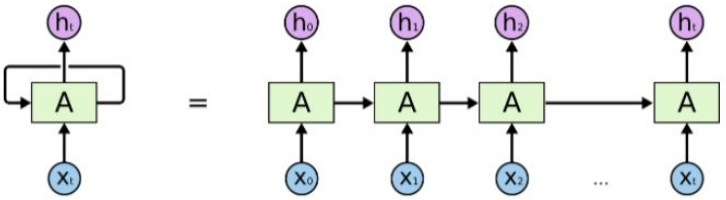


Figura 12, References An unrolled recurrent neural network.

Recurrent Neural Networks: are neural networks that apply the same operation to each element of a sequence of input data and whose output depends both on the input data present and past incorporating a state variant in time. This algorithm is characterized by the memory capacity. [11]

However there is a problem called the "Vanishing gradient problem", it occurs when we try to train a neural network model using gradient-based optimization techniques. The vanishing gradient problem basically means that every time there is a recurrence, the value of the weight decreases, and when multiplied by very small values, decrease very quickly, that is, it is more difficult to update if the gradient has a very low value. [12]

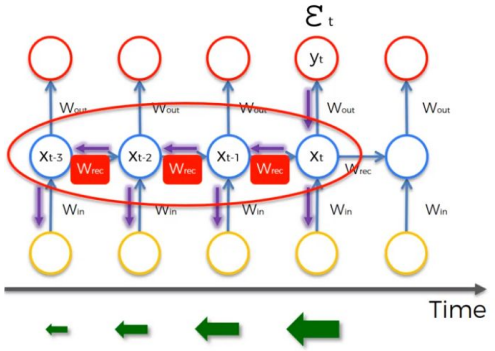


Figura 13,Graphic representation the vanishing gradient problem

One solution to this problem is to introduce an LSTM module (Long short-Term Memory), each module is composed of a cell, an entrance door, an exit door and a forgotten door. The cell remembers the values in arbitrary time intervals and the three doors regulate the flow of information that enters and leaves the cell. [13]
We will not go into more details on this topic. For a better understanding (more technical) about the LSTM, you can consult Understanding LSTM Networks. [14]

We use this module because it is considered an access cell to implement recurrent neural networks.

.

LSTM modules are extremely powerful for handling data with time series information. They can predict an arbitrary number of steps in the future. A module (or cell) LSTM has five essential components that allow you to model data in the long and short term. [15]

1. Cell state (ct): represents the internal memory of the cell that stores both short-term memory and long-term memory.

2. Hidden state (ht): this is the information of calculated exit status, current entry, previous hidden state and current cell entry that eventually uses to predict future prices of the stock market. In addition, the hidden state may decide to recover only the short or long term or both types of memory stored in the cell state to make the next prediction.

3. Input door (it): decides how much information from the current input flows to the cell state.

4. Forget gate (ft) - Decide how much information from the current entry and from the previous cell status flows to the current cell status.

5. Output gate (ot): decides how much information from the current cell state flows into the hidden state, so that, if necessary, LSTM can only select long-term memories or short-term memories and long-term memories .

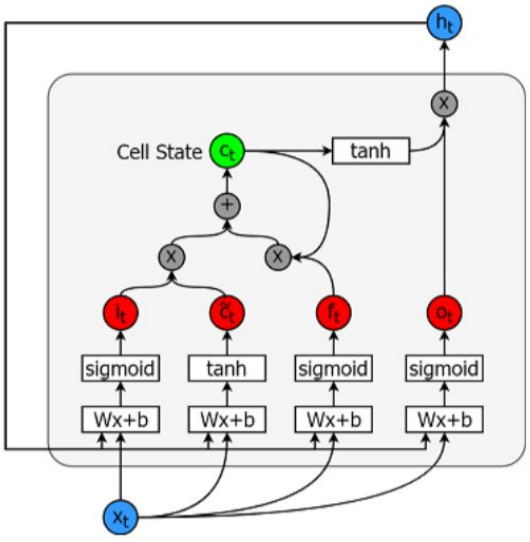


Figure 13, Graphic representation of a LSTM cell.

For an explanation of the LSTM, you can consult the reference [13]

In this part we will talk about how we are going to define the Parameters of the LSTM layer and Regression.

The module will have three layers of LSTM and a linear regression layer, which takes the output of the last Short Term Memory cell and generates the prediction for the next time step. You can use the MultiRNNCell in TensorFlow to encapsulate the three LSTM Cell objects that you previously built. In addition, you can have the LSTM cells implemented in the abandonment, as they improve performance and reduce overfitting.

The input parameters are:

- Historical data of the last five years ago.

- The training parameters have 60 timesteps, which correspond to a period of 3 months, this number of steps is based on the best result after experimenting with different time parameters, finding 60 the best time step value for the neural network to learn and understand about some correlations and trends of the data.

- In the part of the architecture of the neural network we will have a sequential class that represents a sequence of layers

- Adding a LSTM layer where we enter three arguments (units, return_secuence, input_shape).

- Adding the Dropout class with an argument of 20%.

- Adding to Dense layer with argument of (number of units).

The information on the handling of the input data for the chosen algorithm is described as follows:

Data Preprocessing; This part will import the essential libraries to perform the preprocessing of the data set. The input data will be historical from October 2013 to September 2018 (last 5 years) will be a matrix and numeric values stored in .csv format (Comma-separated values) this is the form allowed by neural networks.

We will have two matrix as input a matrix will be used as our training set and another matrix as our test set. This is because when the training is performed in the training data set the neural network will not have knowledge of the values of the test set data so the information that this within the test set data will be values never seen before by the neural network.

Benchmark

For this project we focus mainly on the application of Deep Learning, implemented an RNN algorithm (Recurrent Neural Network). In the field of Deep Learning the application of models in comparison with other algorithms of Machine Learning, can be as well added value, especially when data has a temporal relationship. [16]

To predict the trend of stock price realized with serial information in time, first we obtain the real stock price, then we obtain the expected price of the stock price and finally we visualize the results.

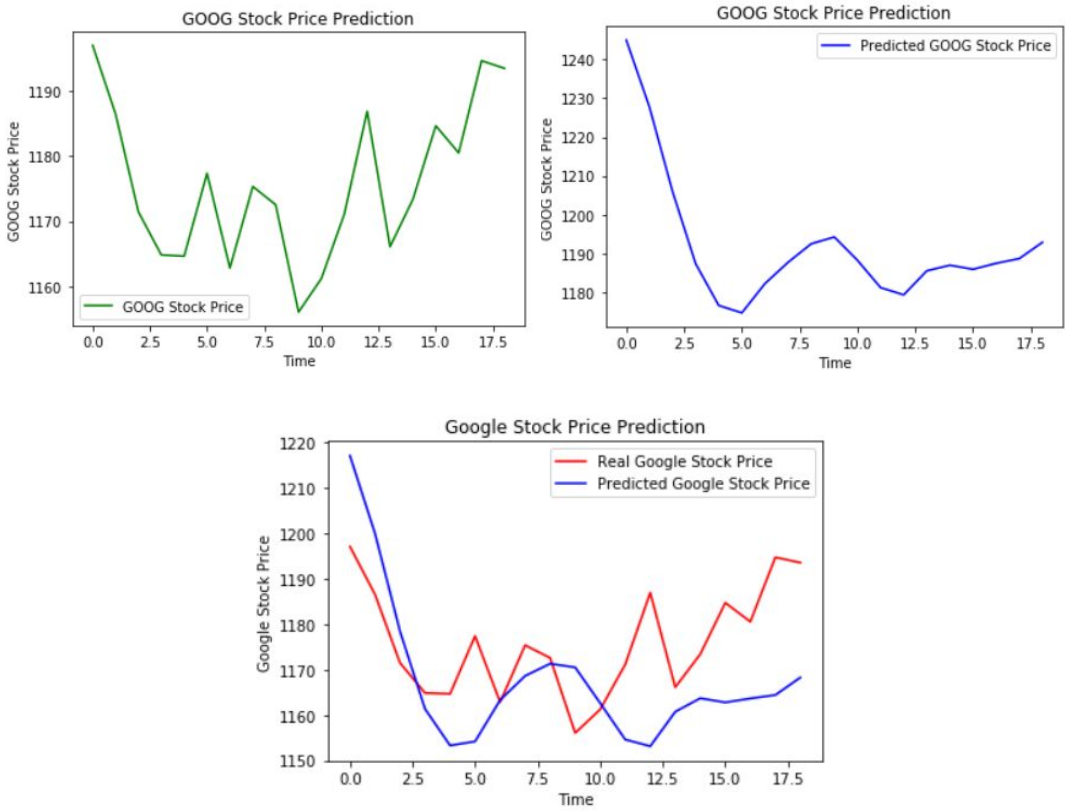


Figure 14, Left graph, Real Stock Price of the symbol GOOG (Google) of period 2018/09/04 - 2018/09/21 in total 13 observations on the right Stock price prediction, Bottom image graphical comparison of results.

These are the predictions that come from the model of RNN that we implement, we also see in the form of spikes sudden increases and decreases these non-linear changes are presented quickly, which makes that the model is left behind because it can not react too fast, however the model can react well in smooth changes, establishing that it is enabled to follow the up and down trends.

These results serve as a reference to measure the performance of the implemented model, obtaining the following metrics of error MAE and RMSE the last two metrics are the most popular for continuous variables;

- **MAE** (Mean Absolute Error), is the average of the absolute difference between the predicted values and the observed value.

- **RMSE** (root-mean-square error), represents the standard deviation of the sample of the differences between the predicted values and the observed values.

The following data are the values obtained from the measurement metric of the Benchmark model.

Score
MAE: 18.0628
RMSE: 22.6832

3 METHODOLOGY

Data preprocessing

In this section we will describe the acquisition and the preprocessing of the data in the following way:

Acquire two sets of data, one will be used for training and another for testing. This process is done by downloading the historical data of five years ago from Yahoo Finance web page which are assigned as the data set of training. The other set of data is downloaded by selecting the first month after the last date of the training data set, ie for this case the last date was 2018/08/30 and the data set for test is 2018/09/03 at 018/09/28. Recall that the indicator we use is Adj Close.

	Date	Open	High	Low	Close	Adj Close	Volume
0	2013-10-01	437.280914	440.966949	437.181549	440.634094	440.634094	3391400
1	2013-10-02	438.512909	441.806488	436.073761	441.125916	441.125916	3009900
2	2013-10-03	441.130859	444.161163	433.232239	435.214355	435.214355	4261500
3	2013-10-04	434.672882	435.919769	432.189026	433.356445	433.356445	2733600
4	2013-10-07	430.922272	434.171143	429.263062	430.072784	430.072784	2603900

```
array([[ 440.634094],
       [ 441.125916],
       [ 435.214355],
       ...,
       [1231.150024],
       [1249.300049],
       [1239.119995]])
```

Figura 15, Above Data set from Yahoo Finance 5 years ago, Below Importing the training set

The next step is Feature Scaling; Basically it is to normalize the values of the stock price, for this we will use the library of the sklearn preprocessing method, importing the MInMaxScaler library, once this is done, a class is created with the MinMaxScaler library and we assign the parameters in the range of 0 to 1.

```
In [9]: # Feature Scaling
        from sklearn.preprocessing import MinMaxScaler

In [10]: sc = MinMaxScaler(feature_range = (0, 1))

In [11]: training_set_scaled = sc.fit_transform(training_set)

In [12]: training_set_scaled

Out[12]: array([[0.01961179],
                [0.02019434],
                [0.01319222],
                ...,
                [0.95596116],
                [0.97745948],
                [0.96540142]])
```

Figura 16 Feature scaling range (0,1)

After normalizing creating data structure with 60 timsteps and 1 output; means that each moment the RNN will observe the 60 days before time t, and depending on the trends it is capturing during the 60 steps, it will try to predict the next result. Sixty is the number of timesteps, it is based on the fact that after some trainings with different values of 20 and 40 the RNN could not learn enough about the price trends. The first thing we must understand is that we will train our model to be able to predict the stock price at the moment, based on the previous 60 stock prices and, therefore, to predict the price of each exercise. We will need the 60 prices of the previous shares of the previous financial days before the actual day.

```
In [13]: # Creating a data structure with 60 timesteps and 1 output
         X_train = []
         y_train = []
         for i in range(60, 1239):
             X_train.append(training_set_scaled[i-60:i, 0])
             y_train.append(training_set_scaled[i, 0])
         X_train, y_train = np.array(X_train), np.array(y_train)

In [14]: X_train

Out[14]: array([[0.01961179, 0.02019434, 0.01319222, ..., 0.14530845, 0.15382862,
                 0.15191042],
                [0.02019434, 0.01319222, 0.01099156, ..., 0.15382862, 0.15191042,
                 0.15521733],
                [0.01319222, 0.01099156, 0.00710214, ..., 0.15191042, 0.15521733,
                 0.15577039],
                ...,
                [0.84715495, 0.8475932 , 0.84430032, ..., 0.92543712, 0.94352413,
                 0.96859945],
                [0.8475932 , 0.84430032, 0.82887838, ..., 0.94352413, 0.96859945,
                 0.95596116],
                [0.84430032, 0.82887838, 0.8253368 , ..., 0.96859945, 0.95596116,
                 0.97745948]])
```

Figura 17, Creating a data structure with 60 timesteps and t+1 output

Last step of our Data preprocessing consists of reshaping the data. It is necessary to reform often when working with data sets that contain variables with some types of sequences, for example, time series data. [17]

```
In [16]: # Reshaping
         X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))

In [46]: X_train

Out[46]: array([[[0.01961179],
                 [0.02019434],
                 [0.01319222],
                 ...,
                 [0.14530845],
                 [0.15382862],
                 [0.15191042]],

                [[0.02019434],
                 [0.01319222],
                 [0.01099156],
```

Figura 18, Reshaping

Implementación

For the implementation we will build the entire architecture of the neural network. In this first step of the implementation the RNN will be initialized as a sequence of layers.

```
In [17]: # Initialising the RNN
         regressor = Sequential()
```

Figure 19, Initialising the RNN

As a second step we will add the first layer of LSTM, we will include Dropout regularization to 20%, That's to avoid overfitting.

```
In [18]: # Adding the first LSTM layer and some Dropout regularization
         regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
         regressor.add(Dropout(0.2))
```

Figure 20, Adding the first LSTM layer and some Dropout Regularization

- Three parameters are included within the LSTM layer:

    1. The number of memory units that we want to have, we want our model to have high dimensionality therefore we need to have a large number of neurons in each of the layers. so the number of neurons that gives the best results is 50. This value can capture well the ascending and descending tendencies.

    2. Return sequencing and we set it to true because we are building a stacked LSTM therefore will have several layers.

    3. Input shape this parameter corresponds to the time steps and indicators.

- The layer of Dropout that was added has a value of 0.2, is equivalent to 20 percent of abandonment, this means, describes the neurons that you want to discard during training. It will ignore the neurons that are in the forward propagation and the backward propagation in each interaction of the training, therefore 20 percent of 50 is 10 neurons that will ignore during each interaction of the training.

We follow the same process to add three more layers, which two layers will have only the parameters of units and return_sequence with the same values. For the last LSTM layer will keep the units parameter with a volume of 50 neurons, also changing the return_sequence parameter from True to False. Recall that this parameter has False value by default so it is not necessary to write it.

```
# Adding the second LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
```

```
# Adding the third LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
```

```
# Adding the fourth LSTM layer and some Dropout regularization
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))
```

Figure 21, Adding the next three layers

In the last step of the architecture of the neural network is to add the fully connected output layer to the previous LSTM and this will be a dense layer with the unit parameter equal to 1 that corresponds to the number of neurons that should be, as we are By predicting a real value corresponding to the stock price, the output has only one dimension.

```
In [22]: # Adding the output layer
         regressor.add(Dense(units = 1))
```

Figure 22, Adding the output layer

The next step is to compile the neural network with the optimizer and loss function. We used the compilation method which is another method of the sequential class.
The parameter that we use for optimizer is 'adam'. Adam, an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments is suitable for problems that are large in terms of data. [18]

```
In [23]: # Compiling the RNN
         regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
```

Figure 23, Compiling the RNN

In the second argument we use the loss function with the parameter of 'mean_squared_error' this parameter can measure by means of the quadratic differences between the predictions and the real values.

The final step in the construction of the neural network is known as fitting the RNN to the training set. What we do is connect our training set, we train the model to obtain the parameters that we will use on the test data with the predict () function. We use the following arguments;

```
In [25]: # Fitting the RNN to the Training set
         regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)

         Epoch 1/100
         1179/1179 [==============================] - 15s - loss: 0.0378
         Epoch 2/100
         1179/1179 [==============================] - 13s - loss: 0.0046
         Epoch 3/100
         1179/1179 [==============================] - 13s - loss: 0.0037
         Epoch 4/100
         1179/1179 [==============================] - 13s - loss: 0.0036
         Epoch 5/100
         1179/1179 [==============================] - 14s - loss: 0.0037
         Epoch 6/100
         1179/1179 [==============================] - 13s - loss: 0.0034
         Epoch 7/100
         1179/1179 [==============================] - 13s - loss: 0.0034
         Epoch 8/100
         1179/1179 [==============================] - 13s - loss: 0.0035
         Epoch 9/100
         1179/1179 [==============================] - 13s - loss: 0.0027
```

Figure 24,Fitting the RNN to the Training set

- X_train, is the first argument, the inputs of the training set have the independent variables that will be the input of the new network and will be propagated to the result that will be the prediction.
- y_train, is the test set by which the predictions as a result of the training will be compared.

- epochs, Number of epochs to train the model. Note that in conjunction with initial_epoch, epochs is to be understood as "final epoch". The model is not trained for a number of iterations given by epochs, but merely until the epoch of index epochs is reached. [20]

- batch_size, Number of samples per gradient update. If not specified, batch_size will default to 32.

For a Regression, the way to evaluate the performance of the model there are two metrics called MAE (Mean Absolute Error) is the average of the absolute difference between the predicted values and the observed value. The second metric is RMSE (Mean square error). It is calculated as the root of the mean squared differences between the predictions and the actual values.

Score

MAE: 18.0628

RMSE: 22.6832

Figure 25, illustration of the two validation metrics for regression models.

Note:

The first problem was to obtain our database, the chosen source was the Yahoo Finance API but it turns out that this API no longer works, therefore the solution to the problem had to be found using fix-yahoo-finance.

After having that single price ordered from oldest to most recent, the data was normalized, dividing every value by the first one, so that the first value would become 1 and serve as a reference point to see how the stock price varied from then on.

After doing our first model (benchmark), we observed that the results that were yielded by these algorithms were not too good. However I thought that for our predictions would be close of the real stock price so evaluating the model with thismetrics does not make much sense, because the model was more suited to in the directions taken by our predictions, rather than the closeness of their values to the real stock price.

Refinement

After several adjustments to improve the prediction of the model, an algorithm with improved results was obtained. This model was implemented with several symbols observing that the model adapts well in different datasets with the adjustments made. The adjustments that were made come next:

In the part 1 Data preprocessing a change was made in the input data, changing from five historical years to ten historical years.

The structure of entry and exit was reshaping eliminating the 60 timesteps, this change decreased the training time making the process faster.

```
# Getting the inputs and the ouputs
X_train = training_set[0:1238]
y_train = training_set[1:1239]

# Reshaping
X_train = np.reshape(X_train, (1238, 1, 1))
```

Figure 26, Reshaping structure

```
# Creating a data structure with 60 timesteps and 1 output
X_train = []
y_train = []
for i in range(60, 1258):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
```

```
# Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

Figure 27, Structure before the change

In Part 2 Building the RNN, we use an LSTM layer with unit parameters equal to 4 neurons;

```
# Adding the input layer and the LSTM layer
regressor.add(LSTM(units = 4, activation = 'sigmoid', input_shape = (None, 1)))
```

Figure 28, Adding the LSTM layer

In the parameter of units, the number of neurons trying with several values was decreased and leaving the value 4 as the number that better results delivered in the prediction with test data from the month of September 2018.

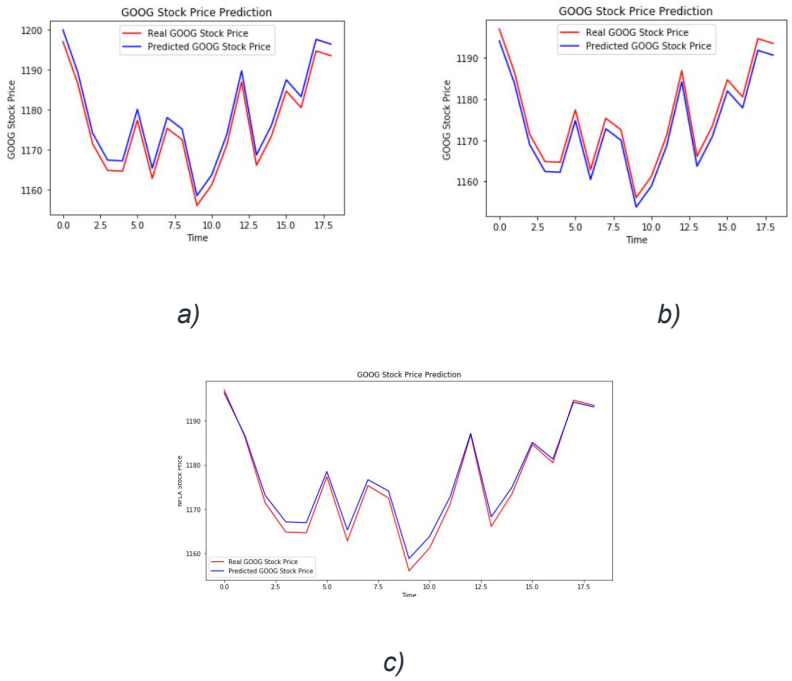


Figure 29, image a) graph with parameter of units = 10 obtaining an MAE score: 1.05 RMSE: 1.7465. In the imagen b) with parameters of units = 6 score of MAE: 2.6357 RMSE: 2.5760. In the picture c) parameter units = 4 MAE score: 1.70 RMSE: 1.6451

Also in Fitting the RNN to training set changes are made in the number of epochs increasing the value from 100 to 200.

```
# Fitting the RNN to the Training set
regressor.fit(X_train, y_train, batch_size = 32, epochs = 200)
```

Figure 30, Fitting the RNN the Training set

The changes made in this step created a model with better results, the symbol of the 3M company was used to make a test with the model that made the changes.

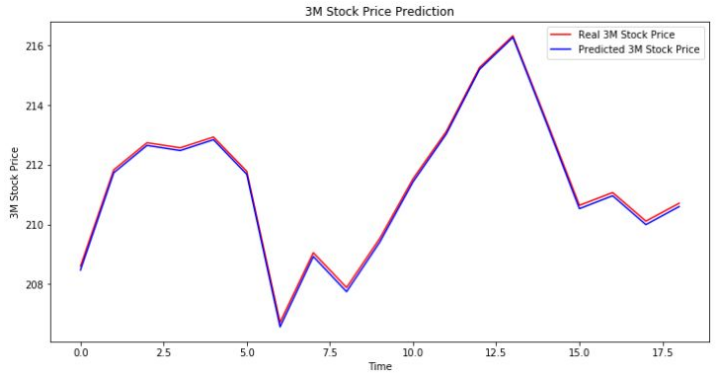


Figure 31 a, Graph of the model with the adjustments made

Score

MAE: 0.0826

RMSE: 0.0836

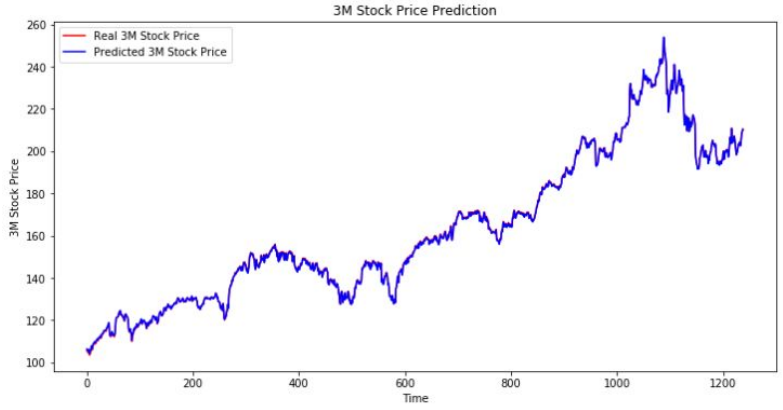


Figure 32 b, Graph of the model with the adjustments made

Score

MAE: 0.1439

RMSE: 0.2159

4 RESULT

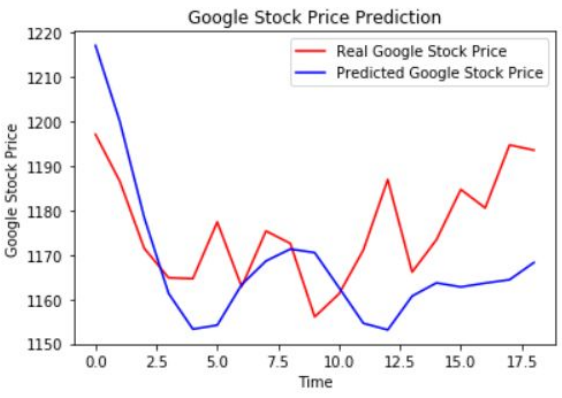
Model Evaluation and Validation

The final model was derived as a result of having made different adjustments, in each adjustment made the result was observed, this information served to have a reference point that would indicate in which adjustments we obtained the best

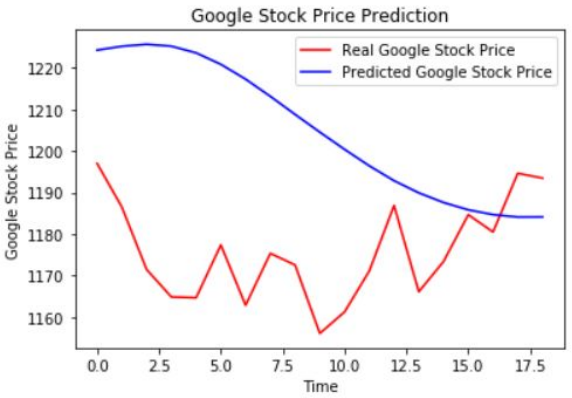
results. The analysis we made for the improvement was in the phase test, that is, after training the model we made our prediction, comparing it with the test_set remember that this data set is made up of the price stock values obtained in the month after the of the training data set. After making the prediction, we make a comparative visualization of the results, observing in the graph, if there are improvements and what score we obtained. Finally we arrive at a model, with better prediction of stock price at the moment, this prediction is adjusted very similar to the ascending and descending tendencies of the values of the data set, obtaining narrower evaluation metrics. The following table shows the different scenarios with a sensitivity analysis per model.

Modelo	history data yrs	tiemestep	Layer LSTM	units	epochs	training time	**MAE**	**RMSE**
1	5	60	4	50	100	10. s	**13.7930**	**16.8707**
2	5	60	4	3	100	5. s	**31.2821**	**36.6369**
3	5	60	1	3	100	1. s	**13.4607**	**17.5804**
4	5	20	4	3	100	1. s	**18.0345**	**2.7743**
5	5	20	1	3	100	0. s	**13.9261**	**15.8454**
6	5	--	1	4	100	0. s	**7.6194**	**7.7538**
7	5	--	1	4	250	0. s	**3.0563**	**3.0565**
8	5	--	1	4	200	1. s	**0.0826**	**0.0836**
9	10	--	1	4	200	0. s	**4.7109**	**407162**
10	10	120	4	50	100	58. s	**21.4898**	**25.5574**
11	10	--	1	4	100	0. s	**0.5193**	**0.5199**

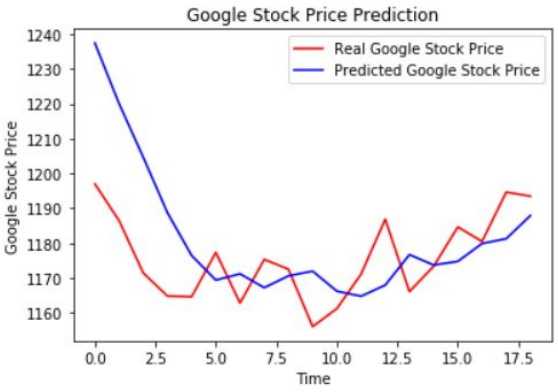
The following illustrations are the graphic representation between the Real stock price and the Prediction stock price showing the final date of the historical data of 10 years ago.



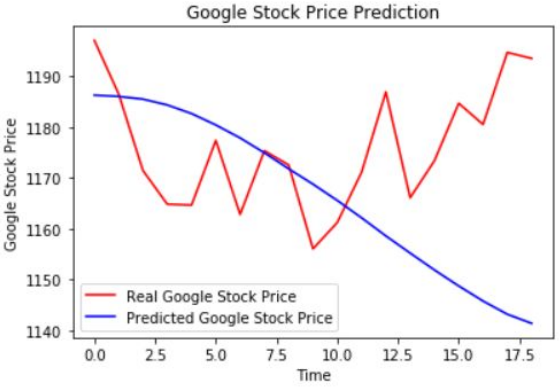
Model 1



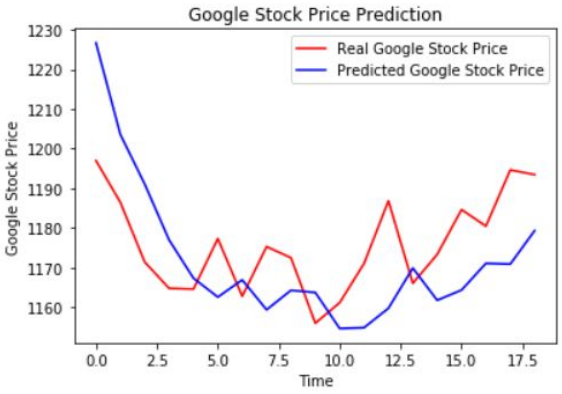
Model 2



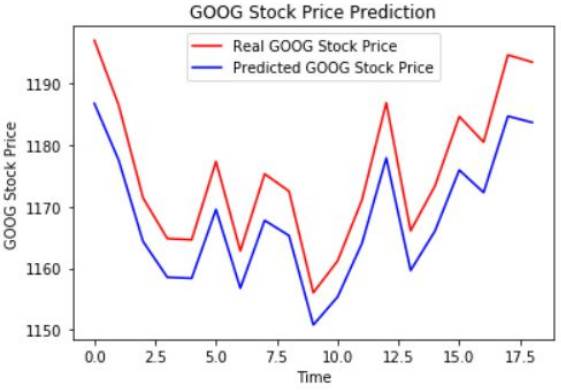
Model 3



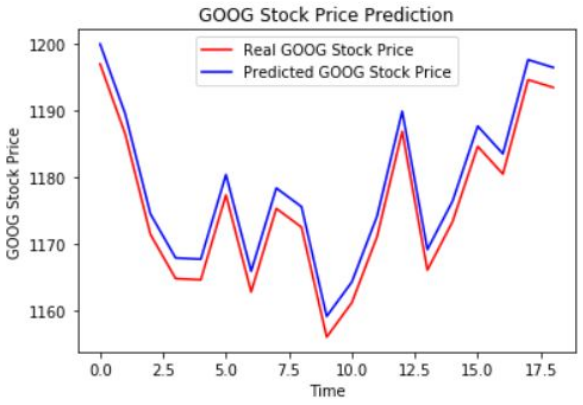
Model 4



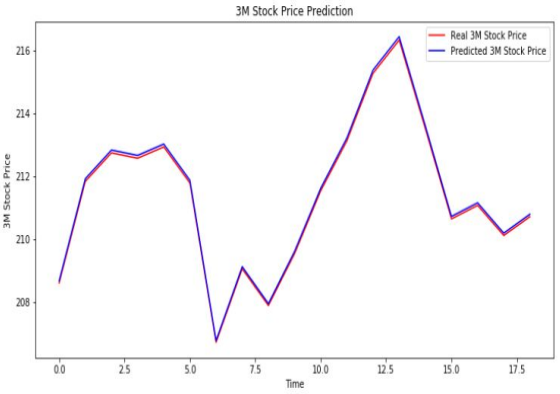
Model 5



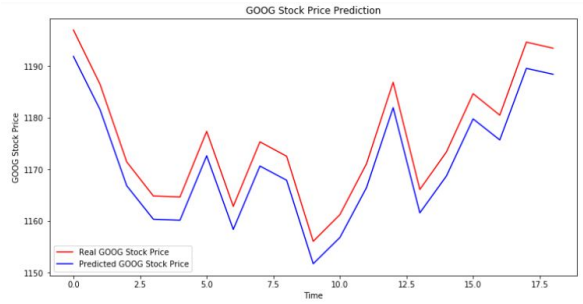
Model 6



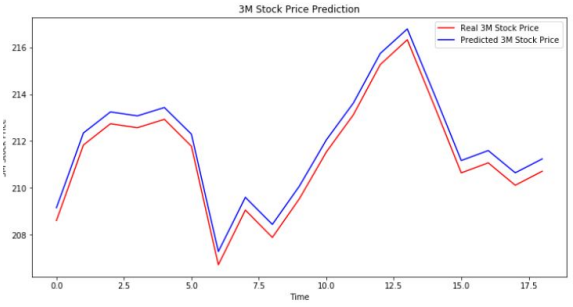
Model 7



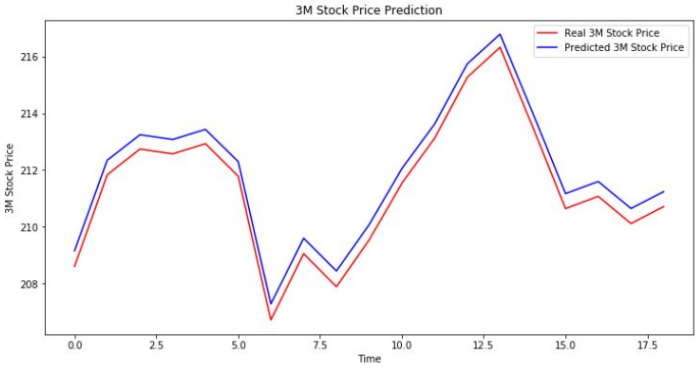
Model 8



Model 9



Model 10



Model 11

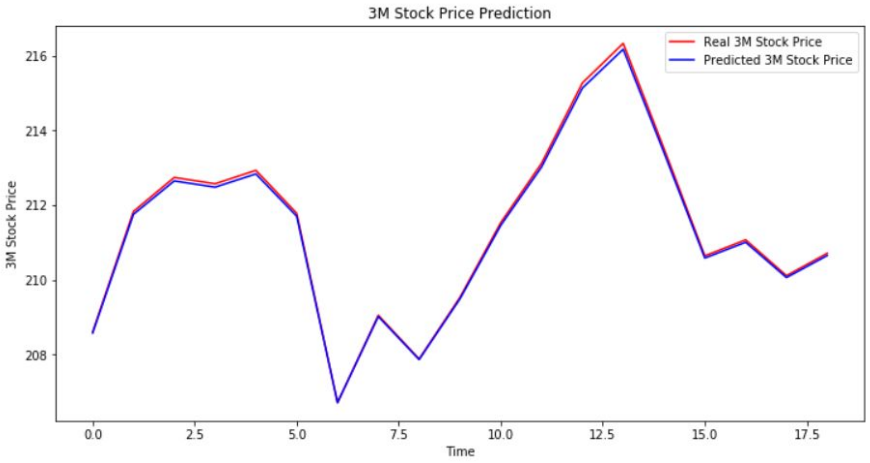


Figure 33 a, Graph of the Model 8 Prediction dataset test_set 2018/09/03 - 2018/09/28

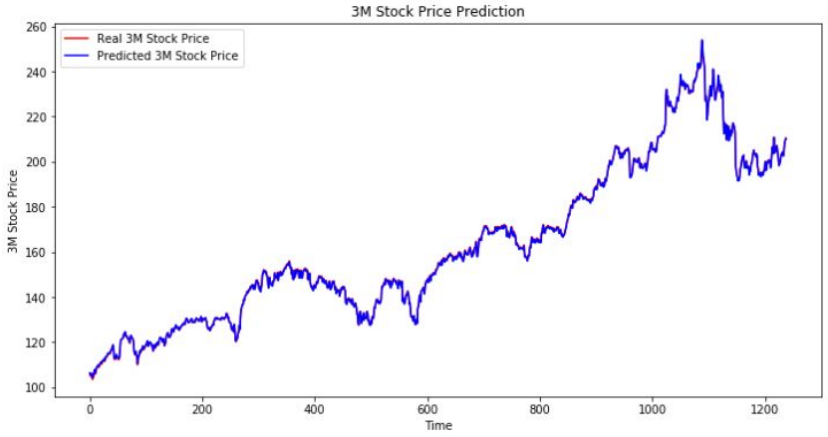


Figure 33 b, Graph of the Model 8 Prediction dataset test_set 2018/09/03 - 2018/09/28

Justificatión

The final solution of the stock price prediction model together with the results are compared with the model previously established in the Benchmark.

Recall that the RNN model is more adjusted to an upward and downward trend with difficulty to adapt to sudden increases and decreases with a change in the stock price;

```
print(real_stock
[[1197.      ]
 [1186.47998 ]
 [1171.439941]
 [1164.829956]
 [1164.640015]
 [1177.359985]
 [1162.819946]
 [1175.329956]
 [1172.530029]
 [1156.050049]
 [1161.219971]
 [1171.089966]
 [1186.869995]
 [1166.089966]
 [1173.369995]
 [1184.650024]
 [1180.48999 ]
 [1194.640015]
 [1193.469971]]
```

```
print(predicted
[[1245.0436]
 [1227.8015]
 [1205.9772]
 [1187.3217]
 [1176.5275]
 [1174.627 ]
 [1182.1897]
 [1187.6797]
 [1192.456 ]
 [1194.2057]
 [1188.1985]
 [1181.144 ]
 [1179.2723]
 [1185.4797]
 [1186.9241]
 [1185.8718]
 [1187.4166]
 [1188.633 ]
 [1192.8224]]
```

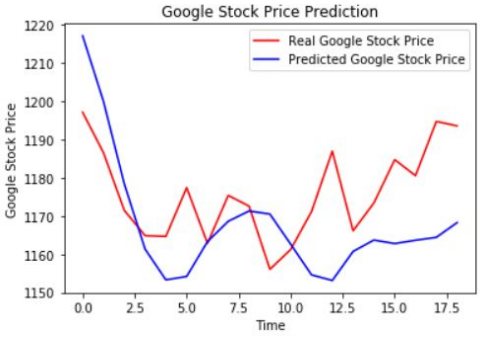


Figure 34, image and graph of the real price vs predicted stock Benchmark model

score of the model Benchmark

Score

MAE: 18.0628

RMSE: 22.6832

The following information is the evaluation of the model with the best results, which comes from the study carried out in the previous section of Result. To determine which was the best model, we used as base the lowest value of the MAE and RMSE evaluation metric using a range of 1 to 4, with 4 being the best model and 1 being the worst model.

Best	Good	Acceptable	Worst
4	3	2	1

Model	MAE	RMSE
8	0.0826	0.0836
11	0.5193	0.5199
7	3.0563	3.0565
2	31.2821	36.6369

Figure 35, Evaluation of 4 Models that goes from the best to the worst model, with two intermediate ranges

In the previous figure we show a table with the models that obtained the best results after the changes made to improve, we observe that the model number 8 has a better score of MAE: 0.0.826 and RMSE: 0.0836. If we compare the model 8 with the Benchmark model, we see that the model 8 has improvements in the prediction made, the Benchmark model has an evaluation metric of; MAE: 18.0628 and RMSE: 22.6832. Therefore there is no doubt that model 8 is better.

Model	MAE	RMSE
Benchmark	18.0628	22.6832
8	0.0826	0.0836

Figure 36, comparison of the evaluation metric between the Benchmark model and the Model 8

Stock Price Adj Close

Model Benchmark

Real	Predicted
1197.00	1245.04
1186.47	1227.80
1171.43	1205.97
1164.82	1187.32
1164.64	1176.52

a)

Model 8

Real	Predicted
208.60	208.66
211.83	211.91
212.74	212.83
212.57	212.65
212.92	213.02

b)

Figure 37, Table of Comparison of the 5 stock price of the models; a) benchmark, b) model 8

In the previous table we show the results of the real prices and the predictions of each model highlighting that the final solution has a better precision in the values of the prediction before the real values of the model, without a doubt that the evidence shows us that the improved model solved the problem raised in the project.

5 CONCLUSION

Free-form Visualization

This project was made with a Deep Learning model called RNN in which it predicts the trend at the moment of the stock price. In the evidences shown above we see that the final model exceeded the Benchmark, improving the model, in this model we obtained very small validation metrics and a minimum variation in cents of the stock price. The model was also able to adapt well to the sudden changes of the ascending and descending tendencies with excellent performance by implementing it with other symbols, obtaining predictions with values similar to MAE: 0.1038 and RMSE: 0.1067. This information can be used to promote strategy and create commercial mechanisms that support the decision to invest in the stock market.

Reflection

Trying to predict the stock price on a given day is reduced to a regression problem. The application of deep learning to predict trends and time series is a tool that addresses this problem.

The high-level process used for the development of this project includes the following steps:

- Incorporate the required libraries (Keras, tensorflow, Pandas, Matplotlib, Sklearn, Numpy).
- Develop a data request implementation in python using Yahoo Finance, to access the price data.

- Preprocessing of the acquired data set.
- Create a deep learning model.
- Define the architecture of the model using Keras.
- Train the model.
- Capture model predictions.
- Calculate the error.
- Visualize document and compare results.

Do this project with the intention of learning a new algorithm of Deep Learning used for more structured data that includes sequences, predictions of trends and time series.

After carrying out the project and making the evaluations, it was impressive to see how the algorithm learned to predict stock prices, letting me see that the information can be used to create commercial strategies that provide informative as well as economic benefits to the user. The process of optimization was manually adjusting different values in the parameters until arriving at an optimal model that delivered a smaller margin of error. After the model was used to test its performance with other data set of some different symbols chosen arbitrarily. Under these conditions the predictions made were surprising, because I could appreciate that the model also performed optimally with other previously trained data.

The development process in its different stages overcame its own difficulties, which were satisfactorily resolved every time a problem appeared, sometimes it took more time to solve them than many other solutions could find them in forums other links (included as references), in summary, at the end of the road you reach the goal, create a stock price prediction.

Improvement

Deep Learning is based on a modular system, it is an algorithm that has different blocks that can be put together to form an architecture that can give an improved performance to certain types of problems. These blocks grant an infinity of combinations to form different architectures and each one gives us different performances.
This way of building neural networks is still a search function of combinations, to see which works better is usually a form of trial and error, that is, there is no logical process from which it is determined that change is applied to get exactly one result, so we have a very large universe of possibilities and there is no technique to say this is the best architecture that should be applied to build the best result that is often beyond the capabilities of non-experts. However, this limitation is a problem of Artificial Intelligence that can solve this problem and is known as AutoML. [19]

I have no doubt that to improve the algorithm it is possible if I dedicate time to find an architecture that offers better performance. On the other hand we could include a user interface, also prepare the model to be used as an application that provides tools so that it can help to make decisions to the investors.

My experience in this project was really great, I can say that my mind stretched a lot, I am very satisfied, I understand that there is a long way to learn and I would like to learn. Udacity I can take more than my expectations, I am very happy to have had this experience.

This trip that I started was the result of the concern to make own projects to improve aspects of the real world, I think this is the first stage of the beginning of a longer trip. I want to add my gratitude to the entire Udacity team. Thanks a thousand thanks!

Last comment: I feel blessed for having participated in this course, the study I have of the university I did in Mexico, Spanish is my native language, I would like to tell you that the content of this report was written by me, so I ask the English speaking reader to be lenient if any sentence could sound a little weird.

References

1.- Wikipedia “Stock market prediction” (https://en.wikipedia.org/wiki/Stock_market_prediction), on 09/11/2018

2.- Serenity Stocks (https://www.serenitystocks.com/blog/quotes-warren-buffett-and-benjamin-graham) on 09/11/2018

3.- Wikipedia “Stock market prediction” (https://en.wikipedia.org/wiki/Stock_market_prediction), on 09/11/2018

4.- Choosing the Right Metric for Evaluating Machine Learning Models,

(https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4), on 10/10/2018

5.- List of S&P 500 companies, (https://en.wikipedia.org/wiki/List_of_S%26P_500_companies), on 10/10/2018

6.- Investopedia, Standard & Poor's 500 Index - S&P 500, (https://www.investopedia.com/terms/s/sp500.asp), on 10/10/2018

7.- Adjusted Closing Price, (https://www.investopedia.com/terms/a/adjusted_closing_price.asp), on 10/11/2018

8.- Standar Deviation, (https://www.investopedia.com/terms/s/standarddeviation.asp), on 10/13/2018

9.- Setting up a Bollinger Band with Python, (https://medium.com/python-data/setting-up-a-bollinger-band-with-python-28941e2fa300), on 10/14/2018

10.- Bollinger Band, (https://www.investopedia.com/terms/b/bollingerbands.asp), on 10/14/2018

11.- Redes neuronales Recurrentes (RNN), Dr. Erick Zamora, (https://www.scribd.com/doc/295974898/Redes-Neuronales-Recurrentes), on 10/15/18

12.- The Vanishing Gradient Problem, (https://medium.com/@anishsingh20/the-vanishing-gradient-problem-48ae7f501257), on 10/15/2018

13.- What is the vanishing gradient problem?, (https://www.quora.com/What-is-the-vanishing-gradient-problem), on 10/18/2018

14.- Understanding LSTM Networks (http://colah.github.io/posts/2015-08-Understanding-LSTMs/), on 10/18/2018

15.- Understanding LSTM and its diagrams, (https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714), on 10/15/2018

16.- Indra den Bakker. Python Deep Learning Cookbook, 2017 Pack Publishing

17.- University of Virginia Library, Research Data Services (https://data.library.virginia.edu/stata-basics-reshape-data/), on 10/24/2018

18.- Adam: A Method for Stochastic Optimization (https://arxiv.org/abs/1412.6980), on 10/25/2018

19.- Automated machine learning (AutoML), (https://en.wikipedia.org/wiki/Automated_machine_learning), on 11/01/2018

20.- Train a Keras model (https://keras.rstudio.com/reference/fit.html), on 11/03/2018