**Stock Market Prediction**

Introduction

Stock market prediction has been a significant area of research in Machine Learning. [Machine learning algorithms](https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/) such as [regression](https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/), [classifier](https://www.analyticsvidhya.com/blog/2021/09/a-complete-guide-to-understand-classification-in-machine-learning/), and [support vector machine](https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/) (SVM) help predict the stock market. This article presents a simple implementation of analyzing and forecasting stock market prediction using machine learning. The case study focuses on a popular online retail store, and Random Forest is a powerful tree-based technique for predicting stock prices.

**Learning Objectives**

* In this tutorial, we will learn about the best ways possible to predict stock prices using a long-short-term memory (LSTM) for time series forecasting.
* We will learn everything about stock market prediction using LSTM.

What is the Stock Market?

The stock market is the collection of markets where stocks and other securities are bought and sold by investors. Publicly traded companies offer shares of ownership to the public, and those shares can be bought and sold on the stock market. Investors can make money by buying shares of a company at a low price and selling them at a higher price. The stock market is a key component of the global economy, providing businesses with funding for growth and expansion. It is also a popular way for individuals to invest and grow their wealth over time.

Importance of Stock Market

| **Importance** | **Description** |
| --- | --- |
| Capital Formation | It provides a source of capital for companies to raise funds for growth and expansion. |
| Investment Opportunities | Investors can potentially grow their wealth over time by investing in the stock market. |
| Economic Indicators | The stock market can indicate the overall health of the economy. |
| Job Creation | Publicly traded companies often create jobs and contribute to the economy’s growth. |
| Corporate Governance | Shareholders can hold companies accountable for their actions and decision-making processes. |
| Risk Management | Investors can use the stock market to manage their investment risk by diversifying their portfolio. |
| Market Efficiency | The stock market helps allocate resources efficiently by directing investments to companies with promising prospects. |

What is Stock Market Prediction? [Problem Statement]

Let us see the data on which we will be working before we begin implementing the software to anticipate stock market values. In this section, we will examine the stock price of Microsoft Corporation (MSFT) as reported by the National Association of Securities Dealers Automated Quotations (NASDAQ). The stock price data will be supplied as a Comma Separated File (.csv) that may be opened and analyzed in Excel or a Spreadsheet.

MSFT’s stocks are listed on NASDAQ, and their value is updated every working day of the stock market. It should be noted that the market does not allow trading on Saturdays and Sundays. Therefore, there is a gap between the two dates. The Opening Value of the stock, the Highest and Lowest values of that stock on the same day, as well as the Closing Value at the end of the day are all indicated for each date.

The Adjusted Close Value reflects the stock’s value after dividends have been declared (too technical!). Furthermore, the total volume of the stocks in the market is provided. With this information, it is up to the job of a Machine Learning/Data Scientist to look at the data and develop different algorithms that may extract patterns from the historical data of the Microsoft Corporation stock.

Stock Market Prediction Using the Long Short-Term Memory Method

We will use the Long Short-Term Memory(LSTM) method to create a Machine Learning model to forecast Microsoft Corporation stock values. They are used to make minor changes to the information by multiplying and adding. Long-term memory (LSTM) is a deep learning artificial recurrent neural network (RNN) architecture.

Unlike traditional feed-forward neural networks, LSTM has feedback connections. It can handle single data points (such as pictures) as well as full data sequences (such as speech or video).

**Program Implementation**

We will now go to the section where we will utilize Machine Learning techniques in Python to estimate the stock value using the LSTM.

Step 1: Importing the Libraries

As we all know, the first step is to import the libraries required to preprocess Microsoft Corporation stock data and the other libraries required for constructing and visualizing the LSTM model outputs. We’ll be using the Keras library from the TensorFlow framework for this. All modules are imported from the Keras library.

#Importing the Libraries

**import** pandas **as** PD

**import** NumPy **as** np

%matplotlib inline

**import** matplotlib. pyplot **as** plt

**import** matplotlib

**from** sklearn. Preprocessing **import** MinMaxScaler

**from** Keras. layers **import** LSTM, Dense, Dropout

**from** sklearn.model\_selection **import** TimeSeriesSplit

**from** sklearn.metrics **import** mean\_squared\_error, r2\_score

**import** matplotlib. dates **as** mandates

**from** sklearn. Preprocessing **import** MinMaxScaler

**from** sklearn **import** linear\_model

**from** Keras. Models **import** Sequential

**from** Keras. Layers **import** Dense

**import** Keras. Backend **as** K

**from** Keras. Callbacks **import** EarlyStopping

**from** Keras. Optimisers **import** Adam

**from** Keras. Models **import** load\_model

**from** Keras. Layers **import** LSTM

**from** Keras. utils.vis\_utils **import** plot\_model

**Step 2: Getting to Visualising the Stock Market Prediction Data**

Using the Pandas Data Reader library, we will upload the stock data from the local system as a Comma Separated Value (.csv) file and save it to a pandas DataFrame. Finally, we will examine the data.

#**Get** the Dataset

df=pd.read\_csv(“MicrosoftStockData.csv”,na\_values=[‘**null**’],index\_col=’Date’,parse\_dates=True,infer\_datetime\_format=True)

df.head()

**Step 3: Checking for Null Values by Printing the DataFrame Shape**

In this step, firstly, we will print the structure of the dataset. We’ll then check for null values in the data frame to ensure that there are none. The existence of null values in the dataset causes issues during training since they function as outliers, creating a wide variance in the training process.

#Print the shape of Dataframe  and Check for Null Values

print(“Dataframe Shape: “, df. shape)

print(“Null Value Present: “, df.IsNull().values.any())

Output:

>> Dataframe Shape: (7334, 6)

>>Null Value Present: False

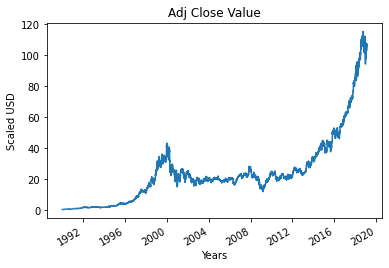
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Date** | **Open** | **High** | **Low** | **Close** | **Adj Close** | **Volume** |
| **1990-01-02** | 0.605903 | 0.616319 | 0.598090 | 0.616319 | 0.447268 | 53033600 |
| **1990-01-03** | 0.621528 | 0.626736 | 0.614583 | 0.619792 | 0.449788 | 113772800 |
| **1990-01-04** | 0.619792 | 0.638889 | 0.616319 | 0.638021 | 0.463017 | 125740800 |
| **1990-01-05** | 0.635417 | 0.638889 | 0.621528 | 0.622396 | 0.451678 | 69564800 |
| **1990-01-08** | 0.621528 | 0.631944 | 0.614583 | 0.631944 | 0.458607 | 58982400 |

**Step 4: Plotting the True Adjusted Close Value**

The Adjusted Close Value is the final output value that will be forecasted using the Machine Learning model. This figure indicates the stock’s closing price on that particular day of stock market trading.

#Plot the True Adj Close Value

df[‘Adj Close’].plot()



**Step 5: Setting the Target Variable and Selecting the Features**

The output column is then assigned to the target variable in the following step. It is the adjusted relative value of Microsoft Stock in this situation. Furthermore, we pick the features that serve as the independent variable to the target variable (dependent variable). We choose four characteristics to account for training purposes:

* Open
* High
* Low
* Volume

#Set Target Variable

output\_var = PD.DataFrame(df[‘Adj Close’])

#Selecting the Features

features = [‘Open’, ‘High’, ‘Low’, ‘Volume’]

**Step 6: Scaling**

To decrease the computational cost of the data in the table, we will scale the stock values to values between 0 and 1. As a result, all of the data in large numbers is reduced, and therefore memory consumption is decreased. Also, because the data is not spread out in huge values, we can achieve greater precision by scaling down. To perform this, we will be using the MinMaxScaler class of the sci-kit-learn library.

#Scaling

scaler = MinMaxScaler()

feature\_transform = scaler.fit\_transform(df[features])

feature\_transform= pd.DataFrame(columns=features, data=feature\_transform, index=df.index)

feature\_transform.head()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Open** | **High** | **Low** | **Volume** |
| **1990-01-02** | 0.000129 | 0.000105 | 0.000129 | 0.064837 |
| **1990-01-03** | 0.000265 | 0.000195 | 0.000273 | 0.144673 |
| **1990-01-04** | 0.000249 | 0.000300 | 0.000288 | 0.160404 |
| **1990-01-05** | 0.000386 | 0.000300 | 0.000334 | 0.086566 |
| **1990-01-08** | 0.000265 | 0.000240 | 0.000273 | 0.072656 |

As shown in the above table, the values of the feature variables are scaled down to lower values when compared to the real values given above.

**Step 7: Creating a Training Set and a Test Set for Stock Market Prediction**

Before inputting the entire dataset into the training model, we need to partition it into training and test sets. The Machine Learning LSTM model will undergo training using the data in the training set, and its accuracy and backpropagation will be tested against the test set.

To accomplish this, we will employ the TimeSeriesSplit class from the sci-kit-learn library. We will configure the number of splits to be 10, indicating that 10% of the data will serve as the test set, while the remaining 90% will train the LSTM model. The advantage of employing this Time Series split lies in its examination of data samples at regular time intervals.

#Splitting to Training set and Test set

timesplit= TimeSeriesSplit(n\_splits=10)

**for** train\_index, test\_index **in** timesplit.split(feature\_transform):

X\_train, X\_test = feature\_transform[:len(train\_index)], feature\_transform[len(train\_index): (len(train\_index)+len(test\_index))]

y\_train, y\_test = output\_var[:len(train\_index)].values.ravel(), output\_var[len(train\_index): (len(train\_index)+len(test\_index))].values.ravel()

**Step 8: Data Processing For LSTM**

Once the training and test sets are finalized, we will input the data into the LSTM model. Before we can do that, we must transform the training and test set data into a format that the LSTM model can interpret. As the LSTM needs that the data to be provided in the 3D form, we first transform the training and test data to NumPy arrays and then restructure them to match the format (Number of Samples, 1, Number of Features). Now, 6667 are the number of samples in the training set, which is 90% of 7334, and the number of features is 4. Therefore, the training set is reshaped to reflect this (6667, 1, 4). Likewise, the test set is reshaped.

#Process the data for LSTM

trainX =np.array(X\_train)

testX =np.array(X\_test)

X\_train = trainX.reshape(X\_train.shape[0], 1, X\_train.shape[1])

X\_test = testX.reshape(X\_test.shape[0], 1, X\_test.shape[1])

**Step 9: Building the LSTM Model for Stock Market Prediction**

Finally, we arrive at the point when we construct the LSTM Model. In this step, we’ll build a Sequential Keras model with one LSTM layer. The LSTM layer has 32 units and is followed by one Dense Layer of one neuron.

We compile the model using Adam Optimizer and the Mean Squared Error as the loss function. For an LSTM model, this is the most preferred combination. The model is plotted and presented below.

#Building the LSTM Model

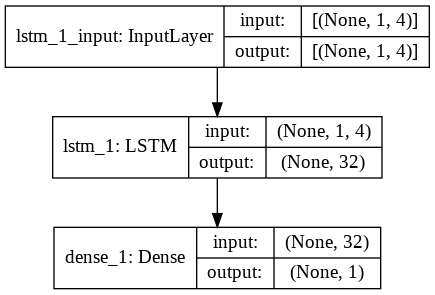
lstm = Sequential()

lstm.add(LSTM(32, input\_shape=(1, trainX.shape[1]), activation=’relu’, return\_sequences=False))

lstm.add(Dense(1))

lstm.compile(loss=’mean\_squared\_error’, optimizer=’adam’)

plot\_model(lstm, show\_shapes=True, show\_layer\_names=True)



**Step 10: Training the Stock Market Prediction Model**

Finally, we use the fit function to train the LSTM model created above on the training data for 100 epochs with a batch size of 8.

#Model Training

history=lstm.fit(X\_train, y\_train, epochs=100, batch\_size=8, verbose=1, shuffle=False)

Eросh 1/100

834/834 [==============================] – 3s 2ms/steр – lоss: 67.1211

Eросh 2/100

834/834 [==============================] – 1s 2ms/steр – lоss: 70.4911

Eросh 3/100

834/834 [==============================] – 1s 2ms/steр – lоss: 48.8155

Eросh 4/100

834/834 [==============================] – 1s 2ms/steр – lоss: 21.5447

Eросh 5/100

834/834 [==============================] – 1s 2ms/steр – lоss: 6.1709

Eросh 6/100

834/834 [==============================] – 1s 2ms/steр – lоss: 1.8726

Eросh 7/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.9380

Eросh 8/100

834/834 [==============================] – 2s 2ms/steр – lоss: 0.6566

Eросh 9/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.5369

Eросh 10/100

834/834 [==============================] – 2s 2ms/steр – lоss: 0.4761

.

.

.

.

Eросh 95/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.4542

Eросh 96/100

834/834 [==============================] – 2s 2ms/steр – lоss: 0.4553

Eросh 97/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.4565

Eросh 98/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.4576

Eросh 99/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.4588

Eросh 100/100

834/834 [==============================] – 1s 2ms/steр – lоss: 0.4599

Finally, we can observe that the loss value has dropped exponentially over time over the 100-epoch training procedure, reaching a value of 0.4599.

**Step 11: Making the LSTM Prediction**

Now that we have our model ready, we can use it to forecast the Adjacent Close Value of the Microsoft stock by using a model trained using the LSTM network on the test set. We can accomplish this by employing simple prediction model on the LSTM model

#LSTM Prediction

y\_pred= lstm.predict(X\_test)

Step 12: Comparing Predicted vs True Adjusted Close Value – LSTM

Finally, now that we’ve projected the values for the test set, we can display the graph to compare both Adj Close’s true values and Adj Close’s predicted value using the LSTM Machine Learning model.

#Predicted vs True Adj Close Value – LSTM

plt.plot(y\_test, label=’True Value’)

plt.plot(y\_pred, label=’LSTM Value’)

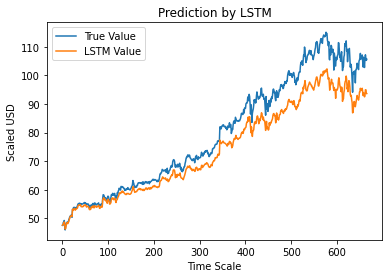
plt.title(“Prediction by LSTM”)

plt.xlabel(‘Time Scale’)

plt.ylabel(‘Scaled USD’)

plt.legend()

plt.show()



The graph above demonstrates that the extremely basic single LSTM network model created above detects some patterns. We may get a more accurate depiction of every specific company’s stock value by fine-tuning many parameters and adding more LSTM layers to the model.

Conclusion

However, with the introduction of Machine Learning and its strong algorithms, the most recent market research and Stock Price Prediction using machine learning advancements have begun to include such approaches in analyzing stock market data. The Opening Value of the stock, the Highest and Lowest values of that stock on the same day, as well as the Closing Value at the end of the day are all indicated for each date. Furthermore, the total volume of the stocks in the market is provided; with this information, it is up to the job of a Machine Learning Data Scientist to look at the data and develop different algorithms that may help in finding appropriate stocks values.

Predicting the stock market was a time-consuming and laborious procedure a few years or even a decade ago. However, with the application of machine learning for stock market forecasts, the procedure has become much simpler. Machine learning not only saves time and resources but also outperforms people in terms of performance. it will always prefer to use a trained computer algorithm since it will advise you based only on facts, numbers, and data and will not factor in emotions or prejudice. It would be interesting to incorporate sentiment analysis on news & social media regarding the stock market in general, as well as a given stock of interest.

**Key Takeaways**

* Stock Price Prediction using machine learning helps in discovering the future values of a company’s stocks and other assets.
* Predicting stock prices helps in gaining significant profits.