

AHMED DIDER RAHAT

s40183@bht-berlin.de

Deep Learning

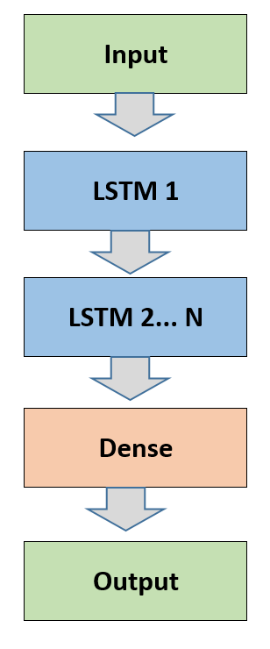
**CRYPTO CURRENCY FUTURE PRICE PREDICTION USING STACK LSTM**

**Introduction:** Crypto currency is digital currency in modern finance system. It’s an end-to-end system that can enable anyone anywhere to send and receive payments. Instead of being physical money carried around and exchanged in the real world, crypto currency payments exist purely as digital entries in the cyber space. As a result, the interest on crypto investment increase dramatically. In this project I try to implement stack **LSTM** to predict the future price of two most widely used crypto currency **BITCOIN** and **ETHEREUM**.

**LSTM:** Long short-term memory (LSTM) is an artificial neural network used in the field deep learning. LSTM has feedback connections. It is a recurrent neural network that can process not only single data points, but also entire sequences of data. LSTM networks are well-suited to classifying, processing and making predictions based on **time series data**.

**Time series data**: A time series is a series of data points indexed in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data.

**Stacked-LSTM:** The Stacked LSTM is an extension to this model that has multiple hidden LSTM layers where each layer contains multiple memory cells.



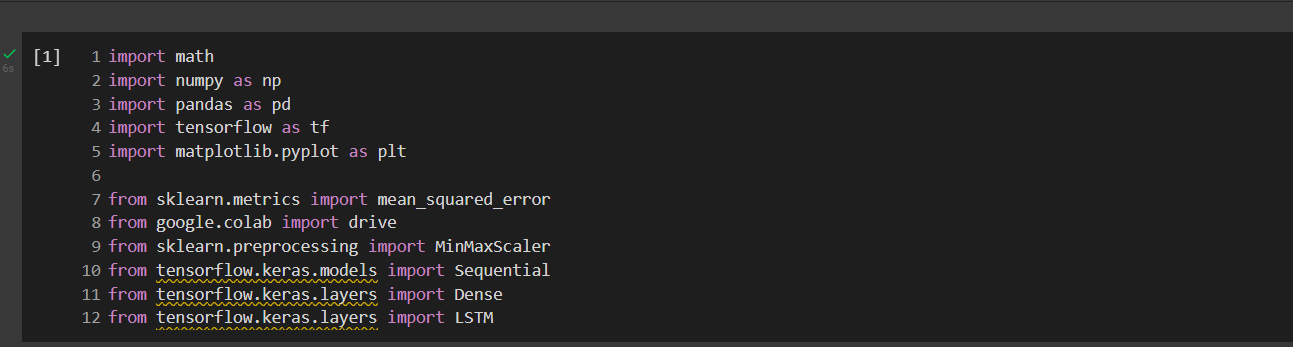
**Figure 01:** Stacked LSTM with N-Hidden LSTM Layers

**My Dataset:** To train and test my implementation I used two type of crypto currency data (Bitcoin and Ethereum). All the data are downloaded from [finance.yahoo.com](https://finance.yahoo.com/).

* **Bitcoin data:** There are total 2,829 data points and each of them has 7 columns (Date, Open, High, Low, Close, Adj. Close, Volume). These data are available from 9/17/2014 to 6/15/2022.
* **Ethereum data:** Total number of 1,680 data points and also has same columns as Bitcoin and the data points are distributed from 2017-11-09 to 2022-06-15.

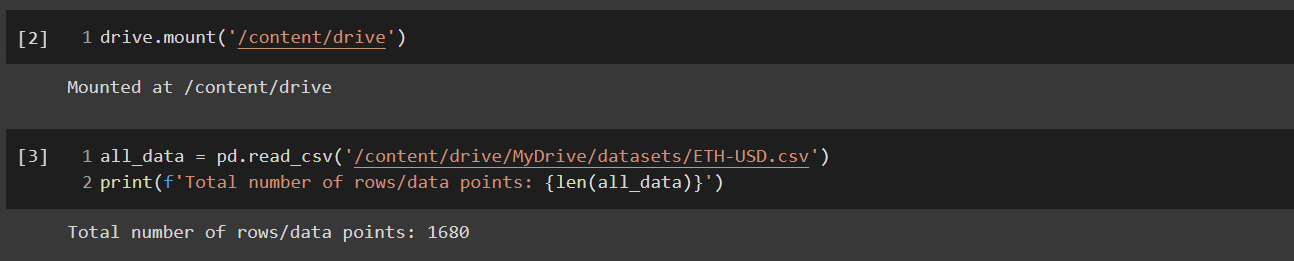
**Implementation:**

1. **IDE:** I used google colab for this project as my IDE.
2. **Import necessary libraries:** I used several libraries in my implementation. So, at first import them in my code base.



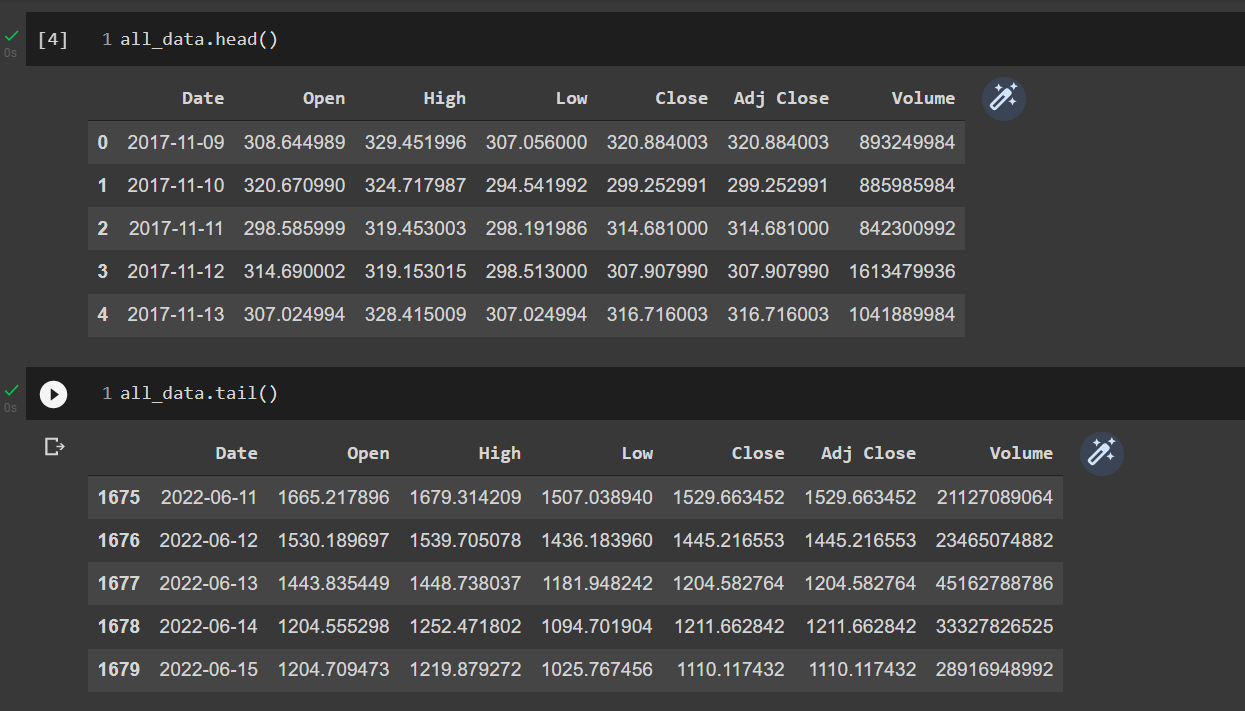
**Figure 02:** Import libraries in IDE

1. **Connect and Read data file:** As I use data file from google drive. So, I used mount to connect the notebook with google drive.



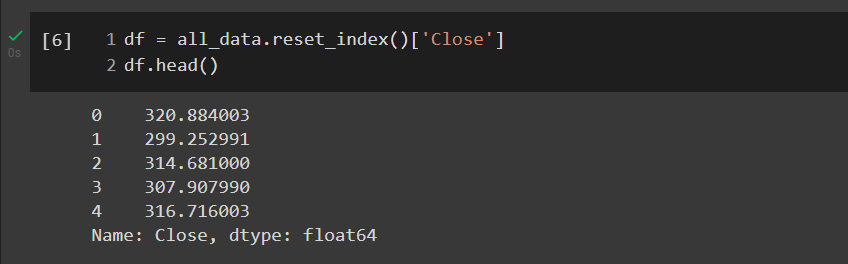
**Figure 03:** Connect with drive and read input file

1. **Print head and trail of the data:** First and last few data of the input data file:



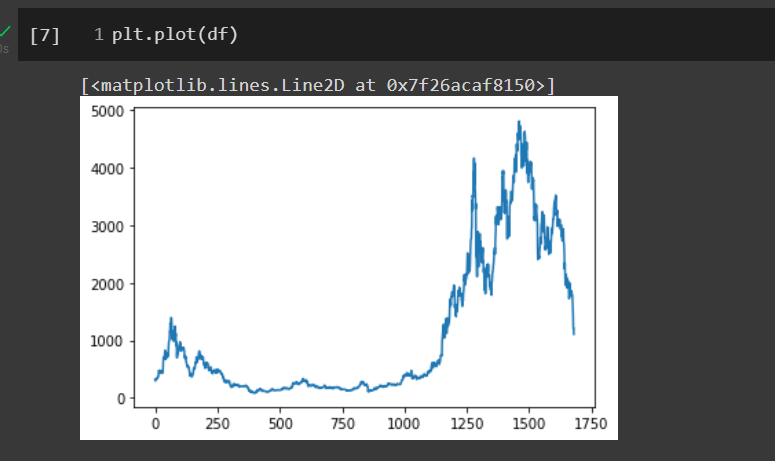
**Figure 04:** head and tail of the data

1. **Remove unnecessary data:** As I am concern with the Close data and want to use time series. So, I only took the values of Close columns.



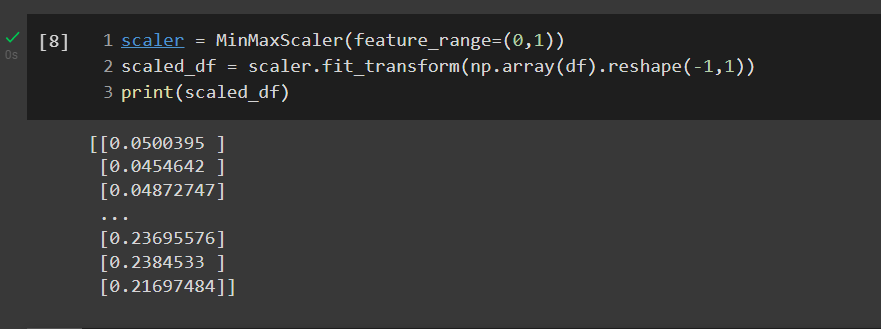
**Figure 05:** only took the Close data for implementation

1. **Input data plotting:** Plot the close data graphically.



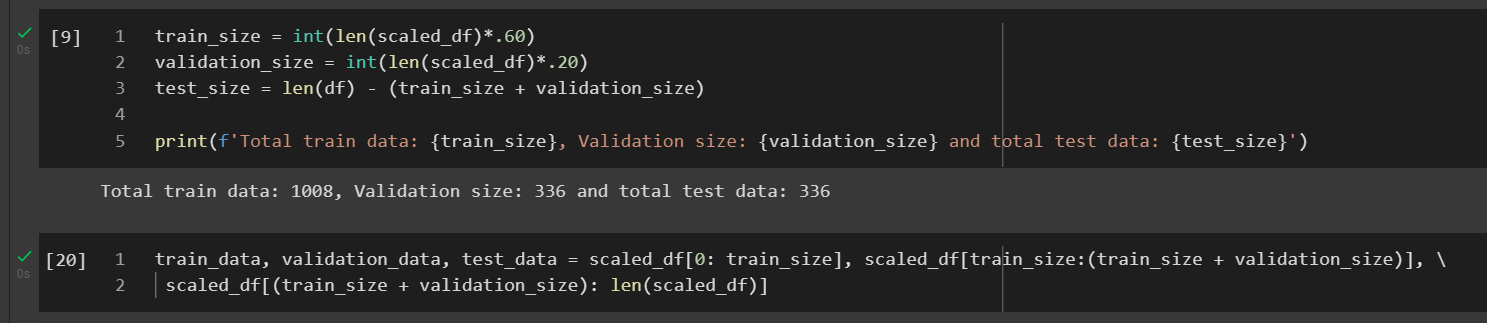
**Figure 06:** Plot close data

1. **Scaling:** As LSTM is highly sensitive to scale. So, I have to scale the data into [0, 1] interval.



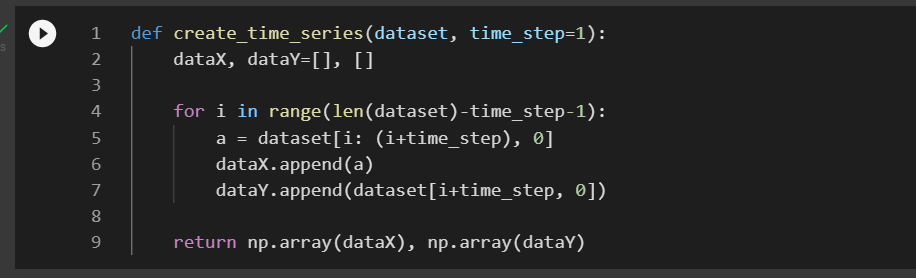
**Figure 07:** Scale data from 0 to 1

1. **Data Split:** I split my data into 3 categories train 60%, validation 20%, and test 20%.

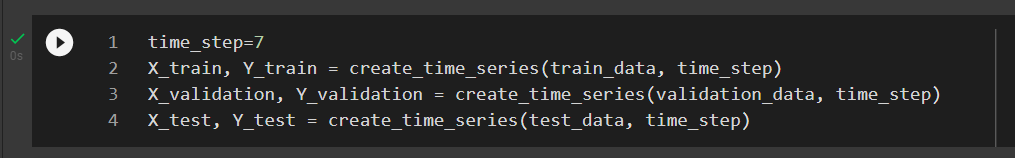


**Figure 08:** Split the input data

1. **Time series creation:** This function create time series from the input data set.

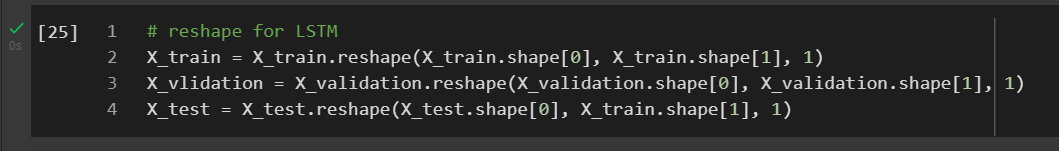


**Figure 09:** time series function



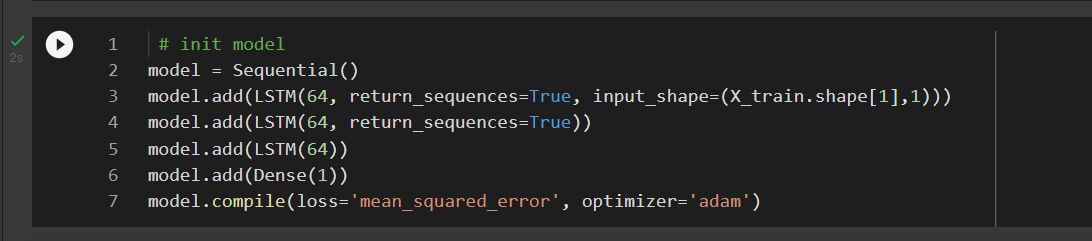
**Figure 10:** time calling

1. **Re-shape time series:** In this part I am reshaping the input sets into 3-d array as per the LSTM requirement.

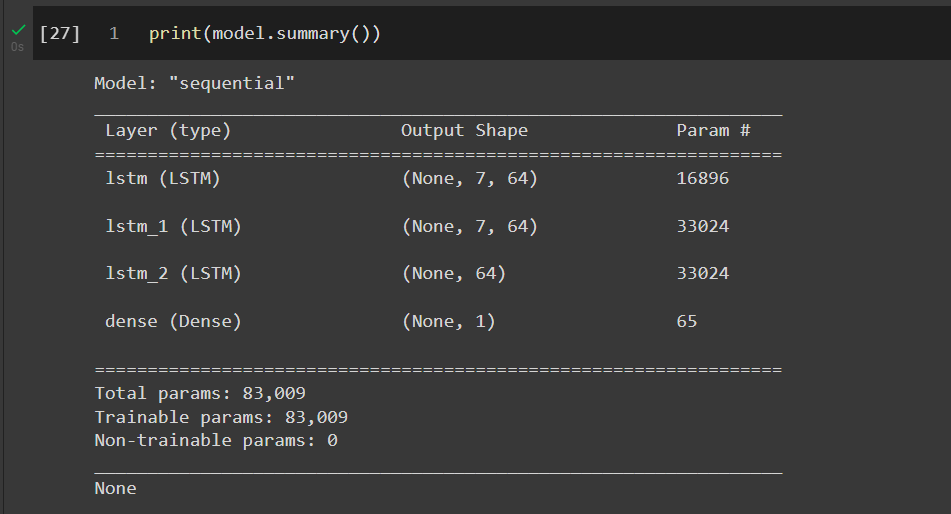


**Figure 11:** re-shape time calling

1. **Model Initialization:** In my code base I used 3 layers of LSTM. Though I used stacked LSTM the first 2 return sequence are true. I also used ‘adam’ as the optimizer and ‘mean\_squared\_error’ as loss function.

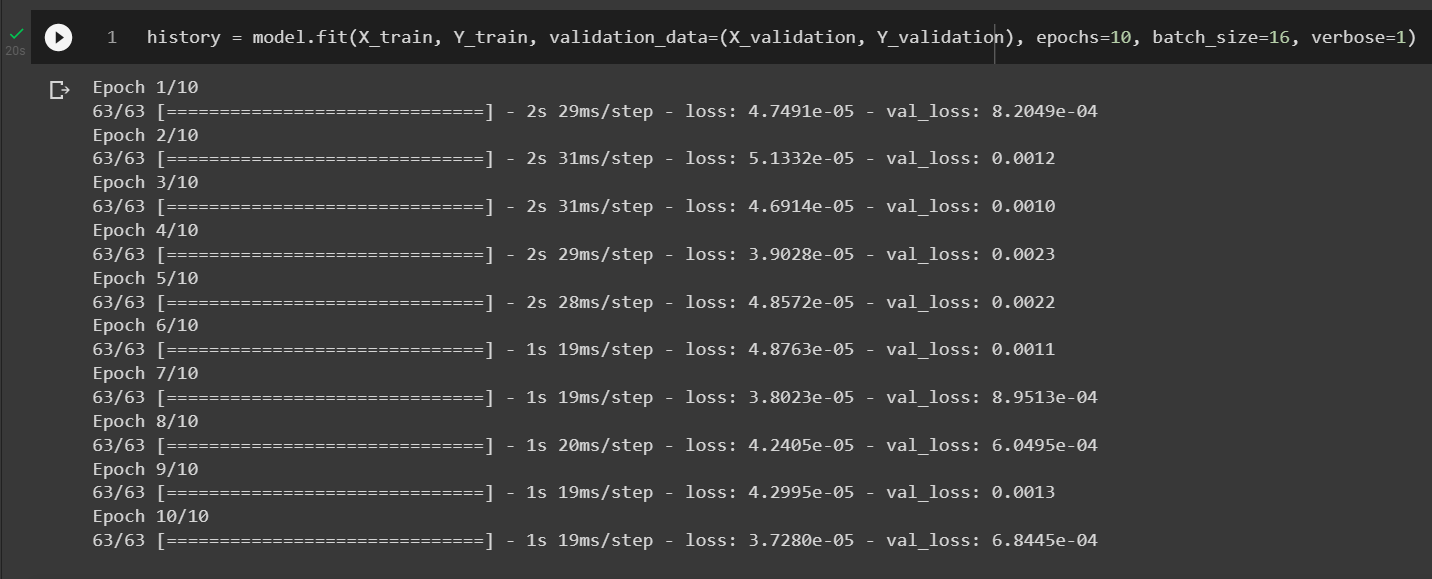


**Figure 12:** Init. LSTM model



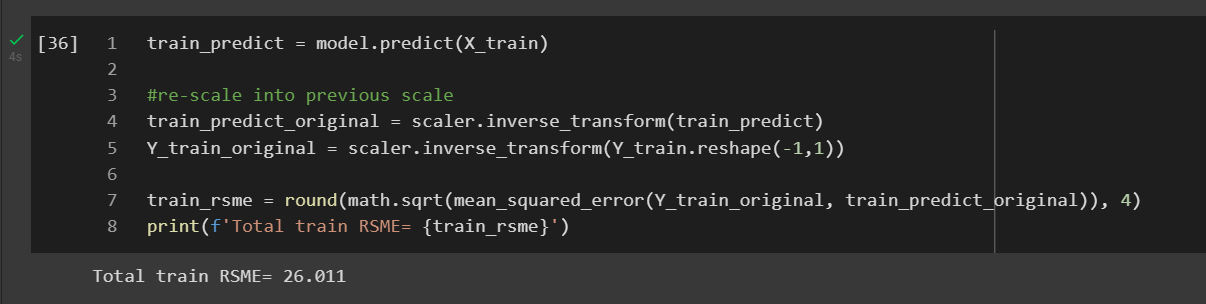
**Figure 13:** Summary of LSTM model

1. **Model Training:** In this portion I trained my model. The method fit has 6 parameters. I used the epochs and batch size as hyper parameter with some other parameter mention in the result analysis portion.

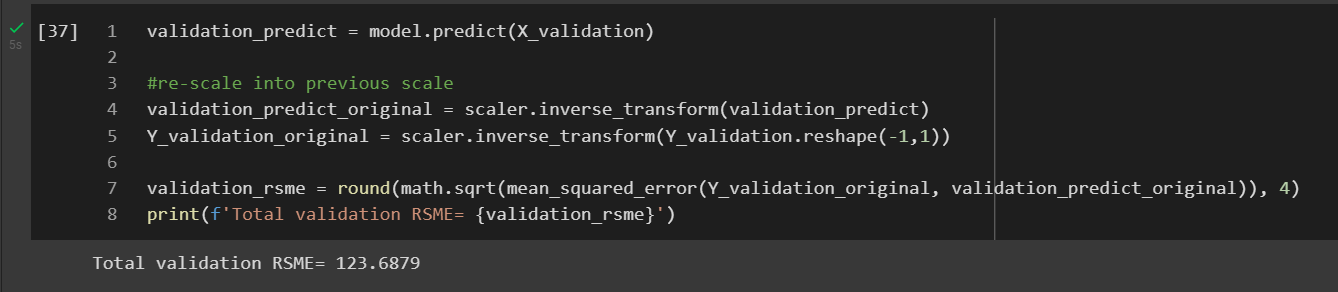


**Figure 14:** Fit the model

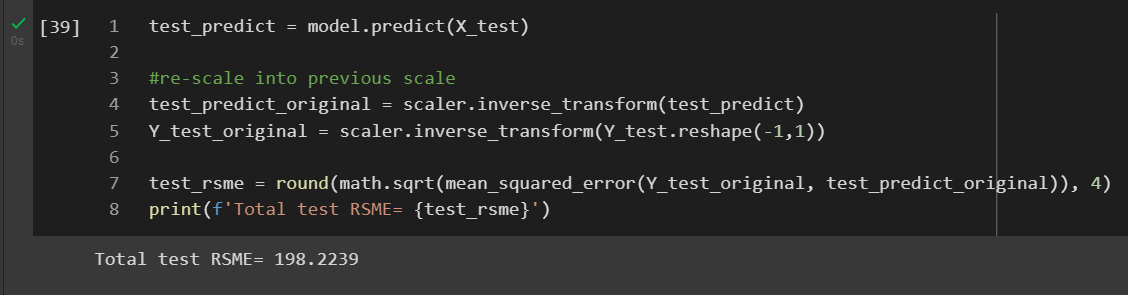
1. **Calculate RMSE values:** To calculate the RMSE value I need to re-scale the prediction and actual data and then used mean\_square\_error function to calculate the value.



**Figure 15:** calculate RMSE from training data

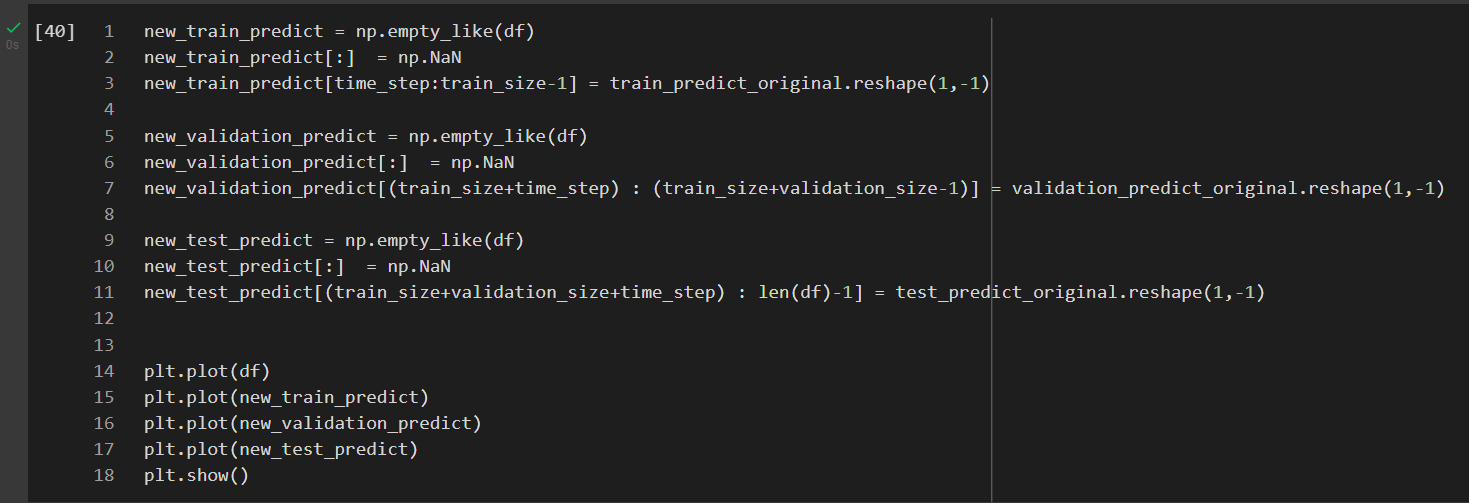


**Figure 16:** calculate RMSE from validation data



**Figure 17:** calculate RMSE from test data

1. **Plot the original vs. predication value:** I used pyplot function to plot the prediction value and predicted value.



**Figure 18:** code for drawing the actual and prediction data

|  |  |
| --- | --- |
|  |  |

**Figure 19:** visualization of actual and prediction data

**Result:** In this project I used 4 hyper parameters. They are time\_step, lstm\_unit, epocs, batch\_size.

* time\_step: This parameter for time series number of previous day for prediction. The values are 7, 14, and 28 days.
* lstm\_unit: Define the LSTM units in each LSTM layers and the values are 64, and 128.
* epochs: The number of epochs is the number of complete passes through the training dataset. Values are 100, and 500.
* batch\_size: The batch size is a number of samples processed before the model is updated. Values are 16, and 64.

As there are total 3, 2, 2, 2 numbers of parameter. So, I have to run total (3 x 2 x 2 x 2) = 24 times for each data set. And I have two dataset. So, I need to run my algorithm 48 times and the all possible values are given below:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SN** | | **Data Set** | **size** | **Time step** | **LSTM unit** | | **Epoch** | **batch size** | **RMSE Value** | | | | **graph** |
| **train** | **validation** | | **test** |
| 1 | | ETHEREUM | 1680 | 7 | 64 | | 100 | 16 | 32.7397 | 135.7386 | | 179.9898 | [ETH\_01.png](#fig_20) |
| 2 | | ETHEREUM | 1680 | 7 | 64 | | 100 | 64 | 27.991 | 157.5079 | | 346.8778 | [ETH\_02.png](#fig_21) |
| 3 | | ETHEREUM | 1680 | 7 | 64 | | 500 | 16 | 24.1677 | 1142.4956 | | 3589.8872 | [ETH\_03.png](#fig_22) |
| 4 | | ETHEREUM | 1680 | 7 | 64 | | 500 | 64 | 27.9442 | 180.9189 | | 349.129 | [ETH\_04.png](#fig_23) |
| 5 | | ETHEREUM | 1680 | 7 | 128 | | 100 | 16 | 32.2787 | 241.2323 | | 535.1814 | [ETH\_05.png](#fig_24) |
| 6 | | ETHEREUM | 1680 | 7 | 128 | | 100 | 64 | 29.4337 | 130.1892 | | 210.2508 | [ETH\_06.png](#fig_25) |
| 7 | | ETHEREUM | 1680 | 7 | 128 | | 500 | 16 | 25.7114 | 673.5333 | | 1763.0242 | [ETH\_07.png](#fig_26) |
| 8 | | ETHEREUM | 1680 | 7 | 128 | | 500 | 64 | 24.5529 | 154.3715 | | 322.0906 | [ETH\_08.png](#fig_27) |
| 9 | | ETHEREUM | 1680 | 14 | 64 | | 100 | 16 | 25.7608 | 167.3771 | | 306.2787 | [ETH\_09.png](#fig_28) |
| 10 | | ETHEREUM | 1680 | 14 | 64 | | 100 | 64 | 27.0288 | 166.5529 | | 368.1227 | [ETH\_10.png](#fig_29) |
| 11 | | ETHEREUM | 1680 | 14 | 64 | | 500 | 16 | 24.0209 | 352.0767 | | 950.0765 | [ETH\_11.png](#fig_30) |
| 12 | | ETHEREUM | 1680 | 14 | 64 | | 500 | 64 | 25.2772 | 171.015 | | 320.046 | [ETH\_12.png](#fig_31) |
| 13 | | ETHEREUM | 1680 | 14 | 128 | | 100 | 16 | 31.1989 | 181.9564 | | 383.8307 | [ETH\_13.png](#fig_32) |
| 14 | | ETHEREUM | 1680 | 14 | 128 | | 100 | 64 | 26.3979 | 167.5377 | | 327.0955 | [ETH\_14.png](#fig_33) |
| 15 | | ETHEREUM | 1680 | 14 | 128 | | 500 | 16 | 25.4947 | 575.5442 | | 1918.1468 | [ETH\_15.png](#fig_34) |
| 16 | | ETHEREUM | 1680 | 14 | 128 | | 500 | 64 | 25.2096 | 158.4983 | | 354.5904 | [ETH\_16.png](#fig_35) |
| 17 | | ETHEREUM | 1680 | 28 | 64 | | 100 | 16 | 25.3652 | 219.656 | | 440.0884 | [ETH\_17.png](#fig_36) |
| 18 | | ETHEREUM | 1680 | 28 | 64 | | 100 | 64 | 28.4892 | 203.3473 | | 455.5522 | [ETH\_18.png](#fig_37) |
| 19 | | ETHEREUM | 1680 | 28 | 64 | | 500 | 16 | 21.902 | 845.7621 | | 2059.503 | [ETH\_19.png](#fig_38) |
| 20 | | ETHEREUM | 1680 | 28 | 64 | | 500 | 64 | 24.269 | 190.8735 | | 419.5623 | [ETH\_20.png](#fig_39) |
| 21 | | ETHEREUM | 1680 | 28 | 128 | | 100 | 16 | 25.7966 | 150.3631 | | 297.699 | [ETH\_21.png](#fig_40) |
| 22 | | ETHEREUM | 1680 | 28 | 128 | | 100 | 64 | 26.3979 | 149.5504 | | 278.656 | [ETH\_22.png](#fig_41) |
| 23 | | ETHEREUM | 1680 | 28 | 128 | | 500 | 16 | 21.1161 | 651.9912 | | 1181.6978 | [ETH\_23.png](#fig_42) |
| 24 | | ETHEREUM | 1680 | 28 | 128 | | 500 | 64 | 27.0435 | 185.1505 | | 415.8409 | [ETH\_24.png](#fig_43) |
| For BIT-COIN data: | | | | | | | | | | | | | |
| 25 | | BITCOIN | 2829 | 7 | 64 | | 100 | 16 | 260.1122 | 362.4712 | | 4031.6923 | [BTC\_01.png](#fig_44) |
| 26 | | BITCOIN | 2829 | 7 | 64 | | 100 | 64 | 322.7874 | 409.0278 | | 4510.9961 | [BTC\_02.png](#fig_45) |
| 27 | | BITCOIN | 2829 | 7 | 64 | | 500 | 16 | 253.065 | 369.4141 | | 3086.9748 | [BTC\_03.png](#fig_46) |
| 28 | | BITCOIN | 2829 | 7 | 64 | | 500 | 64 | 261.1643 | 372.9401 | | 4166.797 | [BTC\_04.png](#fig_47) |
| 29 | | BITCOIN | 2829 | 7 | 128 | | 100 | 16 | 261.5801 | 364.0091 | | 3393.2677 | [BTC\_05.png](#fig_48) |
| 30 | | BITCOIN | 2829 | 7 | 128 | | 100 | 64 | 291.1744 | 483.1328 | | 3646.7425 | [BTC\_06.png](#fig_49) |
| 31 | | BITCOIN | 2829 | 7 | 128 | | 500 | 16 | 277.5947 | 383.6192 | | 5101.0248 | [BTC\_07.png](#fig_50) |
| 32 | | BITCOIN | 2829 | 7 | 128 | | 500 | 64 | 330.3293 | 377.9306 | | 8309.7074 | [BTC\_08.png](#fig_51) |
| 33 | | BITCOIN | 2829 | 14 | 64 | | 100 | 16 | 298.4092 | 454.4378 | | 2969.9365 | [BTC\_09.png](#fig_52) |
| 34 | | BITCOIN | 2829 | 14 | 64 | | 100 | 64 | 282.0142 | 367.1674 | | 4227.4353 | [BTC\_10.png](#fig_53) |
| 35 | | BITCOIN | 2829 | 14 | 64 | | 500 | 16 | 287.4418 | 401.0623 | | 11047.1719 | [BTC\_11.png](#fig_54) |
| 36 | | BITCOIN | 2829 | 14 | 64 | | 500 | 64 | 269.5564 | 370.1261 | | 7332.2935 | [BTC\_12.png](#fig_55) |
| 37 | | BITCOIN | 2829 | 14 | 128 | | 100 | 16 | 268.4907 | 378.7115 | | 4515.9218 | [BTC\_13.png](#fig_56) |
| 38 | | BITCOIN | 2829 | 14 | 128 | | 100 | 64 | 286.0174 | 385.6968 | | 2392.9758 | [BTC\_14.png](#fig_57) |
| 39 | | BITCOIN | 2829 | 14 | 128 | | 500 | 16 | 237.2501 | 403.4383 | | 22797.9305 | [BTC\_15.png](#fig_58) |
| 40 | | BITCOIN | 2829 | 14 | 128 | | 500 | 64 | 298.1996 | 497.6476 | | 6523.0263 | [BTC\_16.png](#fig_59) |
| 41 | | BITCOIN | 2829 | 28 | 64 | | 100 | 16 | 334.0174 | 512.9509 | | 6483.3487 | [BTC\_17.png](#fig_60) |
| 42 | | BITCOIN | 2829 | 28 | 64 | | 100 | 64 | 302.7308 | 498.1252 | | 4850.3347 | [BTC\_18.png](#fig_61) |
| 43 | | BITCOIN | 2829 | 28 | 64 | | 500 | 16 | 270.0163 | 389.4572 | | 41867.1592 | [BTC\_19.png](#fig_62) |
| 44 | | BITCOIN | 2829 | 28 | 64 | | 500 | 64 | 271.0288 | 387.5522 | | 4667.3347 | [BTC\_20.png](#fig_63) |
| 45 | | BITCOIN | 2829 | 28 | 128 | | 100 | 16 | 363.8542 | 522.943 | | 4832.1972 | [BTC\_21.png](#fig_64) |
| 46 | | BITCOIN | 2829 | 28 | 128 | | 100 | 64 | 283.0244 | 428.8339 | | 3599.4624 | [BTC\_22.png](#fig_65) |
| 47 | | BITCOIN | 2829 | 28 | 128 | | 500 | 16 | 257.7507 | 396.5924 | | 39996.7215 | [BTC\_23.png](#fig_66) |
| 48 | | BITCOIN | 2829 | 28 | 128 | | 500 | 64 | 271.4777 | 422.1988 | | 5872.7336 | [BTC\_24.png](#fig_67) |
| Figure-20: ETH\_1.png | | | | | Figure-21: ETH\_02.png | | | | | Figure-22: ETH\_03.png | | | |
| Figure-23: ETH\_04.png | | | | | Figure-24: ETH\_05.png | | | | | Figure-25: ETH\_06.png | | | |
| Figure-26: ETH\_07.png | | | | | Figure-27: ETH\_08.png | | | | | Figure-28: ETH\_09.png | | | |
| Figure-29: ETH\_10.png | | | | | Figure-30: ETH\_11.png | | | | | Figure-31: ETH\_12.png | | | |
| Figure-32: ETH\_13.png | | | | | Figure-33: ETH\_14.png | | | | | Figure-34: ETH\_15.png | | | |
| Figure-35: ETH\_16.png | | | | | Figure-36: ETH\_17.png | | | | | Figure-37: ETH\_18.png | | | |
| Figure-38: ETH\_19.png | | | | | Figure-39: ETH\_20.png | | | | | Figure-40: ETH\_21.png | | | |
| Figure-41: ETH\_22.png | | | | | Figure-42: ETH\_23.png | | | | | Figure-43: ETH\_24.png | | | |
| Figure-44: BTC\_01.png | | | | | Figure-45: BTC\_02.png | | | | | Figure-46: BTC\_03.png | | | |
| Figure-47: BTC\_04.png | | | | | Figure-48: BTC\_05.png | | | | | Figure-49: BTC\_06.png | | | |
| Figure-50: BTC\_07.png | | | | | [Figure-51: BTC\_08.png](#tab_32) | | | | | Figure-52: BTC\_09.png | | | |
| Figure-53: BTC\_10.png | | | | | [Figure-54: BTC\_11.png](#tab_35) | | | | | Figure-55: BTC\_12.png | | | |
| Figure-56: BTC\_13.png | | | | | Figure-57: BTC\_14.png | | | | | [Figure-58: BTC\_15.png](#tab_39) | | | |
| Figure-59: BTC\_16.png | | | | | Figure-60: BTC\_17.png | | | | | Figure-61: BTC\_18.png | | | |
| [Figure-62: BTC\_19.png](#tab_43) | | | | | Figure-63: BTC\_20.png | | | | | Figure-64: BTC\_21.png | | | |
| Figure-65: BTC\_22.png | | | | | [Figure-66: BTC\_23.png](#tab_47) | | | | | Figure-67: BTC\_24.png | | | |

**Discussions:** As crypto currencies are highly dependent on others factor (like world politics, socio-economic situation or even on some ones twitter), so it too difficult to predict the future price only with previous data. But if we look at the data, there may have some points.

1. For Ethereum data set, best accuracy gained when I use the parameter [(time = 7 days, LSTM unit = 64, epochs = 100, batch size = 16)](#tab_eth_1).
2. Using different parameters (on Ethereum data set), the RMSE value for test data are too high. But the shape of prediction is pretty much similar with the actual value except ([fig-22](#fig_22), [fig-26](#fig_26), [fig-30](#fig_30), [fig-34](#fig_34), [fig-38](#fig_38), and [fig-42](#fig_42)), where I used (epochs = 500, batch size = 16).
3. For Bitcoin the test RMSE value is huge 2392.9758, When applying the parameter [(time = 14 days, LSTM unit = 128, epochs = 100, batch size = 64)](#tab_btc_14). But in maximum case the actual price and predicted price has almost same shape except ([fig-51](#fig_51), [fig-54](#fig_54), [fig-58](#fig_58), [fig-62](#fig_62), [fig-66](#fig_66)).

References:

* <https://www.kaspersky.com/resource-center/definitions/what-is-cryptocurrency>
* <https://en.wikipedia.org/wiki/Long_short-term_memory>
* <https://en.wikipedia.org/wiki/Time_series>
* <https://machinelearningmastery.com/stacked-long-short-term-memory-networks/>
* <https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>
* <https://finance.yahoo.com/>
* Source code: <https://github.com/AhmedDiderRahat/crypto_price_prediction/tree/main/code>

***Consent:*** *Any one can use the code and other resources from this project.*