Bitcoin Price Prediction

Ashlesha Bansode
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
ashbansode@umbc.edu, prachib1@umbc.edu,pranavganore@umbc.edu
December 19, 2018

***Abstract*—The aim of our project was to predict bitcoin prices using empirical bitcoin dataset by leveraging one of the numerous machine learning techniques; of which we chose to use a variant of recurrent neural network algorithm known as 'Long short-term memory (LSTM)' algorithm. This paper spans the details about design, implementation and results along with the challenges which we faced while working on this project.**

***Keywords*—LSTM,Web scraper, Long Short Term Memory(LSTM), Web Scrapper, Bitcoin, Flask, Keras, Tensorflow**

I. Introduction

Being the most popular and trending crypto-currency in the market, we selected bitcoins as a crypto-currency choice for our prediction project. Bitcoin kick started the crypto revolution and it has completely changed the methods of handling money, making transactions or doing any kind of business. It all began with bitcoin. Bitcoin is rightly said notoriously volatile. Its price has increased 2000 per cent in 2017 and around 70% decrease in 2018. But since September, we can see a constant value at around $6,500 with slight fluctuations of around few hundred dollars.

However, the gist here is that, there is a lot of population who is interesting in following the trend of bitcoin price, as either they are the ones with stakes invested in bitcoins in the form of coins mined or are the ones who save their prospective interests towards bitcoins. Whatever be the reasons, be just for being a watchdog or be the peeping tom of bitcoin trend, be the financial enthusiasts planning to invest their assets in the world of cryptocurrency, all are interested in knowing how bitcoin prices will delve in future. For the very reason we thought to construct a time-series analysis-based prediction of bitcoin prices, enabling user to get the predictions till any dates they wish.

II. Related Work and literature survey

The website Bitcoin Price Prediction Tracker[1] was one of the sources which we found who provides a GUI similar to what we are providing i.e. providing the option to user to see prices say up to anydate till 2020, but the actual difference lies in how they are predicting prices, this website simply took prices for a certain period and calculated the % increase in bitcoin trend and incremented their prediction graph accordingly, on the other hand we are using a recurring neural net processes on real-time (or we can say based on the most recent bitcoin price) along with the entire empirical bitcoin data, which considers the increasing as well as decreasing trends that is considering all the treanding variations for the entire span since bitcoin came into existence into market.

This is another website [2] which provides prediction for 25 days ahead from next day, but does not give the user any flexibility to choose any specific date or neither shows any graphical details. However, not much can be known about how they actually predicted the prices, but one thing is obvious that the difference lies in the approach and how results are offered.

III. Dataset

CoinMarketCap[3] provides a datastore of prices along with few other related attributes of those cryptocurrencies for public. They have recorded the bitcoin data since early 2013 and keeps updating daily. We chose to scrape our data in Realtime from this source as it was logical to keep on dynamically expanding the dataset as the actual prices are exposed eventually.

Date	Open*	High	Low	Close**	Volume	Market Cap
Sep 24, 2018	6,704.77	6,713.56	6,580.90	6,595.41	4,177,310,000	115,889,159,338
Sep 23, 2018	6,715.32	6,766.15	6,679.42	6,710.63	4,197,500,000	116,058,736,872
Sep 22, 2018	6,735.05	6,814.56	6,616.80	6,721.98	4,509,660,000	116,386,818,047
Sep 21, 2018	6,513.87	6,794.33	6,496.36	6,734.95	6,531,940,000	112,552,829,858
Sep 20, 2018	6,398.85	6,529.26	6,395.95	6,519.67	4,348,110,000	110,553,109,981
Sep 19, 2018	6,371.85	6,448.46	6,208.34	6,398.54	4,431,340,000	110,074,137,715

Figure 1. Dataset

As in the above snapshot of how our dataset in consideration here looks like. Dataset tuples spans over 7 attribute columns for a date, with date as index. 'Open' and 'close' depicts the opening closing price of the bitcoin for that date respectively. It may be with reference to the stock exchange opening and closing time ranges or 12Am to 11.59PM time ranges which doesn't change much. However, 'High' here depicts the highest recorded price on that date and 'Low' represents the lowest recorded price on that day. Along with circulating supply and market capitalization, volume is one of the most prominent metrics in crypto.

The volume of any cryptocurrency is the total spot trading volume reported by all exchanges over the last 24 hours for that cryptocurrency. Some market pairs are excluded from the sum, denoted by two asterisks (**) on the markets tab, if the exchange does not enforce a trading fee or otherwise offers significant incentives to trade on the market pair. Market pairs with these characteristics are rather susceptible to wash trading, resulting in artificially inflated reported volumes. From our experience, we have found that it is better to exclude these markets to give a better representation of relative trading volumes for the crypto market.[4]

Market Capitalization is one way to rank the relative size of a cryptocurrency. It's calculated by multiplying the Price by the Circulating Supply.

Market Cap = Price * Circulating Supply.[5]

Volume is such an important metric when analyzing cryptos and it can help you in showing a coin's direction. The volume of a token listed on Comarkets is quite simple. It's the amount of the coin that has been traded in the last 24 hours. volume underscores how many people are buying and selling the coin. If the price of Bitcoin goes up and it shows a hefty volume, that tells us lots of people are making moves. Thus, it will likely keep going up. If the price of Bitcoin drops, but there's minimal volume, that could tell us only a small amount of people back the trend. Let's go into more detail on the ramifications. Volume is arguably the most important metric for a cryptocurrency, because of the amount of ways it can be broken down. From volume, you can infer the direction and movements of a coin. It's an essential metric for traders. Volume can have examined in minute detail. You can track volume on Comarkets by the last 24 hours, last week, or last 30 days. This helps reveal if a coin's recent swings are an aberration or the norm. A coin with frequent heavy movements won't attract attention if it has high volume. If a coin normally has less volume, heavy trading in the last 24 hours could indicate there's some support behind the move it may be making.

IV. Architecture

The figure below depicts the service oriented architecture of our system in the form of block diagram.

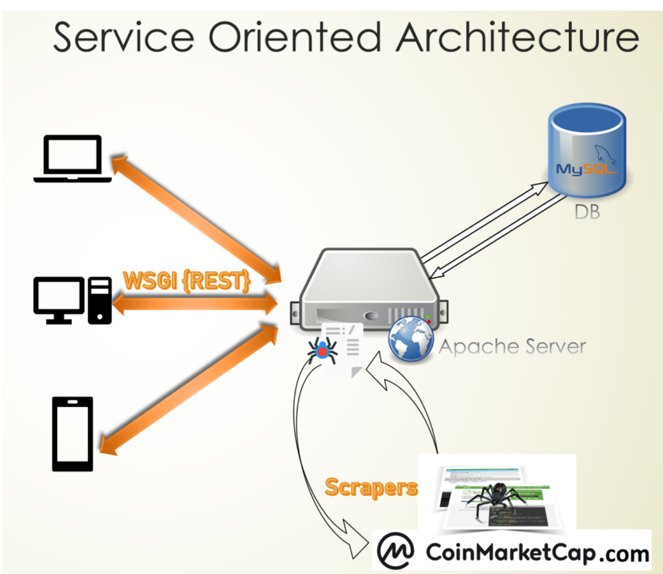


Figure 2. Architecture

We are providing users with an online web platform where they can either request for the daily (i.e. next day) prediction of the bitcoins in graphical format, or a user can select a specific date in future to get the daily bitcoin predictions till that date. Firstly to avail the prediction services, user needs to get an account signed up on our website. Below are the two screenshots of these Login and signup webpages.

Once a user login successfully, he/she is redirected to the input selection page where they can select what type of time range they want as output as seen in below fig

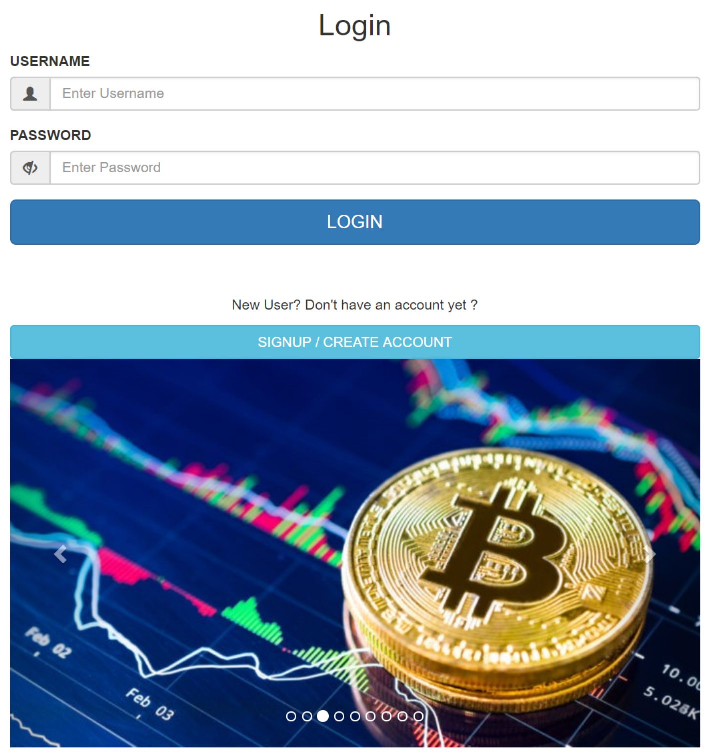


Figure 3. Login Page

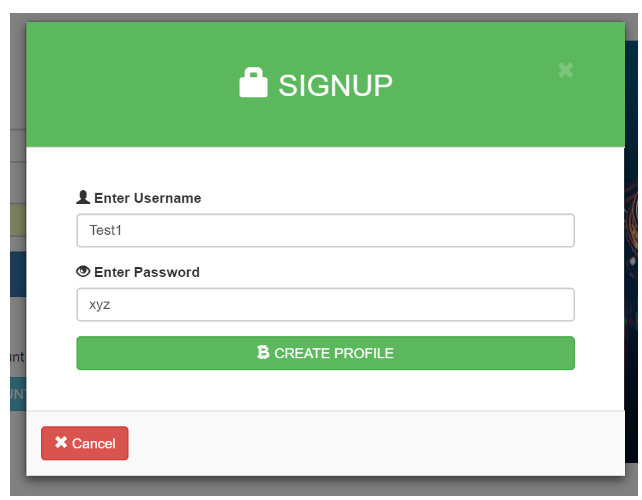


Figure 4. Signup page

Now when user submits the input, a POST REST call is made to our prediction service which invokes our prediction algorithm. The prediction service when called first invokes the scraper which gets the entire empirical bitcoin data from 28

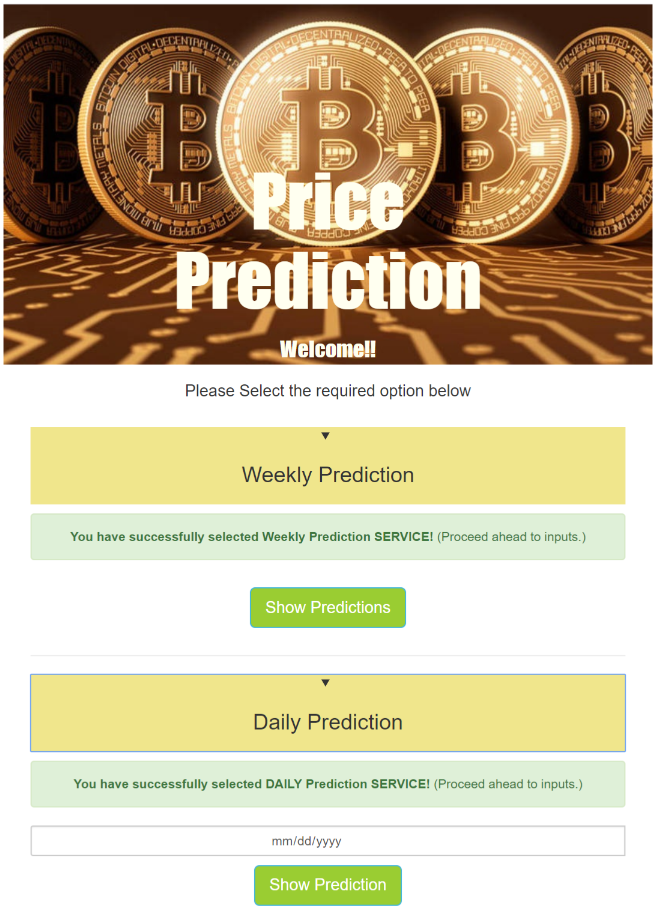


Figure 5. User Input Page

Apr 2013 to last day closing price (i.e. most recent) in Realtime in the format as specified in the section Dataset above.

For reference, Web crawling or spidering is the process of systematically extracting data from a website using a Web crawler, spider or robot. A Web scraper methodically harvests data from a website. And we have done a simple program which does content scraping using python on the webpage.[6]

Now once we have the updated dataset with us i.e. after scraper is done with fetching latest data, our LSTM algorithm works on this data and generates the graph for the prediction. The details of which can be traversed in the section Prediction model – LSTM.

All these things like frontend and backend and middleware computations are integrated together by FLASK framework. Flask is a micro web framework written in Python. It is classified as a microframework because it does not require tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program. Flask is commonly used with MongoDB, which gives it more control over databases and history.[7]

```
from flask import url_for, request
app = Flask(__name__)

@app.route('/index')
def index():
    return render_template('/logins.html')

@app.route('/success')
def success():
        return render_template('try_popup.html')

@app.route('/daily',methods = ['POST', 'GET'])
def daily():
    date = request.form['date_from']
    pff.daily(date)
    return render_template('Dated_Prediction.html')

@app.route('/weekly',methods = ['POST', 'GET'])
def weekly():
    pff.weekly()
    return render_template('Dated_Prediction.html')
```

Figure 6. Flask Code in Python for rendering templates after the request is routed

Flask runs on its inbuilt Apache server which hosts it. We have used MySQL database in backend for storing user credentials for authentication and user profile generation i.e. login mechanisms. (As constructing a login mechanism was not a goal here).

A. Technologies used

1) Bootstrap , HTML: Bootstrap is a free front-end framework for faster and easier web development. Bootstrap also gives you the ability to easily create responsive designs.

2) Backend : MySQL Server: MySQL is a relational SQL database Management System. It stores a collection of data.We have used MySQL database as the connectivity in python using python SQL connector is easily implemented. Our database save the details of the user information.We have created a database called bitcoin in mysql where we have a table maintained for user login and password details. Table name is users with columns as name and password.

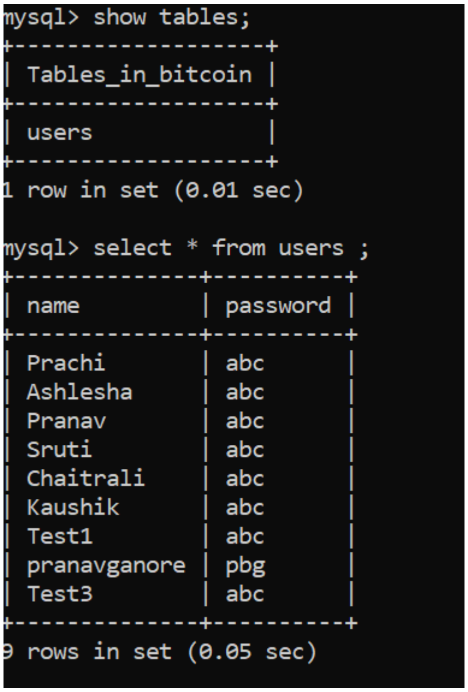
Connectivity code - cnx = mysql.connector.connect(user='abc', password='abc', host='127.0.0.1', database='Bitcoin')

3) Scraper : Python: We have used lxml which is a python package which deals with HTML.Its a HTML parser providing special API for HTML elements.It is used for processing common HTML tasks

4) Prediction Model : Python: Python provides many libraries to implement machine learning algorithm. Keras contains various numerical layering blocks used in many neutral network.It runs on top of TensorFlow.

V. Long Short Term Memory LSTM

Just like other Recurrent Neural Networks algorithms, LSTM is mostly used for Time Series Forecasting. The problems like sequence prediction and time-series forecasting



```
mysql> show tables;
+-------------------+
| Tables_in_bitcoin |
+-------------------+
| users             |
+-------------------+
1 row in set (0.01 sec)

mysql> select * from users ;
+-------------+----------+
| name        | password |
+-------------+----------+
| Prachi      | abc      |
| Ashlesha    | abc      |
| Pranav      | abc      |
| Sruti       | abc      |
| Chaitrali   | abc      |
| Kaushik     | abc      |
| Test1       | abc      |
| pranavganore | pbg     |
| Test3       | abc      |
+-------------+----------+
9 rows in set (0.05 sec)
```

Figure 7. MySQL

perfectly fits into this class of algorithm. Since our dataset for bitcoin is also a timeseries we preferred to implement Long short-term memory algorithm. We use Keras framework for deep learning. Our model consists of two stacked LSTM layers. The input is fed into the lower layer of LSTM and then the output of that layer is forwarded to the next layer. These layers are 256 units each and the densely connected output layer with one neuron. We are using Adam optimizer.

Our input to algorithm contained a scrapped dataset from coinMarket cap as explained in above sections. We preferred using the field closing prices per day in time series from our dataset to predict bitcoin prices.

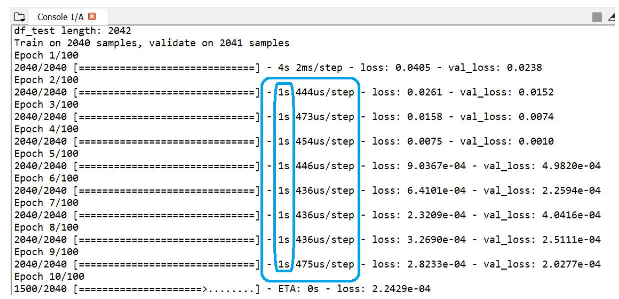
LSTM works on seasonal decomposition [6].It provides useful representation of time series generally and for better understanding and analyzing trend of the data in hand. Time series consists of three components what are called as systematic viz. level, trend, seasonality and one non-systematic component called noise. These components can be explained as follows:

1) Level: The average value in the series.
2) Seasonality: Cycle or pattern in series which is repeating after fixed period
3) Trend: Whether the flow of a value in series is increasing or decreasing.
4) Noise: Some variation in the series occurring randomly.

We used ‘statsmodels.tsa’ package in python to implement seasonal decomposition. The traces seasonality generated in our project are depicted in figure x.x of section x.x. In that figure, you can actually see the price movement as well as trend and seasonality.

As per the requirement of the LSTM, we divided the dataset into train and test. Since we are implementing sequential forecasting, we are predicting future values based on some previous as well as current values. So, our Y label is value from the next/future data points whereas the X input are those empirical dataset points which are already present in dataset. In our project, we have set the value of this parameter to 1 i.e. we are predicting the value of current data point in hand based on a previous value. Also, we have used an early stopping if the result doesn’t improve during 20 epochs i.e. iterations.

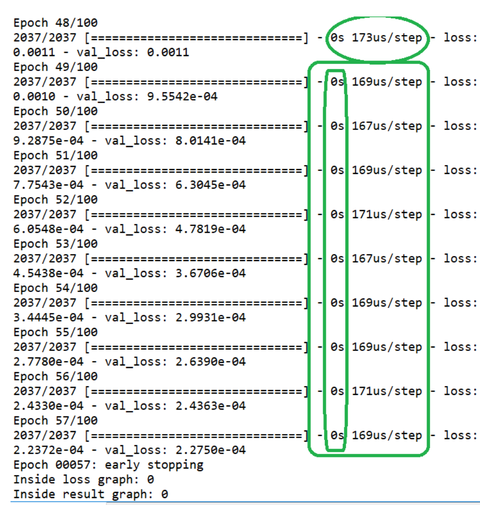
We performed several experiments and found that the optimal number of epochs and batch size is 90 and 100 respectively(As you can observe in the figure). Also, it is important to set shuffle value as ‘False’ because we don’t want to shuffle time series data which would obviously garble the output.




```
df_test length: 2042
Train on 2040 samples, validate on 2041 samples
Epoch 1/100
2040/2040 [==============================] - 4s 2ms/step - loss: 0.0405 - val_loss: 0.0238
Epoch 2/100
2040/2040 [==============================] - 1s 444us/step - loss: 0.0261 - val_loss: 0.0152
Epoch 3/100
2040/2040 [==============================] - 1s 473us/step - loss: 0.0158 - val_loss: 0.0074
Epoch 4/100
2040/2040 [==============================] - 1s 454us/step - loss: 0.0075 - val_loss: 0.0010
Epoch 5/100
2040/2040 [==============================] - 1s 446us/step - loss: 9.0367e-04 - val_loss: 4.9820e-04
Epoch 6/100
2040/2040 [==============================] - 1s 436us/step - loss: 6.4101e-04 - val_loss: 2.2594e-04
Epoch 7/100
2040/2040 [==============================] - 1s 436us/step - loss: 2.3209e-04 - val_loss: 4.0416e-04
Epoch 8/100
2040/2040 [==============================] - 1s 436us/step - loss: 3.2690e-04 - val_loss: 2.5111e-04
Epoch 9/100
2040/2040 [==============================] - 1s 475us/step - loss: 2.8233e-04 - val_loss: 2.0277e-04
Epoch 10/100
1500/2040 [====================>........] - ETA: 0s - loss: 2.2429e-04
```

Figure 8. Batch size = 100 and Epoch = 90

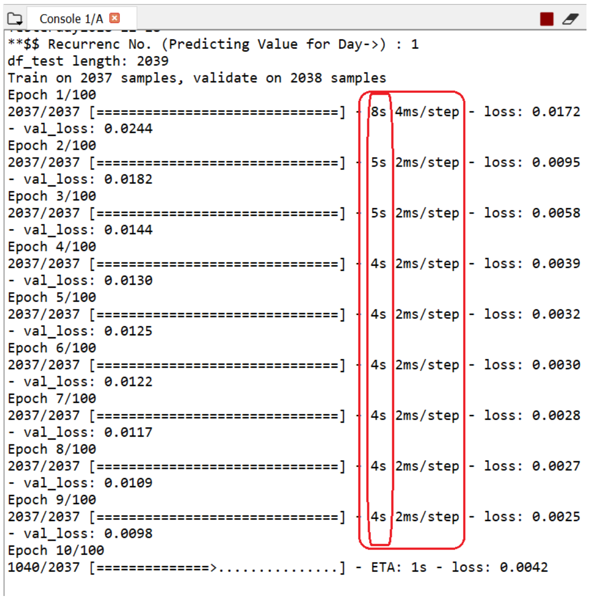
We also observed that increasing the batch size in the multiples of 10 would make the epochs run much faster, for example considerably reducing the time for single epoch when batch is set to 100, from 1-4 seconds to one seventy micro seconds per epoch step on average when batch is set to 10,000(As you can observe in figure). This is obvious because batch size specifies how many records will the LSTM process at a time in a single iteration, but this affects the final predicted output which is underfits with respect to the test data. Keeping the batch size less than 100, produces slightly better



```
Epoch 48/100
2037/2037 [==============================] - 0s 173us/step - loss:
0.0011 - val_loss: 0.0011
Epoch 49/100
2037/2037 [==============================] - 0s 169us/step - loss:
0.0010 - val_loss: 9.5542e-04
Epoch 50/100
2037/2037 [==============================] - 0s 167us/step - loss:
9.2875e-04 - val_loss: 8.0141e-04
Epoch 51/100
2037/2037 [==============================] - 0s 169us/step - loss:
7.7543e-04 - val_loss: 6.3045e-04
Epoch 52/100
2037/2037 [==============================] - 0s 171us/step - loss:
6.0548e-04 - val_loss: 4.7819e-04
Epoch 53/100
2037/2037 [==============================] - 0s 167us/step - loss:
4.5438e-04 - val_loss: 3.6706e-04
Epoch 54/100
2037/2037 [==============================] - 0s 169us/step - loss:
3.4445e-04 - val_loss: 2.9931e-04
Epoch 55/100
2037/2037 [==============================] - 0s 169us/step - loss:
2.7780e-04 - val_loss: 2.6390e-04
Epoch 56/100
2037/2037 [==============================] - 0s 171us/step - loss:
2.4330e-04 - val_loss: 2.4363e-04
Epoch 57/100
2037/2037 [==============================] - 0s 169us/step - loss:
2.2372e-04 - val_loss: 2.2750e-04
Epoch 00057: early stopping
Inside loss graph: 0
Inside result graph: 0
```

Figure 9. Batch size = 10000 and Epoch = 100

predictions but not considerably skewed out to be considered for compromising on the speed of epochs by reducing the batch size. Hence, we concluded after running multiple experiments



```
**$$ Recurrenc No. (Predicting Value for Day->) : 1
df_test length: 2039
Train on 2037 samples, validate on 2038 samples
Epoch 1/100
2037/2037 [==============================] - 8s 4ms/step - loss: 0.0172
- val_loss: 0.0244
Epoch 2/100
2037/2037 [==============================] - 5s 2ms/step - loss: 0.0095
- val_loss: 0.0182
Epoch 3/100
2037/2037 [==============================] - 5s 2ms/step - loss: 0.0058
- val_loss: 0.0144
Epoch 4/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0039
- val_loss: 0.0130
Epoch 5/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0032
- val_loss: 0.0125
Epoch 6/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0030
- val_loss: 0.0122
Epoch 7/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0028
- val_loss: 0.0117
Epoch 8/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0027
- val_loss: 0.0109
Epoch 9/100
2037/2037 [==============================] - 4s 2ms/step - loss: 0.0025
- val_loss: 0.0098
Epoch 10/100
1040/2037 [==============>..............] - ETA: 1s - loss: 0.0042
```

Figure 10. Batch size = 16 and Epoch = 100

that keeping batch size to 100 and epochs to 90 produces probably the predicted prices closer to the actual ones with respect to trend as well as price values.

VI. Experimental Results

The figure below shows trend and seasonality of the bitcoin historical data. The four traces in this graph depicts Trend, Observed, seasonal and residual values.

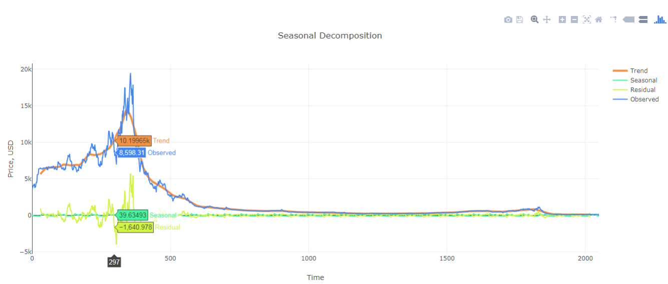


Figure 11. Seasonal Decomposition

In the below graph, Fig. Train and Test Loss during training, we are comparing the train loss and test loss on each iteration of the training process. After some iterations, train and test loss becomes very similar, which is positive.

For the next graph, we used our model to predict labels during training on entire training data.

The Fig. 14 below shows result of our bitcoin price prediction model for the future days from the current day (As of 25th November 2018).

Figure 12. Train and Test Loss during training

Figure 13. Comparison of prices with prices our model predicted

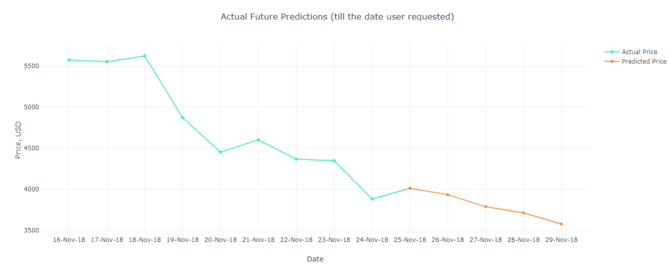


Figure 14. Actual future Predictions (till the date user requested) recorded on 25th November 2018

The fig. 15 below shows result of our bitcoin price prediction model for the future days from the current day (As of 2nd December 2018).

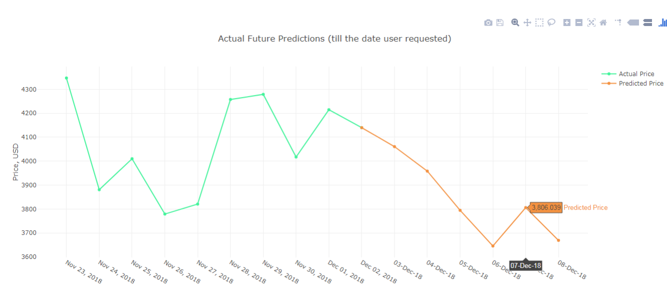


Figure 15. Actual future Predictions (till the date user requested)recorded on 2nd December 2018

VII. Challenges

1) Accuracy in Bitcoin price prediction: Since we know that Bitcoin prices keeps on fluctuating based on

many factors like news, economy etc, sometimes the sudden rise or drop in Bitcoin prices can not be accurately predicted. Improving the accuracy of the model was therefore a biggest challenge.

2) Execution Time: Reducing the execution time of machine learning model was another challenge. Data being huge and increasing the number of epochs for accuracy required most of the execution time.
3) Batch and Epoch: Fine tune the prediction outputs by finding the correct and most suitable combination of batch size and number of epochs along with keeping optimum speed as low as possible.

VIII. Future Scope

Bitcoin price is volatile and is dependent on many factors like stock Market, news, politics etc. Considering these factors was beyond the scope of this project. We would like to work on improving the bitcoin price prediction by taking into consideration above mentioned factors.

IX. Conclusion

Finally, we were able to predict Bitcoin prices using the recurrent neural network's - Long Short Term Memory and there by it was observed that these predictions closely followed the trend of the empirical Bitcoin prices along with the recurrent previous day predictions given as an actual data input for the future dates(Thereby obviously justifying the working of RNN-LSTM). Although the prediction results were based upon the time series analysis as a feature under consideration, but the other numerous factors which directly and indirectly influence the prices of Bitcoins in real time were not considered for building the predictions. Thus, the results are subject to these factors.

Acknowledgment

We would like to express our sincere gratitude to Dr. Milton Halem for his guidance and sharing his knowledge throughout the semester. We would also like to thank Samit Shivadekar and Sai Sree Laya Chukkapalli for their valuable support. During implemention of our project there were many concepts which were taught in the class that helped us.

We learned tremendous amount of things from you in this short period.

References

[1] Bitcoin Price Prediction Tracker(Bircoin) https://bircoin.top/

[2] Bitcoin prediction for 25 days http://30rates.com/btc-to-usd-forecast-today-dollar-to-bitcoin

[3] Dataset https://coinmarketcap.com/

[4] Trading volumes for crypto markets https://coinmarketcap.com/methodology/market-data

[5] Market Cap FAQ https://coinmarketcap.com/faq/

[6] scraping https://coinmarketcap.com/currencies/bitcoin/historical-data

[7] Flask https://en.wikipedia.org/wiki/Flask_(web_framework)

[8] LSTM https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/

[9] Taylor SJ, Letham B. (2017) Forecasting at scale. PeerJ Preprints 5:e3190v2 https://doi.org/10.7287/peerj.preprints.3190v2

[10] Huisu Jang and Jaewook Lee, *An Empirical Study on Modelling and Prediction of Bitcoin Prices with Bayesian Neural Networks based on Blockchain Information* in IEEE Early Access Articles, 2017, vol.99, pp. 1-1

[11] D. Shah and K. Zhang, "Bayesian regression and Bitcoin," in 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2015, pp. 409-415.