# B565 DATA MINING PROJECT REPORT

## TITLE:

Bitcoin Prediction using RNN (recurrent neural network) and LSTM (Long Short-Term Memory)

## AUTHORS:

Vrushab Hanumesh (vvrushab) 2000917736

Tejasram Ramesh (terame) 2000902539

## ABSTRACT:

In this project, we attempt to predict bitcoin price accurately using various parameters affecting the bitcoin price, here we are using dataset containing mainly 5 features open, high, low, close, volume. We are using deep learning and neural nets to solve this problem, the technologies used are RNN (recurrent neural network) and LSTM (Long Short-Term Memory). The goal is to predict the bitcoin value accurately by using the historical dataset from last 7 years.

## KEYWORDS:

Bitcoin, cryptocurrency, price prediction, finance, deep learning, neural networks, recurrent neural network, Long Short-Term Memory.

## INTRODUCTION:

The trend of Decentralized Finance has led to the emergence of new cryptocurrencies like Bitcoin. The anonymity and ease of purchase has attracted a lot of investors. In Decentralized finance technology is at forefront instead of institutions like banks. Institutions have to deal with market regulations, watchdogs, shareholders, Central Bank policies, Market currency circulation etc. On the other hand, decentralized finance works on smart contracts between buyers and sellers directly. The trade is recorded in a public blockchain ledger, this ensures transparency.

There is a stark difference between stock market and Cryptocurrency market. Stock market gives the investor a real asset (percentage of ownership of the company), depending on the company's performance and sales cycles the stock prices also increase and decrease. In many cases they are cyclic. This is not the case with cryptocurrency where the price increases due to reduction in volume of mining of coins and increase in demand. It runs on demand supply more than other factors. In recent times we have seen public figures become advocates for

cryptocurrency and they have been able to wield considerable influence over the crypto markets. This is an interesting phenomenon for Data scientist, who want to apply machine learning or deep learning in cryptomarkets (Algo trading, HFT is done in stock markets).

Due these qualities of the cryptocurrency, Deep Learning is a suitable solution for prediction of cryptocurrency. Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) are good at handling temporal dependencies.

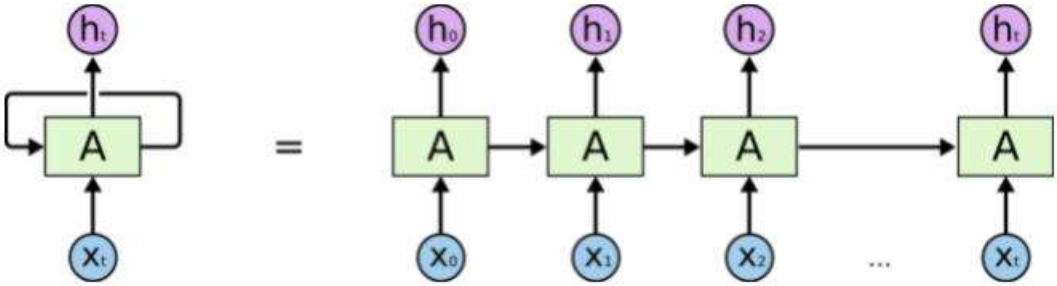
## METHODS:

Our endeavor is to help a cryptocurrency investor make an investment on a coin that will fetch him the best returns in the time duration he desires. The model will help the investor make the right decision in terms of which asset he must purchase (Purchase currency like Bitcoin, Ethereum etc.) and how much return he can expect over a period of time by comparing the growths or downfalls of various crypto currencies.

We extrapolate these results based on real time and historical data that has been fetched from: https://finance.yahoo.com/quote/BTC-USD/

### RECURRENT NEURAL NETWORKS

A traditional Neural network will not be able to make decisions that depend on the previous outputs accurately. This is solved by Recurrent Neural Networks where we pass the output from previous computation back into the layer again. It uses a single activation layer and can be used for translation, forecasting, speech recognition.




### DRAWBACKS OF RNN

Despite RNN having feedback loops that help it maintain a memory over time, its memory capacity is not enough to accurately handle long term temporal dependencies. This is because the gradient of loss function decays exponentially with time.

Example. If I had to predict the next word in the line “the clouds are in the “the next word would be sky.

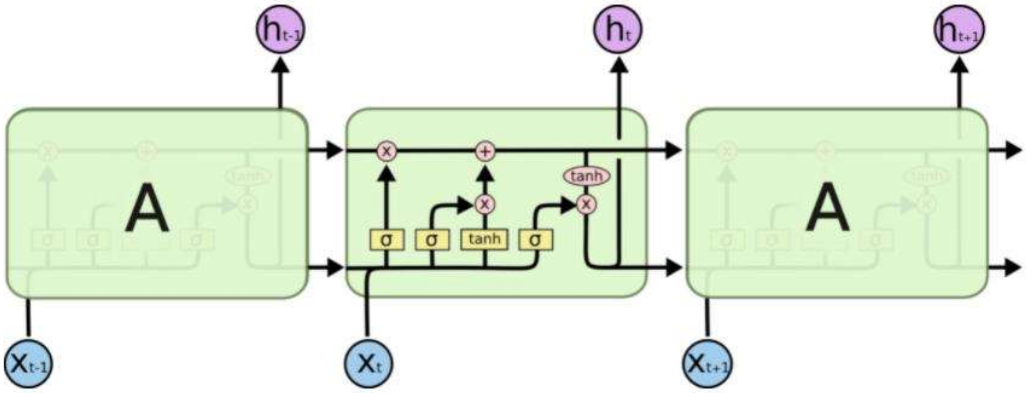
If i had to add more context to it, like “I grew up in France… I speak fluent”, I want the word French, it might give another language. If the gap for the context is small RNN works else it doesn't perform well.

To resolve this issue, we use Long Short-term Memory

### LONG SHORT-TERM MEMORY

Long Short-term Memory is a special type of RNN where there is a presence of an additional memory unit that can maintain memory over a long period of time. Instead of having just a Single Neural Network it has 4.

LSTM deal with the vanishing gradient issue through input and forget gates, which aid in better control over the gradient flow and aid in preservation of long-range dependence.




## IMPLEMENTATION

### STEP 1: IMPORTING THE DATASET AND PREPROCESSING

Data source: https://finance.yahoo.com/quote/BTC-USD/

Fetching the data:

```
data = pd.read_csv('BTC-USD.csv', date_parser = True)
```

```
data.tail()
```

| | Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|---|
| **2075** | 2020-05-23 | 9185.062500 | 9302.501953 | 9118.108398 | 9209.287109 | 9209.287109 | 2.772787e+10 |
| **2076** | 2020-05-24 | 9212.283203 | 9288.404297 | 8787.250977 | 8790.368164 | 8790.368164 | 3.251880e+10 |
| **2077** | 2020-05-25 | 8786.107422 | 8951.005859 | 8719.667969 | 8906.934570 | 8906.934570 | 3.128816e+10 |
| **2078** | 2020-05-26 | NaN | NaN | NaN | NaN | NaN | NaN |
| **2079** | 2020-05-27 | 8834.157227 | 8859.578125 | 8834.157227 | 8856.885742 | 8856.885742 | 2.914432e+10 |

## STEP 2: SPLITTING THE TEST AND TRAIN DATA

```
[3] dataset_training = dataset[dataset['Date']< '2020-01-10'].copy()
    dataset_training
```

| | Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|---|
| **0** | 2014-09-17 | 465.864014 | 468.174011 | 452.421997 | 457.334015 | 457.334015 | 2.105680e+07 |
| **1** | 2014-09-18 | 456.859985 | 456.859985 | 413.104004 | 424.440002 | 424.440002 | 3.448320e+07 |
| **2** | 2014-09-19 | 424.102997 | 427.834991 | 384.532013 | 394.795990 | 394.795990 | 3.791970e+07 |
| **3** | 2014-09-20 | 394.673004 | 423.295990 | 389.882996 | 408.903992 | 408.903992 | 3.686360e+07 |
| **4** | 2014-09-21 | 408.084991 | 412.425995 | 393.181000 | 398.821014 | 398.821014 | 2.658010e+07 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **1936** | 2020-01-05 | 7410.451660 | 7544.497070 | 7400.535645 | 7411.317383 | 7411.317383 | 1.972507e+10 |
| **1937** | 2020-01-06 | 7410.452148 | 7781.867188 | 7409.292969 | 7769.219238 | 7769.219238 | 2.327626e+10 |
| **1938** | 2020-01-07 | 7768.682129 | 8178.215820 | 7768.227539 | 8163.692383 | 8163.692383 | 2.876729e+10 |
| **1939** | 2020-01-08 | 8161.935547 | 8396.738281 | 7956.774414 | 8079.862793 | 8079.862793 | 3.167256e+10 |
| **1940** | 2020-01-09 | 8082.295898 | 8082.295898 | 7842.403809 | 7879.071289 | 7879.071289 | 2.404599e+10 |

1941 rows × 7 columns

```
[4] dataset_test = dataset[dataset['Date']> '2020-01-10'].copy()
    dataset_test
```

| | Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|---|
| **1942** | 2020-01-11 | 8162.190918 | 8218.359375 | 8029.642090 | 8037.537598 | 8037.537598 | 2.552117e+10 |
| **1943** | 2020-01-12 | 8033.261719 | 8200.063477 | 8009.059082 | 8192.494141 | 8192.494141 | 2.290344e+10 |
| **1944** | 2020-01-13 | 8189.771973 | 8197.788086 | 8079.700684 | 8144.194336 | 8144.194336 | 2.248291e+10 |
| **1945** | 2020-01-14 | 8140.933105 | 8879.511719 | 8140.933105 | 8827.764648 | 8827.764648 | 4.484178e+10 |
| **1946** | 2020-01-15 | 8825.343750 | 8890.117188 | 8657.187500 | 8807.010742 | 8807.010742 | 4.010283e+10 |
| **...** | ... | ... | ... | ... | ... | ... | ... |

## STEP 3: APPLYING LSTM

Here tanh and sigmoid activation function is used

```
[11] from tensorflow.keras import Sequential
     from tensorflow.keras.layers import Dense, LSTM, Dropout
```

Applying LSRM

```
[12] reg = Sequential()
     reg.add(LSTM(units = 60, activation = 'tanh', return_sequences = True, input_shape = (X_train.shape[1], 5)))
     reg.add(Dropout(0.2))
```

```
     reg.add(LSTM(units = 70, activation = 'sigmoid', return_sequences = True))
     reg.add(Dropout(0.4))

     reg.add(LSTM(units = 90, activation = 'sigmoid', return_sequences = True))
     reg.add(Dropout(0.5))

     reg.add(LSTM(units = 120, activation = 'sigmoid'))
     reg.add(Dropout(0.6))

     reg.add(Dense(units =1))
```

**STEP 4 : FITTING THE MODEL**

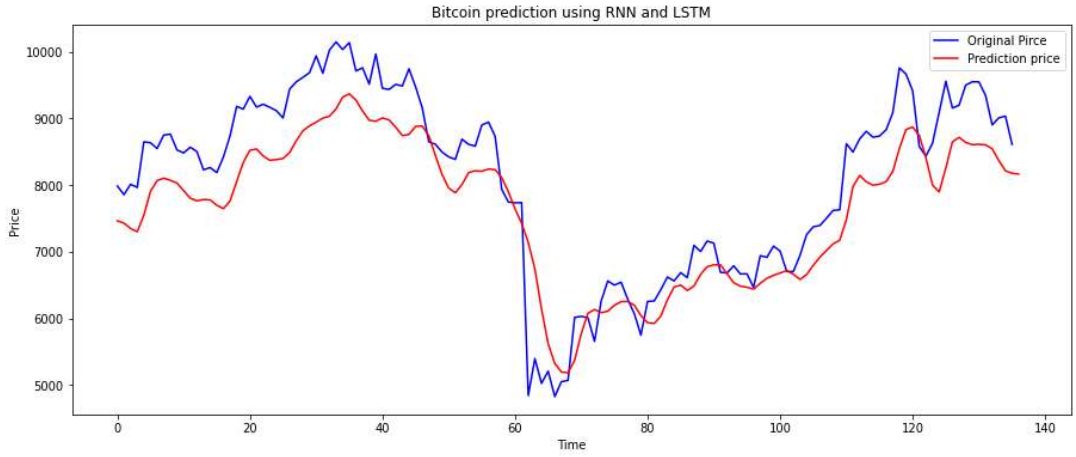
regressor.fit(X_train, Y_train, epochs = 100, batch_size =50)

```
[44] regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

     regressor.fit(X_train, Y_train, epochs = 100, batch_size =50)

Epoch 1/100
38/38 [==============================] - 13s 229ms/step - loss: 0.0288
Epoch 2/100
38/38 [==============================] - 8s 221ms/step - loss: 0.0063
Epoch 3/100
38/38 [==============================] - 8s 223ms/step - loss: 0.0047
Epoch 4/100
38/38 [==============================] - 9s 225ms/step - loss: 0.0046
Epoch 5/100
38/38 [==============================] - 9s 225ms/step - loss: 0.0047
Epoch 6/100
38/38 [==============================] - 9s 223ms/step - loss: 0.0039
Epoch 7/100
```

**RESULTS:**




**FUTURE WORK:**

**Just predicting the crypto value will not be of much use to the customer hence we will implement a function to find the return for X years from now for a specific crypto currency.**

So here we create a return function when it will subtract the predicted value – the value of bitcoin when he invested. This will return his total profit.

**The code here is just for the BTC, we will extend this code to other cryptocurrencies as well, like eth,doge,lite etc.**

As this is a standard template we can just change the dataset by downloading the details of eth, doge, lite etc from yahoo finance site.

**Prepare a detailed report on this and submit it by the deadline date.**

**REFERENCES:**


https://www.jakob-aungiers.com/articles/a/Multidimensional-LSTM-Networks-to-Predict-Bitcoin-Price

https://keras.io/#keras-the-python-deep-learning-library

https://www.tensorflow.org/

http://www.bioinf.jku.at/publications/older/2604.pdf

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

https://dashee87.github.io/deep%20learning/python/predicting-cryptocurrency-prices-with-deep-learning/

https://towardsdatascience.com/cryptocurrency-price-prediction-using-deep-learning-70cfca50dd3a

https://www.analyticsvidhya.com/blog/2021/05/bitcoin-price-prediction-using-recurrent-neural-networks-and-lstm/

https://www.coursera.org/learn/nlp-sequence-models?specialization=deep-learning#syllabus

https://ashutoshtripathi.com/2021/07/02/what-is-the-main-difference-between-rnn-and-lstm-nlp-rnn-vs-lstm/

https://finance.yahoo.com/quote/BTC-USD?p=BTC-USD

http://202.62.95.70:8080/jspui/bitstream/123456789/12673/1/1NH16CS719.pdf


RESEARCH PAPERS:


https://www.researchgate.net/publication/342743088_Bitcoin_price_forecasting_method_based_on_CNN-LSTM_hybrid_neural_network_model

https://www.researchgate.net/publication/339092042_A_LSTM-Method_for_Bitcoin_Price_Prediction_A_Case_Study_Yahoo_Finance_Stock_Market

https://ieeexplore.ieee.org/document/8938251

https://escholarship.org/uc/item/70d9n5sd

https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43905.pdf

https://www.sciencedirect.com/science/article/pii/S1877050920304865