# The Team

FSP4

Team - 9

Tejas

Arushi

Raghav

# FORECASTING CRYPTOCURRENCY TRENDS

While comparing the effectiveness of different LSTMs

# PROBLEM STATEMENT

Given the volatility of cryptocurrencies, how can we forecast the trend in order to make informed investment decisions?

# DATASET

The Alpha Vantage API offers historical and real-time data for stocks, forex, and cryptocurrencies. Several time frames are available ranging from 1-minute bars up to monthly.

# Data Collection and Data Cleaning

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

**Step 1**

Extracted data from Alpha Vantage

**Step 2**

Organised data as per requirements

**Step 3**

Feature Scaling using MinMaxScaler

# Cryptocurrencies by relevance and market cap:

```
Enter which cryptocurrency you would like to forecast (Integer between 1 and 10):
      1. Bitcoin (BTC)
      2. Ethereum (ETH)
      3. Binance Coin (BNB)
      4. Ripple (XRP)
      5. Cardano (ADA)
      6. Litecoin
      7. Lumen (XLM)
      8. EOS.IO (EOS)
      9. QTUM (QTUM)
      10. TRON (TRX)
```

In [52]:
```
cryptolist_ar = ['BTC','ETH','BNB','XRP','ADA','LTC','XLM','EOS','QTUM','TRX']
```

In [53]:
```
import itertools

for (i,j) in zip(cryptolist_ar, crypto_dflist):
    cleaned_data[i + ' close(USD)'] = j['close (USD)']
    cleaned_data[i + ' mktcap(USD)'] = j['market cap (USD)']
```

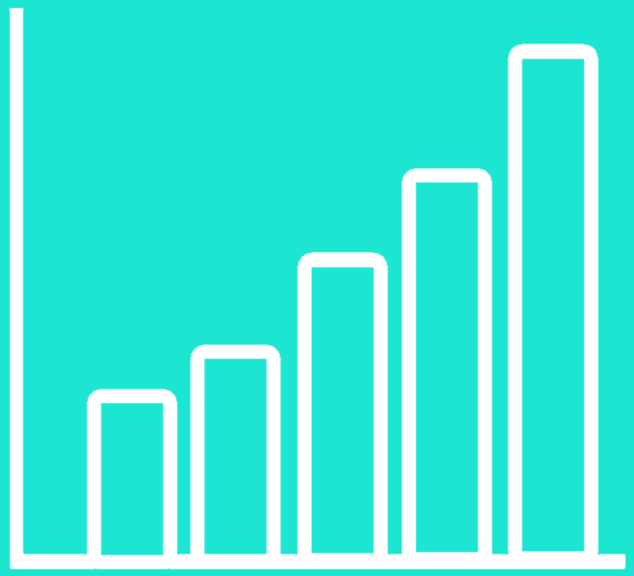
In [54]:
```
cleaned_data.head()
```

Out[54]:

| | timestamp | BTC close(USD) | BTC mktcap(USD) | ETH close(USD) | ETH mktcap(USD) | BNB close(USD) | BNB mktcap(USD) | XRP close(USD) | XRP mktcap(USD) | ADA close(USD) | ... | LTC close(USD) | mktcap( |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2021-04-21 | 56395.68 | 1609.784422 | 2350.71 | 4.141744e+04 | 576.7544 | 426547.695 | 1.40664 | 3.694768e+07 | 1.26773 | ... | 267.23 | 6.471902 |
| 1 | 2021-04-20 | 56425.00 | 72744.482151 | 2330.03 | 9.922408e+05 | 586.3635 | 5730895.325 | 1.38501 | 1.522196e+09 | 1.26689 | ... | 260.68 | 2.111225 |
| 2 | 2021-04-19 | 55633.14 | 78229.042267 | 2161.12 | 8.205923e+05 | 504.0322 | 5031325.713 | 1.30945 | 1.608074e+09 | 1.19450 | ... | 261.38 | 1.973512 |
| 3 | 2021-04-18 | 56150.01 | 124882.131824 | 2235.64 | 1.475384e+06 | 481.4367 | 4468597.460 | 1.40797 | 2.048345e+09 | 1.27693 | ... | 273.36 | 3.346590 |
| 4 | 2021-04-17 | 60006.66 | 58912.256128 | 2317.60 | 6.242323e+05 | 514.6861 | 2949040.221 | 1.53896 | 1.108826e+09 | 1.36802 | ... | 300.86 | 2.574520 |

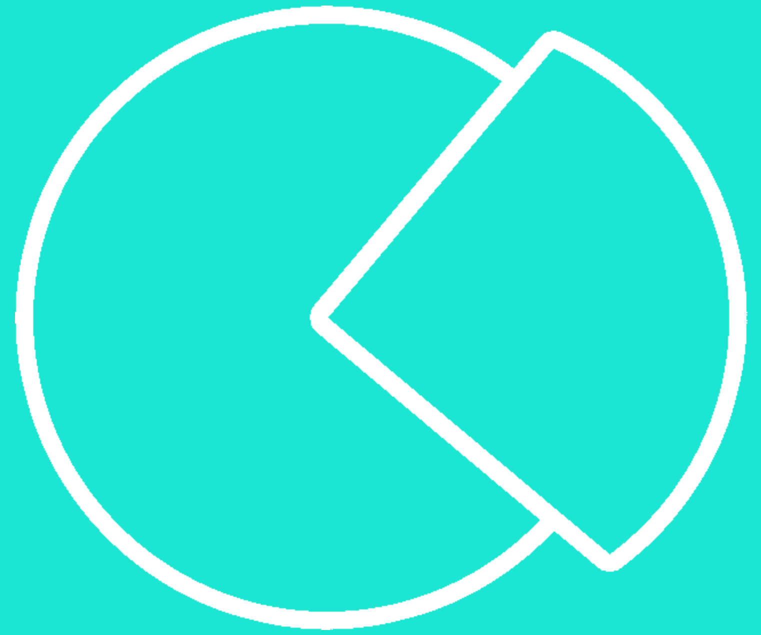
5 rows × 21 columns

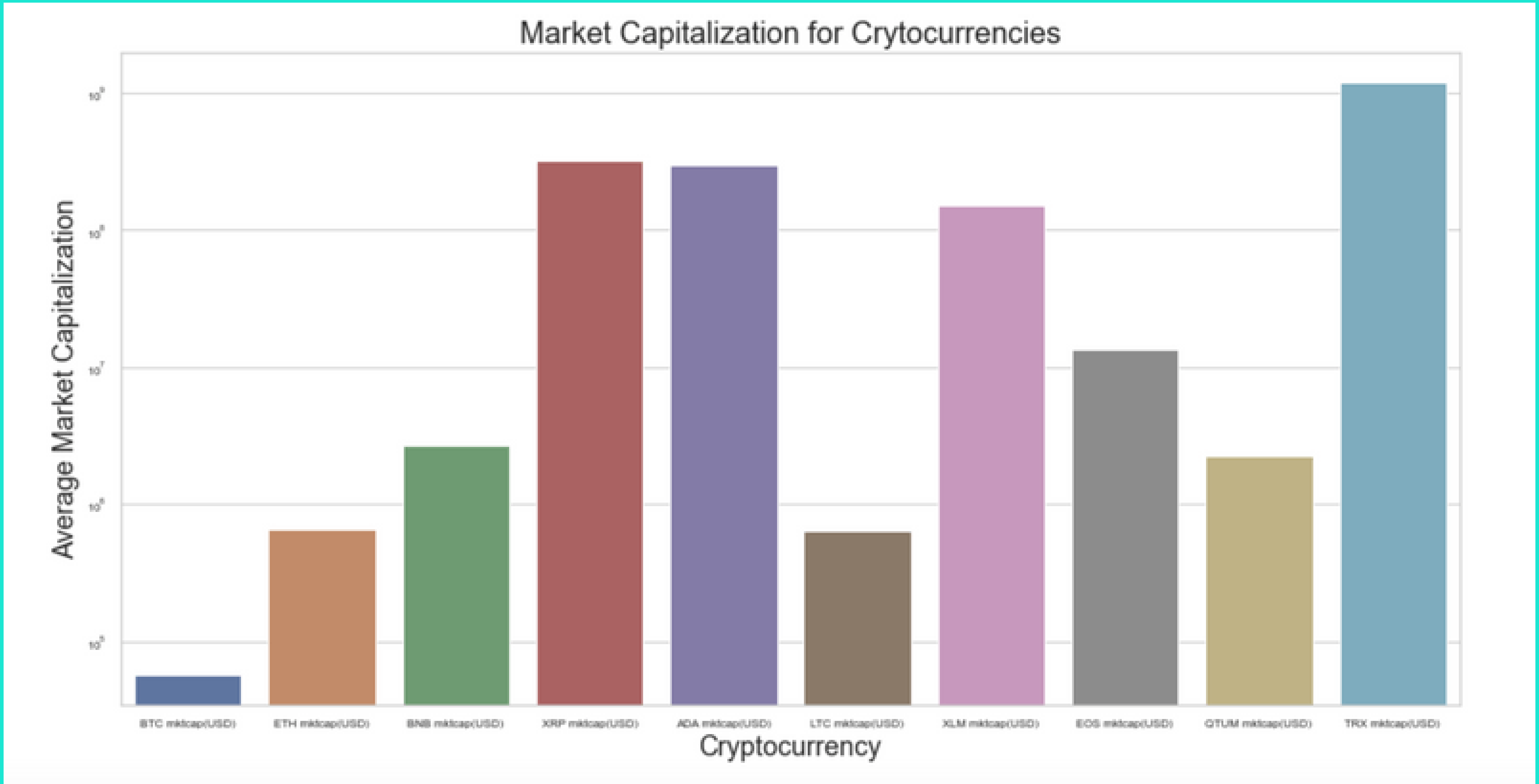
# Data Cleaning



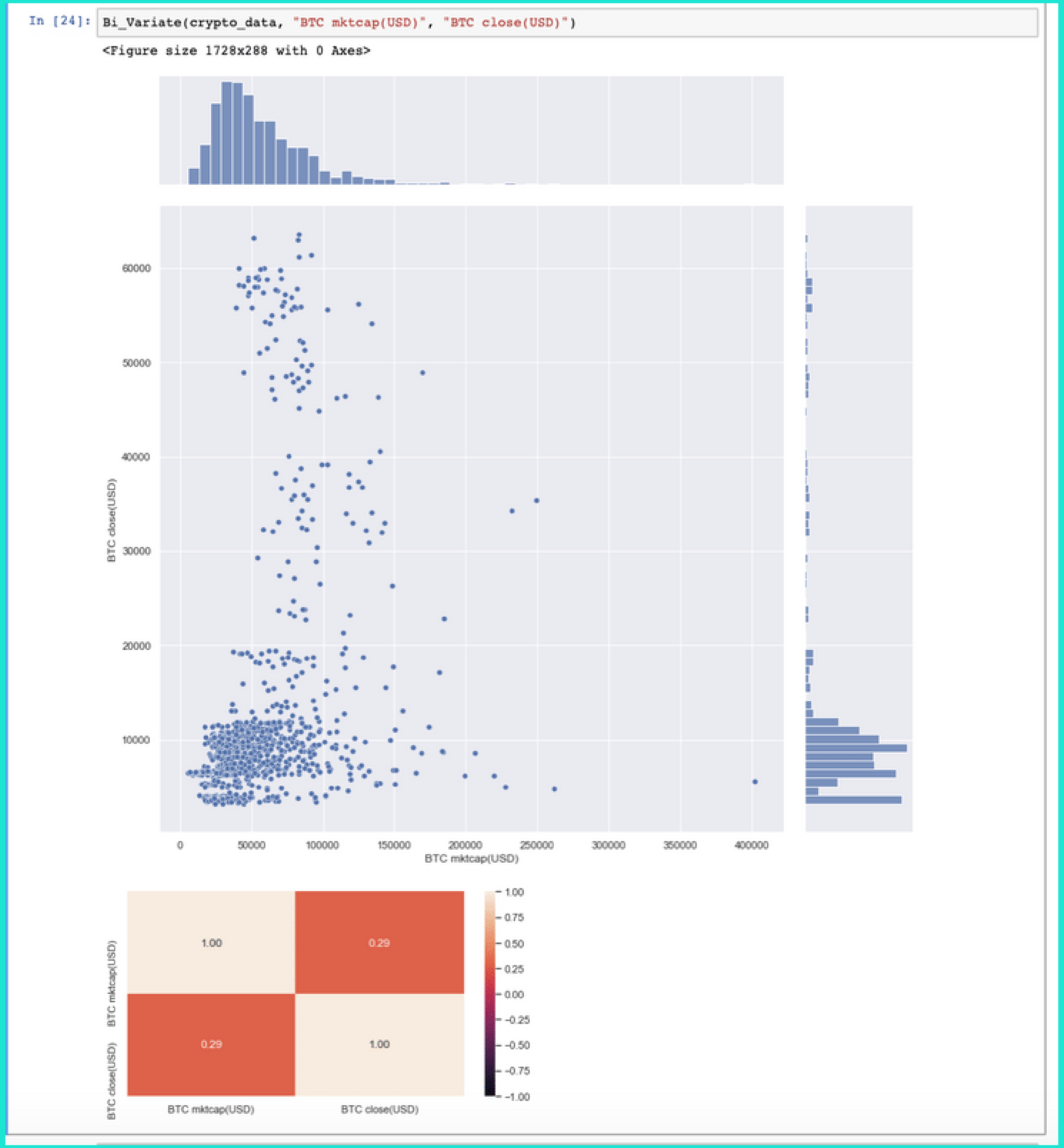
# EXPLORATORY DATA ANALYSIS








Market Capitalisation of Cryptocurrencies

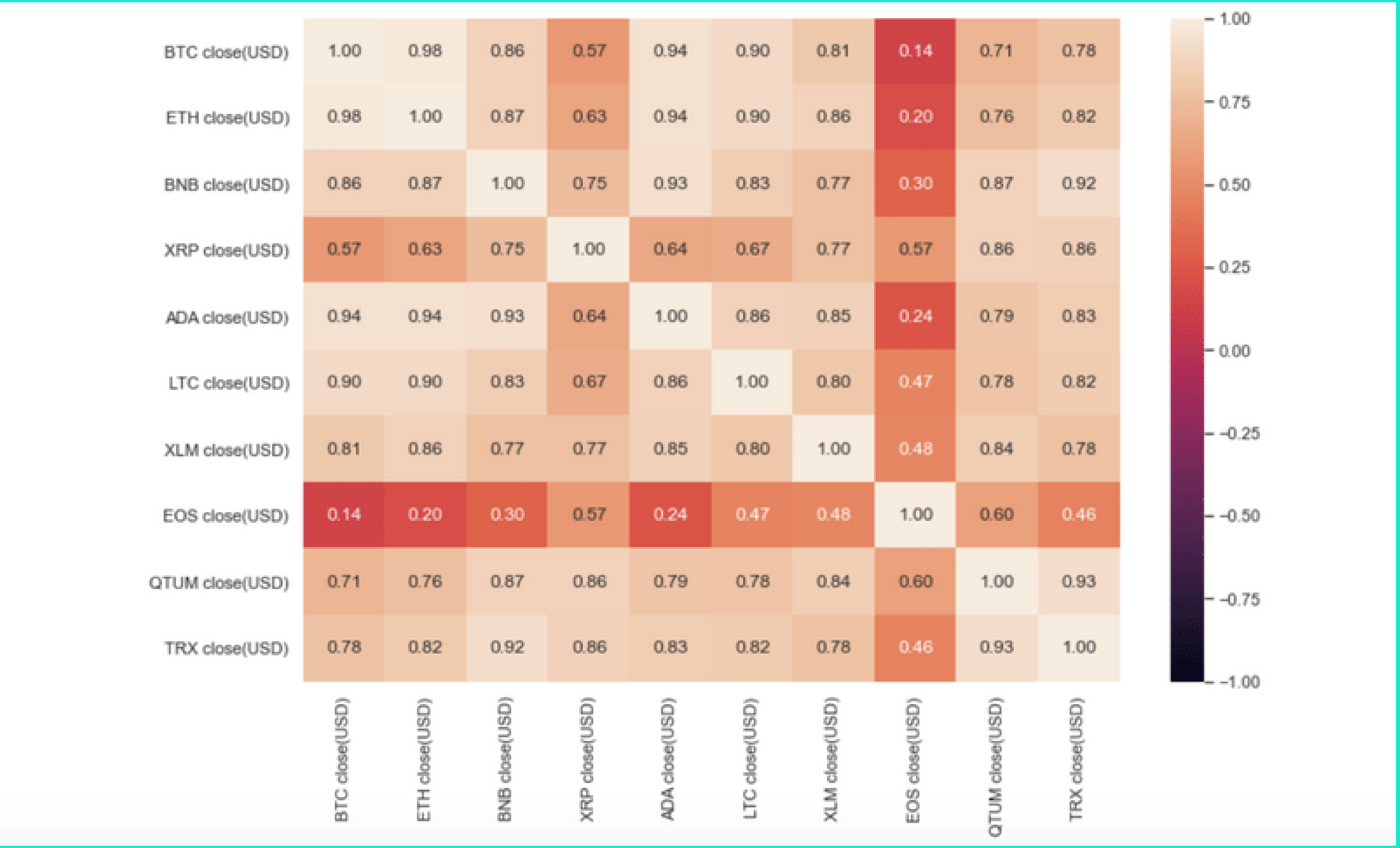



## BIVARIATE STATISTICS

Poor correlation between market cap and closing price.

## DATA VISUALIZATION

Pairplot shows a correlation between closing prices of different cryptocurrencies.




Correlation Heat map of Cryptocurrency closing prices

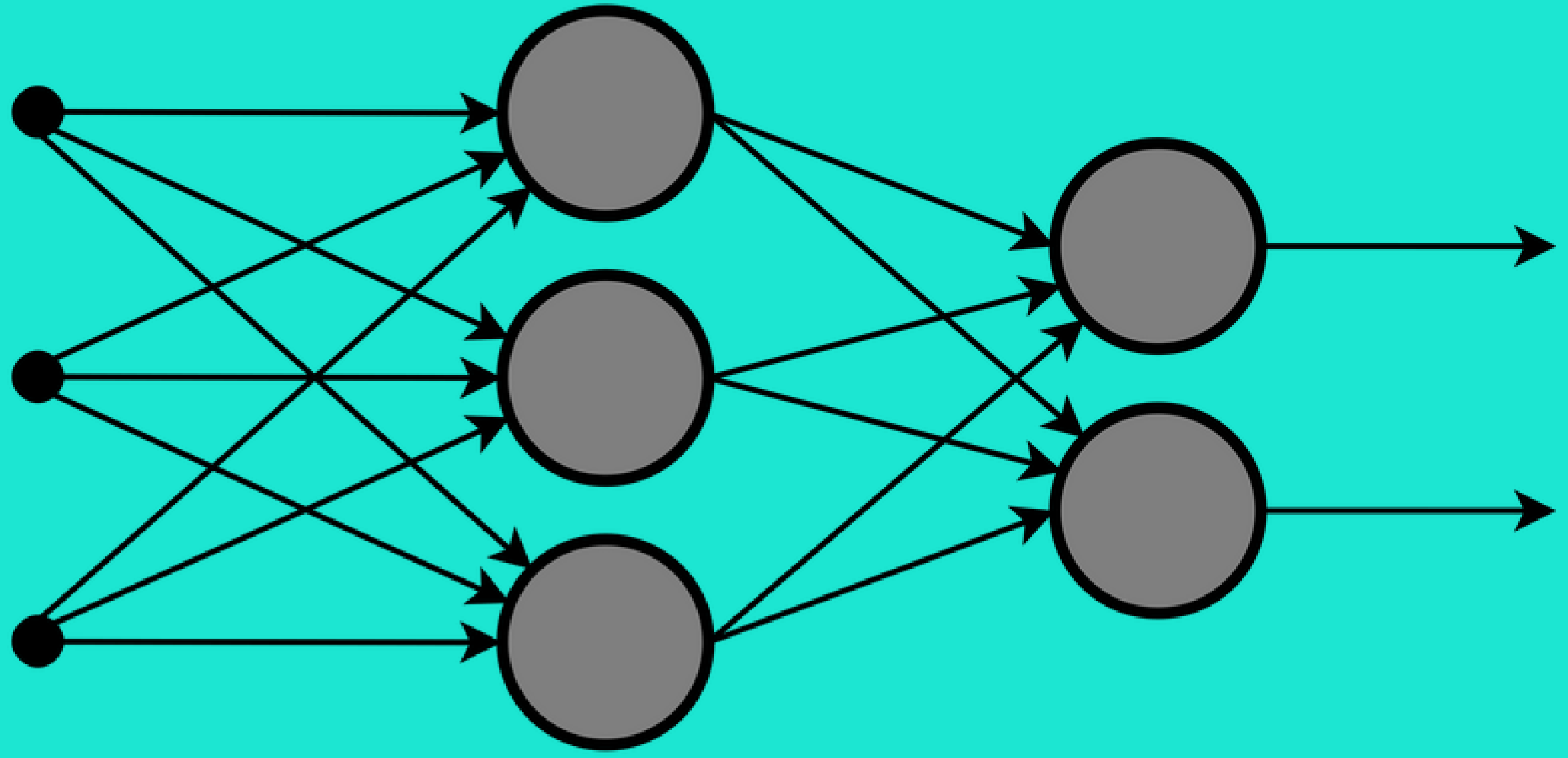
# DEEP LEARNING

Long Short-Term Memory Neural Networks For Time Series Forecasting

# What is an LSTM?

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning.

- Effective for time series forecasting
- Unlike standard feedforward neural networks, LSTM has feedback connections



## Simple LSTM

Single LSTM with hidden dense layer.

## Convolutional LSTM

LSTM with convolutional layers.

## Stacked LSTMs

Multiple LSTMs working concurrently

## Bi-directional LSTM

LSTM which utilizes, both, previous and future data to perform accurate forecasting

```
#Convolutional LSTM
#The shape of input data must be: [samples, timesteps, rows, columns, features]

trainX = trainX.reshape((trainX.shape[0], 1, 1, 1, seq_size))
testX = testX.reshape((testX.shape[0], 1, 1, 1, seq_size))

model = Sequential()
model.add(ConvLSTM2D(filters=64, kernel_size=(1,1), activation='relu', input_shape=(1, 1, 1, seq_size)))
model.add(Flatten())
model.add(Dense(32))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')
model.summary()
```

```
#Stacked LSTM with 1 hidden dense layer
# The shape of the input data must be: [samples, time steps, features]

#trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
#testX = np.reshape(testX, (testX.shape[0], 1, testX.shape[1]))

#model = Sequential()
#model.add(LSTM(50, activation='relu', return_sequences=True, input_shape=(None, seq_size)))
#model.add(LSTM(50, activation='relu'))
#model.add(Dense(32))
#model.add(Dense(1))
#model.compile(optimizer='adam', loss='mean_squared_error')

#model.summary()
```

```
#Bidirectional LSTM
# reshape input to be [samples, time steps, features]
#trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
#testX = np.reshape(testX, (testX.shape[0], 1, testX.shape[1]))
#
##For some sequence forecasting problems we may need LSTM to learn
## sequence in both forward and backward directions
#from keras.layers import Bidirectional
#model = Sequential()
#model.add(Bidirectional(LSTM(50, activation='relu'), input_shape=(None, seq_size)))
#model.add(Dense(1))
#model.compile(optimizer='adam', loss='mean_squared_error')
#model.summary()
```

```
#Single LSTM
#trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
#testX = np.reshape(testX, (testX.shape[0], 1, testX.shape[1]))

#print('Single LSTM with hidden Dense Layer')
#model = Sequential()
#model.add(LSTM(64, input_shape=(None, seq_size)))
#model.add(Dense(32))
#model.add(Dense(1))
#model.compile(loss='mean_squared_error', optimizer='adam')
#monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3, patience=20,
#                          verbose=1, mode='auto', restore_best_weights=True)
#model.summary()
```

# WHY LSTMS?

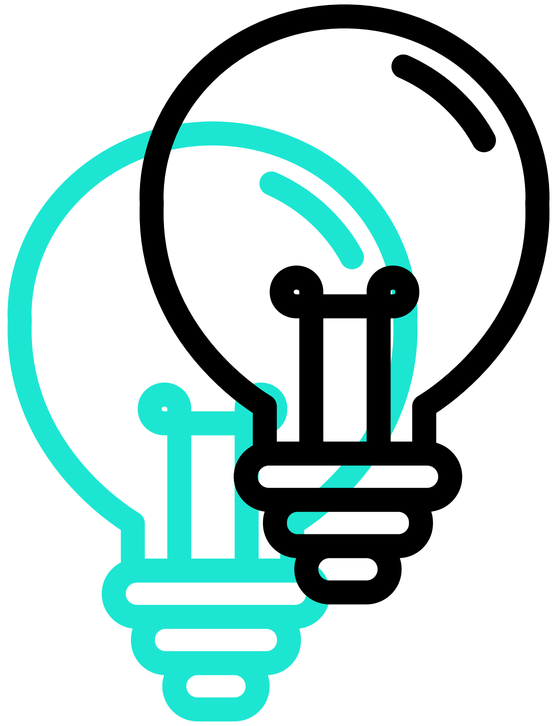
## Reason 1

Allows previous outputs to be used as inputs

## Reason 2

Accurate for Time Series forecasting

# DATA DRIVEN INSIGHTS



**Bi-directional LSTM:**

```
seq_size=10,epochs=50
Train Score: 354.02 RMSE
Test Score: 2421.81 RMSE

seq_size=15,epochs=50
Train Score: 373.15 RMSE
Test Score: 6189.28 RMSE

seq_size=7,epochs=50
Train Score: 358.74 RMSE
Test Score: 2735.18 RMSE

seq_size=10, epochs=100
Train Score: 323.52 RMSE
Test Score: 2654.81 RMSE
```

**Convolutional LSTM:**

```
seq_size=40, epochs=100
Train Score: 324.20 RMSE
Test Score: 3463.53 RMSE

seq_size=10,epochs=50
Train Score: 354.33 RMSE
Test Score: 3981.06 RMSE
```

*Best Case:*

```
seq_size=10, epochs=100
Train Score: 323.22 RMSE
Test Score: 1537.83 RMSE
```

**Stacked LSTMs:**

```
seq_size=10,epochs=100
Train Score: 353.35 RMSE
Test Score: 2899.92 RMSE

seq_size=7,epochs=100
Train Score: 318.81 RMSE
Test Score: 3310.89 RMSE

seq_size=15,epochs=100
Train Score: 392.87 RMSE
Test Score: 4119.80 RMSE

seq_size=10,epochs=50
Train Score: 339.17 RMSE
Test Score: 2191.39 RMSE

seq_size=10,epochs=35
Train Score: 443.97 RMSE
Test Score: 5269.84 RMSE
```

**Single LSTM with Hidden Dense Layer:**

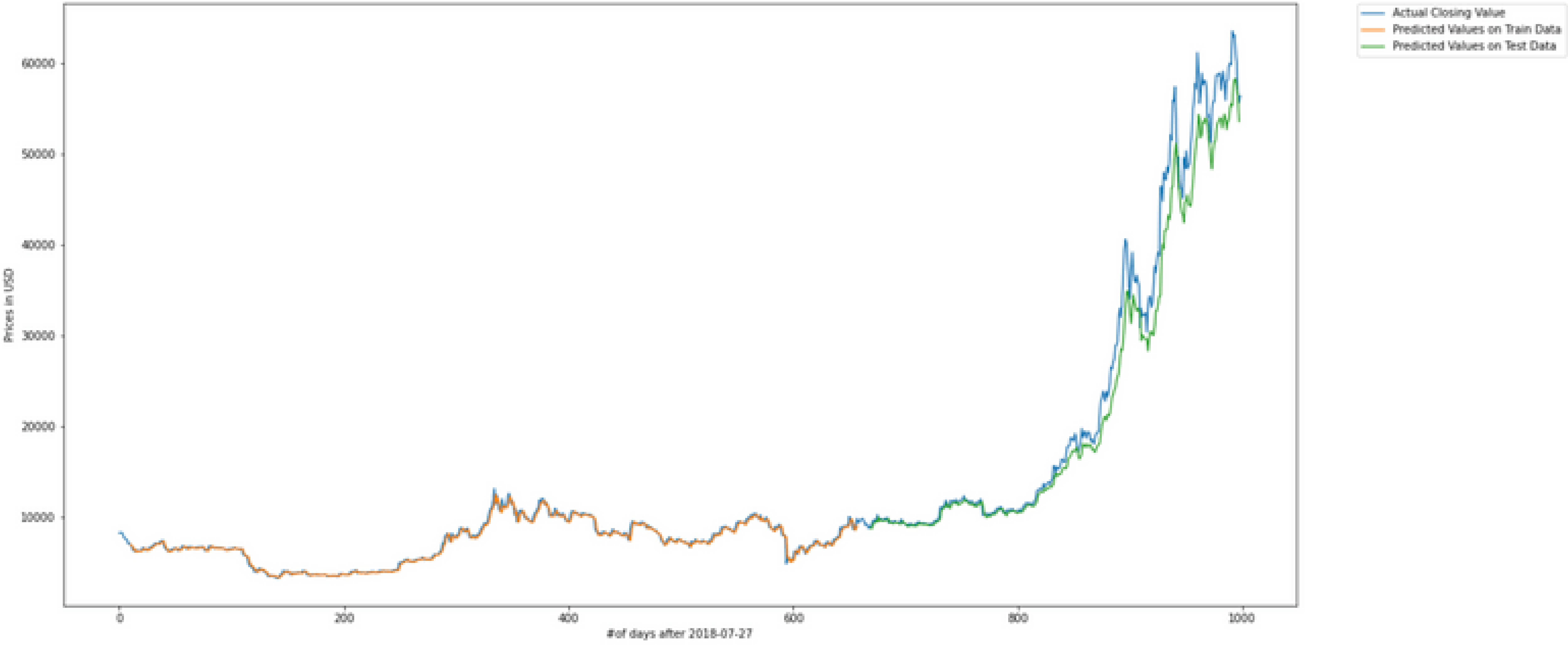
```
seq_size=10, epochs=100
Train Score: 334.80 RMSE
Test Score: 5394.96 RMSE

seq_size=10, epochs=50
Train Score: 347.99 RMSE
Test Score: 3222.51 RMSE
```

*Comparisons were performed on data for Bitcoin (BTC)

# Convolutional LSTM Model Summary:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_lst_m2d (ConvLSTM2D)    (None, 1, 1, 64)          21760
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 32)                2080
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
=================================================================
Total params: 23,873
Trainable params: 23,873
Non-trainable params: 0
_________________________________________________________________
```


Actual Closing Value
Predicted Values on Train Data
Predicted Values on Test Data
Prices in USD
60000
50000
40000
30000
20000
10000
0
200
400
600
800
1000
#of days after 2018-07-27

# RESULTS

- Compared RMSE values of various LSTM techniques
- Convolutional LSTM produces reliable results in most cases
- Sequence size between 7-20, #of Epochs 50-100
- Current closing values (uni-variate) are good indicators of future closing values for cryptocurrencies

# Beyond The Course

- Neural Networks
- Feature Normalization
- Long Short-Term Memory Neural Networks
- Time series Forecasting
- Overfitting

# Team Contributions

## Tejas

1. Data Analysis
2. Deep learning
3. Data Cleaning
4. Voice over

## Raghav

1. Brainstorming
2. Data cleaning
3. Exploratory Analysis

## Arushi

1. Brainstorming
2. Create data frames
3. Slide design & video presentation

# References

1.https://www.3blue1brown.com/
2. https://youtu.be/aircAruvnKk
3.Using a Keras Long Short-Term Memory (LSTM) Model to Predict Stock Prices – kdnuggets https://www.kdnuggets.com/2018/11/keras-long-short-term-memory-lstm-model-predict-stock-prices.html#:~:text=(%2018%3An45%20)-,Using%20a%20Keras%20Long%20Short%2DTerm%20Memory%20(LSTM),Model%20to%20Predict%20Stock%20Prices&text=LSTMs%20are%20very%20powerful%20in,in%20predicting%20its%20future%20price.
4.https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks