STEVENS
INSTITUTE OF TECHNOLOGY
1870

Cryptocurrency Price Prediction Using Machine Learning

Subject: Data Analytics & Machine Learning (MIS 637 - A)

Professor: Mahmoud Daneshmand

Presented By: Het Vyas, Ishan Bhatt, Jyoti Khare,
Sonal Kulshreshtha, Venkata Sandilya Bagavathula

Date: 12/14/2022

INDEX

ABOUT CRYPTOCURRENCY TRADING

- Cryptocurrency trading has become the new trend in today's time.
- As the world moves towards accepting and advancing in blockchain technologies, famous crypto coins like Bitcoin, Etherium, Dogecoin and many more have been introduced as alternatives to payments.
- With this, there have been several trading platforms which help users buy/sell these cryptocurrencies.
- Just like the stock market, the crypto market is becoming a more and more popular source of investment.
- However, due to limited knowledge of this new technology, people are not aware as to when to buy/sell a particular crypto asset.
- This leads to trades based on gut feeling over informed decisions based on historical data and current and future prospective developments.

PROBLEM STATEMENT

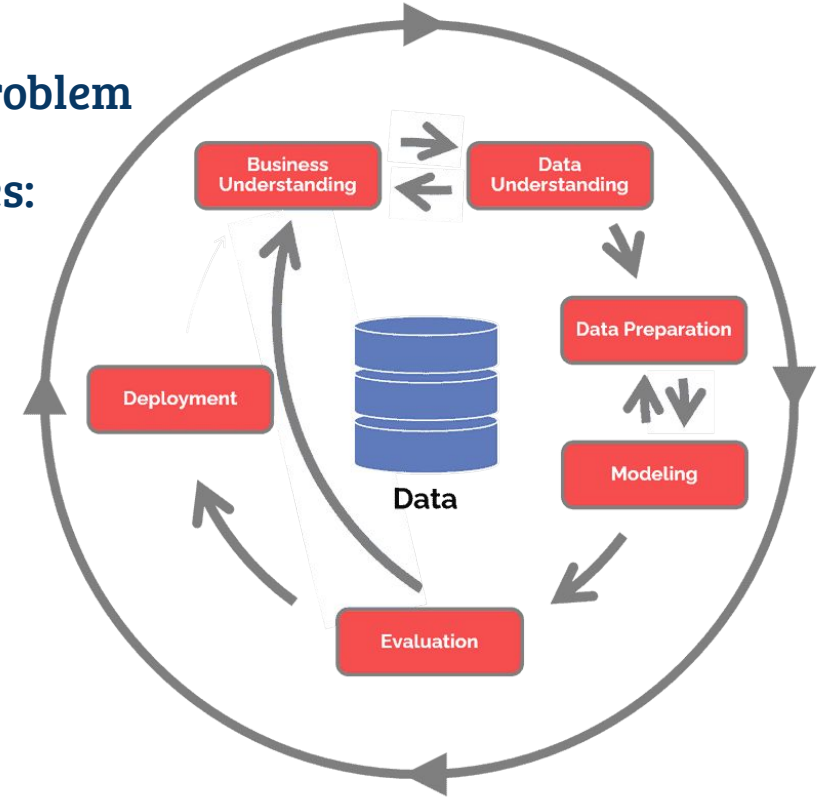
- To overcome the problem of making uninformed decisions while buying/selling cryptocurrency, we've designed a machine learning model which will give buy/sell trade calls based on the previous data recorded for the particular cryptocurrency.

- Also to get better result, we will be using a comprehensive feature set, including technical, blockchain based, sentiment based and asset based features.

CRISP-DM

We'll be using the CRISP-DM to solve the problem statement. CRISP-DM comprises of 6 phases:

1. Business Understanding Phase
2. Data Understanding Phase
3. Data Preparation Phase
4. Modeling
5. Evaluation
6. Deployment



BUSINESS UNDERSTANDING

- The reason behind choosing this topic for final semester project is the lack of awareness amongst the majority of crypto traders for Blockchain technology, its usefulness and how the crypto market operates.
- Also, the crypto market is open 24*7 365 days of the year. So, it's difficult to monitor the moment of price of any particular cryptocurrency throughout.
- This challenge often leads to customer missing out good buying/selling opportunities.
- This in turn leads to uninformed decision making. So, to make proper decisions after analyzing the historical data, recent government monetary policies, tweets from influencers, users can thus make a proper informed decision.

DATA UNDERSTANDING

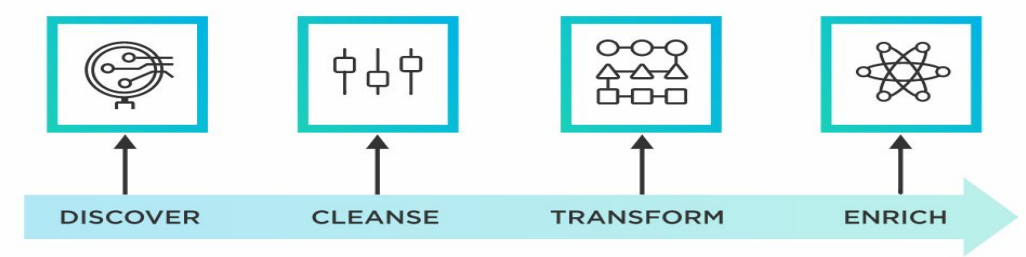
- As part of this step, we will get our historical data from Yahoo Finance's BTC-USD ($).
- The frequency of this data is on a daily basis. We load the same into the Pandas dataframe.
- We'll also be sorting our data based on time with daily intervals which helps us in performing time series analysis.
- Since this data is of type time series, we can also analyze various other properties like stationarity, autocorrelation and seasonality.
- Parameters that we considered for this algorithm.
    - Date: The provided date is considered for relevant data provided for that day.
    - The opening and closing suggests the value of 1 bitcoin at the start and end of the day respectively
    - The Highs and Lows suggest the value of bitcoins at their extreme fluctuations
- We obtained our dataset from : https://finance.yahoo.com/quote/BTC-USD/history?p=BTC-USD
- Similarly, we can obtain the dataset for any cryptocurrency of our choice

DATA UNDERSTANDING

SAMPLE DATA (with data anomalies)

Date	Open	High	Low	Close	Adj Close	Volume
12/17/2021	47653.73047	48004.89453	45618.21484	46202.14453	46202.14453	32902725329
"12-18-2021	46219.25391	47313.82813	45598.44141	46848.77734	46848.77734	26098292690
12/19/2021	46853.86719	48089.66406	46502.95313	46707.01563	46707.01563	25154053861
12/20/2021	46707.0625	47401.71875	45579.80859	46880.27734	46880.27734	30961902129
12/21/2021	46886.07813	49300.91797	46698.77344		48936.61328	27055803928
"12-21-2021	48937.09766	49544.79688	48450.94141	48628.51172	48628.51172	24447979559
12/23/2021	48626.34375	51332.33984	48065.83594	50784.53906	50784.53906	28223878108
12/24/2021	50806.05078	51814.02734	50514.49609	50822.19531	50822.19531	24367912228
12/25/2021	50854.91797	51176.59766	50236.70703	50429.85938	50429.85938	19030650914
12/26/2021	50428.69141	51196.37891	49623.10547	50809.51563	50809.51563	20964372926
12/27/2021	50802.60938	51956.32813	50499.46875	50640.41797	50640.41797	24324345758
12/28/2021	50679.85938		47414.21094	47588.85547	47588.85547	33430376883
12/29/2021	47623.87109	48119.74219	46201.49609	46444.71094	46444.71094	30049226299EE
12/30/2021	46490.60547	47879.96484	46060.3125	47178.125	47178.125	26686491018

DATA PREPARATION



- Data preparation is the process of gathering, combining, structuring and organizing data.
- Data is commonly created with missing values, inaccuracies or other errors, and separate data sets often have different formats that need to be reconciled when they're combined.
- The components of data preparation include data preprocessing, profiling, cleansing, validation, transformation and enrichment.
- Correcting data errors, validating data quality and consolidating data sets are big parts of data preparation phase.

DATA PREPARATION

- As seen in the initial dataset, there were many data anomalies : null values, incorrect data values and incorrect format. (Marked in red)
- As a part of data preparation we can tackle them as follows :
    - Incorrect date format issue : To avoid such anomalies, a consistent data format should be followed. Since most of dates are in MM/DD/YYYY format, we will update the incorrect data as below.

12/18/2021	46219.25391	47313.82813	45598.44141	46848.77734	46848.77734	26098292690

    - Incorrect data values arises if the cell values do not follow the column(property) data type. Like one of the volume has alphanumeric value instead of numeric value. It can be corrected by strictly following the allowed data type values like below:

12/29/2021	47623.87109	48119.74219	46201.49609	46444.71094	46444.71094	30049226299

DATA PREPARATION

- The other most common type of issues are with null values. Some common values to deal with null values is to replace it with mean/median of the dataset or to predict the values using the other values in dataset. For instance the the below null values were predicted as below:

12/21/2021	46886.07813	49300.91797	46698.77344	48936.61328	48936.61328	27055803928

- All these null values or NA anomalies have been taken care by the code as below:

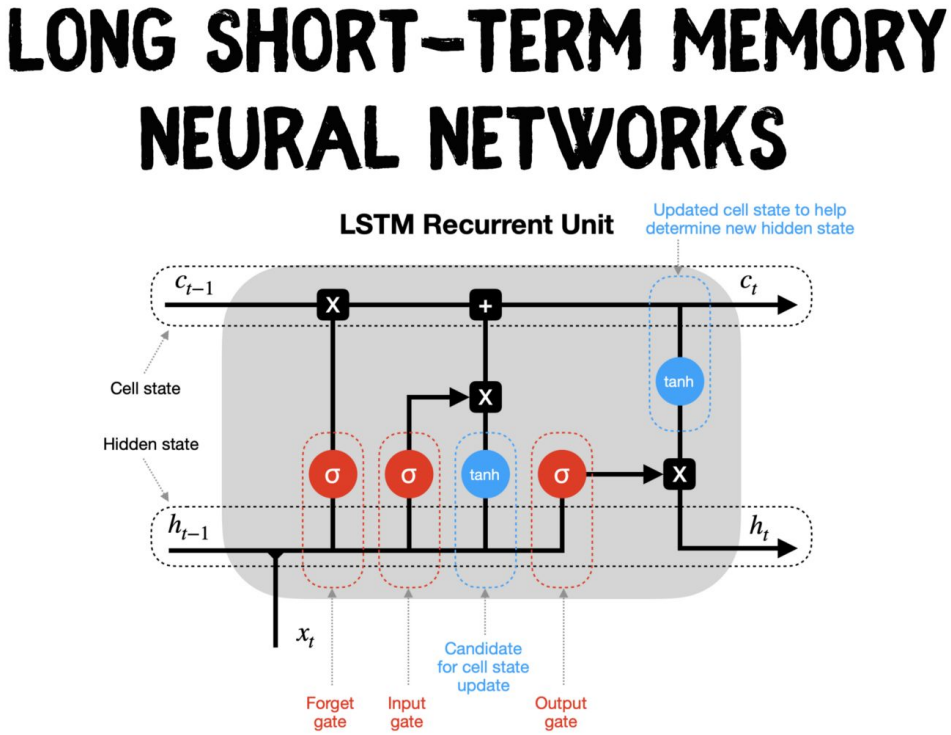
```
[8] # Check for NA values
    print('NA values:',maindf.isnull().values.any())

    # Check for Null values
    print('Null Values:',maindf.isnull().values.sum())

    print('Shape of Dataset:', maindf.shape)
```

```
NA values: False
Null Values: 0
Shape of Dataset: (365, 7)
```

DATA MODELING USING LSTM

- Here, we are using the Long Short-Term Memory which is an artificial recurrent neural network architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can process not only single data points, but also entire sequences of data.

- LSTM networks are a type of recurrent neural networks capable of learning order dependence in sequence prediction problems. This is a behavior required in complex problem domains like machine translation, speech recognition, and more. LSTMs are a complex area of deep learning.



DATA MODELING

- In the model building part, we have prepared the data for training and testing.

Displaying Bitcoin Price from 2014 - 2022

In [9]:
```
# Getting all the closing prices from last one year
closedf = maindf[['Date','Close']]
print("Shape of close dataframe:", closedf.shape)

fig = px.line(closedf, x=closedf.Date, y=closedf.Close,labels={'date':'Date','close':'Close Stock'})
fig.update_traces(marker_line_width=2, opacity=0.8, marker_line_color='red')
fig.update_layout(title_text='Bitcoin close price from 2014-2022', plot_bgcolor='white', font_size=15, font_color='b
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

```
Shape of close dataframe: (365, 2)
```

In [10]:
```
# Getting all the closing price of bitcoin from last one year

closedf = closedf[closedf['Date'] > '2021-07-29']
close_stock = closedf.copy()
print("Total data for prediction: ",closedf.shape[0])
```

```
Total data for prediction:  365
```

In [ ]:
```
closedf
```

Out[37]:

	Date	Close
2508	2021-07-30	42235.546875
2509	2021-07-31	41626.195313
2510	2021-08-01	39974.894531
2511	2021-08-02	39201.945313
2512	2021-08-03	38152.980469

We considered one year's data for training.

Displaying Bitcoin Price of Last 1 Year

In [11]:
```
ig = px.line(closedf, x=closedf.Date, y=closedf.Close,labels={'date':'Date','close':'Close Stock'})
ig.update_traces(marker_line_width=2, opacity=0.8, marker_line_color='red')
ig.update_layout(title_text='Bitcoin close price from last 1 Year', plot_bgcolor='white', font_size=15, font_color='
ig.update_xaxes(showgrid=False)
ig.update_yaxes(showgrid=False)
ig.show()
```

Normalising Data

In [12]:
```
del closedf['Date']
scaler = MinMaxScaler(feature_range=(0,1))
closedf = scaler.fit_transform(np.array(closedf).reshape(-1,1))
print(closedf.shape)
```

```
(365, 1)
```

Slicing Data into Training and Test Data

- We keep the training Data as 60%
- We keep the test Data as 40%

In [13]:
```
training_size = int(len(closedf)*0.60)
test_size = len(closedf) - training_size
train_data, test_data = closedf[0:training_size,:], closedf[training_size:len(closedf),:1]
print("train_data: ", train_data.shape)
print("test_data: ", test_data.shape)
```

```
train_data:  (219, 1)
test_data:  (146, 1)
```

Transform the closed price based on time series analysis forecasting

In [14]:
```
# Converting an array of values into a dataset matrix

def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]   ###i=0, 0,1,2,3-----99   100
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return np.array(dataX), np.array(dataY)
```

In [15]:
```
time_step = 15
X_train, y_train = create_dataset(train_data, time_step)
X_test, y_test = create_dataset(test_data, time_step)

print("X_train: ", X_train.shape)
print("y_train: ", y_train.shape)
print("X_test: ", X_test.shape)
print("y_test", y_test.shape)
```

```
X_train:  (203, 15)
y_train:  (203,)
X_test:  (130, 15)
y_test (130,)
```

In [16]:
```
# reshape input to be of the format [samples, time steps, features] which is required for LSTM

X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)

print("X_train: ", X_train.shape)
print("X_test: ", X_test.shape)
```

```
X_train:  (203, 15, 1)
X_test:  (130, 15, 1)
```

Model Building

In [17]:
```
model=Sequential()

model.add(LSTM(10,input_shape=(None,1),activation="relu"))

model.add(Dense(1))

model.compile(loss="mean_squared_error",optimizer="adam")
```

In [18]:
```
history = model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=200,batch_size=32,verbose=1)
```

```
Epoch 1/200
7/7 [==============================] - 2s 58ms/step - loss: 0.5197 - val_loss: 0.0166
Epoch 2/200
7/7 [==============================] - 0s 12ms/step - loss: 0.4794 - val_loss: 0.0134
Epoch 3/200
7/7 [==============================] - 0s 17ms/step - loss: 0.4440 - val_loss: 0.0113
Epoch 4/200
7/7 [==============================] - 0s 15ms/step - loss: 0.4144 - val_loss: 0.0096
Epoch 5/200
7/7 [==============================] - 0s 17ms/step - loss: 0.3888 - val_loss: 0.0083
Epoch 6/200
7/7 [==============================] - 0s 15ms/step - loss: 0.3694 - val_loss: 0.0069
Epoch 7/200
7/7 [==============================] - 0s 17ms/step - loss: 0.3534 - val_loss: 0.0057
Epoch 8/200
7/7 [==============================] - 0s 17ms/step - loss: 0.3379 - val_loss: 0.0045
Epoch 9/200
7/7 [==============================] - 0s 17ms/step - loss: 0.3227 - val_loss: 0.0038
Epoch 10/200
```

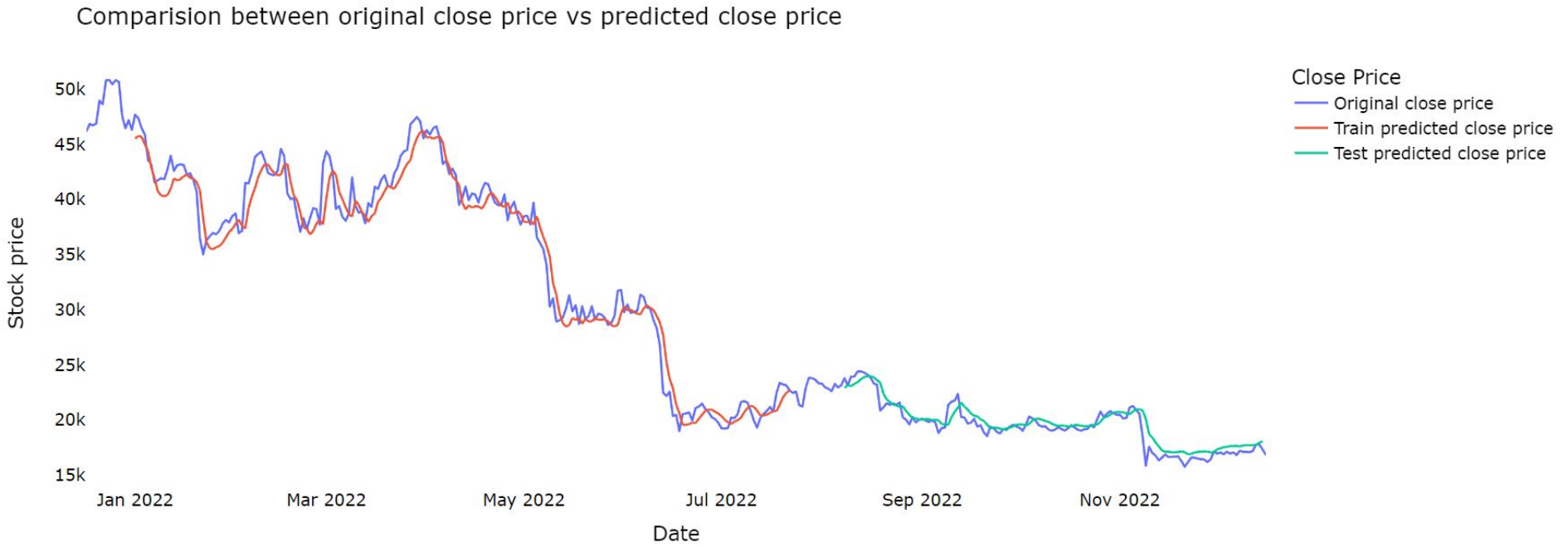
EVALUATION

Evaluation metrics are used to measure the quality of the statistical or machine learning model. There are various metrics on which the models can be assessed for accuracy and better understand the performance of the model

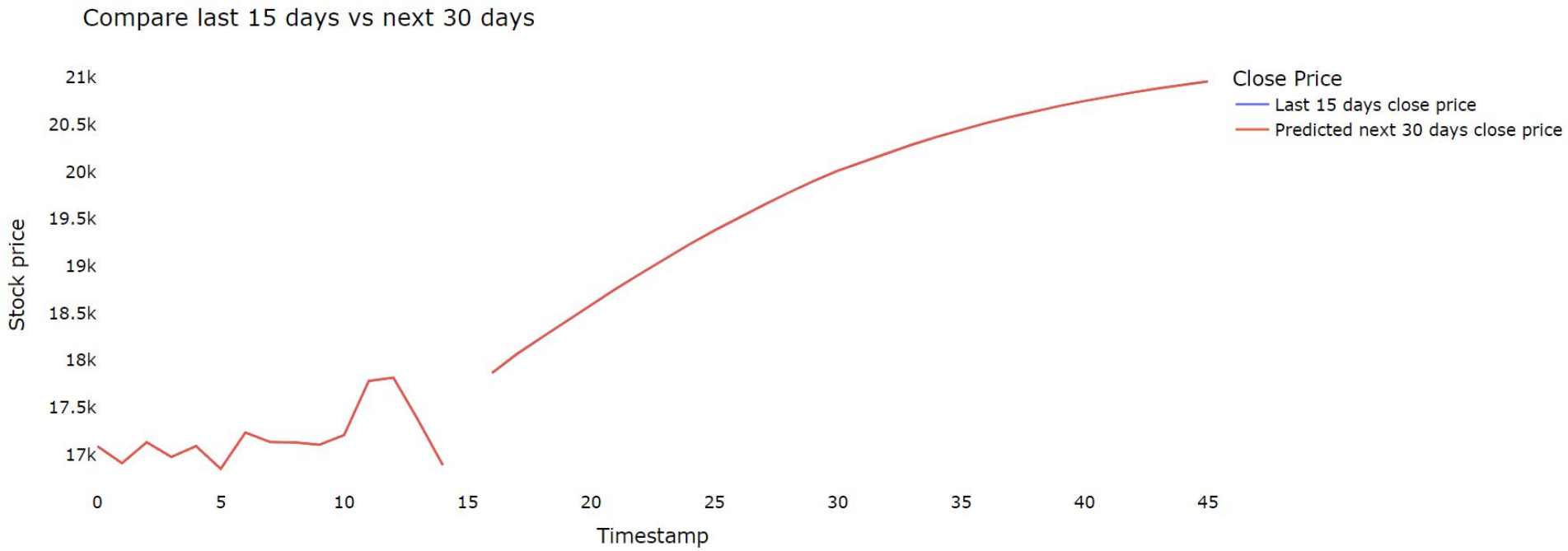
Mean Squared Error(MSE) is perhaps the most popular metric used for regression problems. It essentially finds the average squared error between the predicted and actual values. The MSE is a measure of the quality of an estimator — it is always non-negative, and values closer to zero are better.

Explained variance compares the variance within the expected outcomes, and compares that to the variance in the error of our model. This metric essentially represents the amount of variation in the original dataset that our model is able to explain.

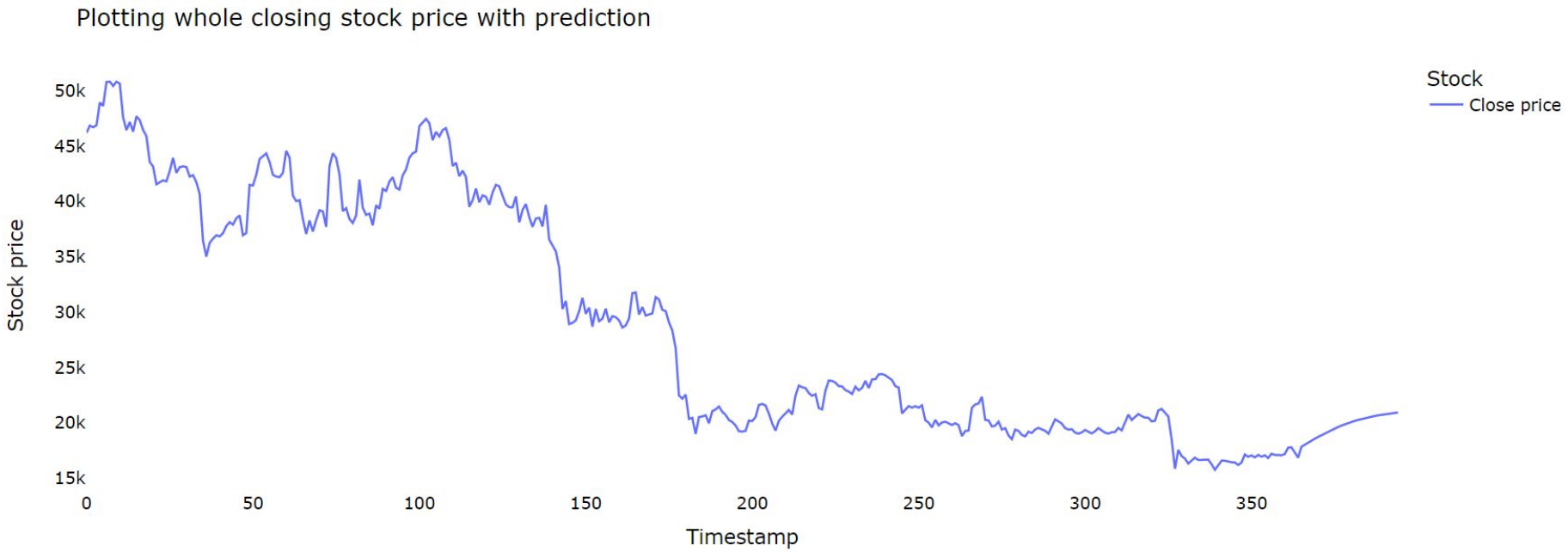
Furthermore, we analyzed the original and the predicted closing prices and plotted the graph for the same. You can see that the values are pretty closely accurate signifying that the model did perform well on the dataset.



In this graph we have predicted the next stock market values for the next 30 days and compared it with previous 15 days of the predicted value



At the end plotting the plotting the combined graphs of the predicted closing values to get overall representation of the stock market



DEPLOYMENT

- Machine learning model deployment is the process of placing a finished machine learning model into a live environment where it can be used for its intended purpose. ML model can be integrated with API or deployed in other environments where it can be made available to the end users.
- This model for Cryptocurrency prediction can be deployed across various cryptocurrency investing apps and can be used to better predict the values based on live data being received.
- As scope of this project, we have not deployed the model to any server for now.

SCOPE OF FUTURE WORK

- As part of this project, we can further enhance the model by including the news from different global economies, tweets by influencers and so on.
- Not only that, the crypto market is also highly affected by the sentiments of the people which is difficult to account for.
- All these factors can be taken into consideration to further enhance the efficiency of predictions of our model.

CONCLUSION

- As part of the LSTM Bidirectional model, we've partitioned the dataset as 60% training data and 40% test data.
- On successful model building and training, we were able to achieve an accuracy of 85.47% on our test data.
- Also, LSTM model is ideally the best fit for prediction of time series data sets.

REFERENCES

- Book ‘Discovering Knowledge in Data: An Introduction to Data Mining’ by Daniel T. Larose.
- Data Source: https://finance.yahoo.com/crypto/

STEVENS
INSTITUTE OF TECHNOLOGY
1870

THANK YOU

Stevens Institute of Technology
1 Castle Point Terrace, Hoboken, NJ 07030