BHARATI VIDYAPEETH'S
COLLEGE OF ENGINEERING

PROJECT REPORT

TOPIC: "Bitcoin Price Analysis Prediction"

Submitted by : Group 8

Vishal Sharma (10011502720)
Vishwas Mishra (10111502720)
Shivam (08311502720)
Vinayak (09811502720)

Declaration of Candidates

We here-by declare that the project with title “Bitcoin Price Analysis Prediction”, has been completed by us in partial fulfillment of “BACHELOR OF TECHNOLOGY” degree examination as prescribed by Bharati Vidyapeeth's College of Engineering, New Delhi and this has not submitted for any other examination and does not form the part of any other course undertaken by us.

Date

31 August, 2022

Signed By

Vishal Sharma
Vishwas Mishra
Shivam
Vinayak

ACKNOWLEDGEMENT

With immense pride and sense of gratitude, We take this golden opportunity to express our sincere regards to Prof. Dharmender Saini (Principal of Bharati Vidyapeeth's College of Engineering, New Delhi).

We are extremely thankful to our project guides for the guideline throughout the project. We tender our sincere regards to the coordinator, for giving us outstanding guidance, enthusiastic suggestion and invaluable encouragement which helped us in the completion of our project.

We want to thank our mentor Mr. Harshit Gaur, for his constant support and guidance. We would also like to thank our instructors Mrs. Neetu Singh, Mrs. Deepika Vashney and Dr. Priyanka Behera for providing their mentorship throughout the course. We came to know and learn about so many new things that we are really thankful for. We will fail in our duty if we don't thank the non-teaching staff of the college for their cooperation. We would like to thank all those who helped me in making this project complete and successful.

CERTIFICATE

This is to certify that Group 8 has submitted the project report titled "Bitcoin Price Analysis Prediction", towards partial fulfillment of BACHELOR OF TECHNOLOGY degree examination. This has not been submitted for any other examination and does not form part of any other course undergone as prescribed by Bharati Vidyapeeth's College of Engineering. It is further certified that we have ingeniously completed our project.

TABLE OF CONTENTS

Preface

Cryptocurrencies are a digital way of money in which all transactions are held electronically. It is a virtual currency which doesn't exist in the form of hard notes physically. Here, we are emphasizing the difference of fiat currency which is decentralized that without any third-party intervention all virtual currency users can get the services. However, getting services of these cryptocurrencies impacts on international relations and trade, due to its high price volatility. There are several virtual currencies such as bitcoin, ripple, ethereum, ethereum classic, litecoin, etc.

In our study, we focused on a popular cryptocurrency, i.e., bitcoin. From many types of virtual currencies, bitcoin has a great acceptance by different bodies such as investors, researchers, traders, and policy-makers.

To the best of our knowledge, our target is to implement the efficient deep learning-based prediction models, specifically long short-term memory (LSTM) to handle the price volatility of bitcoin and to obtain high accuracy. Our study involves

LIST OF FIGURES

INTRODUCTION

Bitcoins are a form of decentralized cryptocurrency which are used to store value. One can store it, exchange it and make payments with it. Bitcoin is the pioneer in the crypto-space since it was developed and put to use in late 2008.

In the current report, we are trying to show an interesting view of economics questions surrounding prediction of crypto assets, specifically bitcoin, and the nature of crypto currency following trends.

Bitcoin price prediction has been an active area of research for a long time. Bitcoin, as a pioneer within the blockchain monetary renaissance, plays an overwhelming part in an entire cryptocurrency market capitalization environment. Hence, it is the incredible interest of machine learning and data mining community to be able to:

(I) predict Bitcoin price changes

(II) grant experiences to get at what drives the Bitcoin instability and way better assess related dangers in the crypto currency domain.

Many researchers worked on machine learning algorithms and sentiment analysis from social media to find out the bitcoin market price prediction.

BITCOIN

Ticker : BTC, XBT
Price today : ₹16,14,223 (floating)
Block Reward : 6.25 BTC (current)
Supply Limit : 21,000,000 BTC

Initial concept

Imagine you're at a table with some friends. You come up with a set of rules to create a type of money that you can only use between each other. You decide that math is really hard, but doable, given enough time and energy. You decide that there's a certain math problem that you all like to do, and you know you can find ways to do it faster and faster as time goes on. You set how many times you want to try to solve a really hard math problem as the basis for work.

Mining

Every so often, when someone in the group solves a really hard math problem, that person tells the others of their triumph. The others can quickly verify that the solver is truthful, so after checking their work, they all write down that that person has gained some money. When someone lies about solving a problem, everyone knows and ignores their claim. This continues until we've solved the math problem the number of times we've set (this will take more than 100 years). That briefly describes "mining".

Transactions

A transaction is when someone sends money to someone else.
When someone in the group spends money, they tell everyone else in the group to whom they're sending their money. However, it's done in a way that makes it really hard to actually tell who is sending and who is receiving. It's not 100% impossible to tell, but it would take someone a lot of time to figure it out.
When someone sends money, everyone can check to ensure that that person actually has enough money to send what they claim to have sent. This is especially important to the receiver. If enough people say that the sender doesn't have enough money to send, the sender's transaction is ignored. He'll be able to try again in a little bit (and hopefully he'll be more honest or careful this time!).

The really cool thing is that everyone who wants to send or receive money knows how much everyone else is sending or receiving, even though they can't really identify the person behind the transaction. Plus, a person can look at the history of every transaction since the beginning of this fun system. This is called the public ledger.
Whenever someone sends money, they send a little bit extra along with their amount. This is called the transaction fee, and it's kinda like a tip. It's given as a reward to the person who solves the really hard math problem as an extra bonus! Over time, the reward money the solver earns will go away. Eventually, this transaction fee reward will be larger than the reward for solving ever was! So, this transaction fee gives people a reason to keep solving math problems forever.

Blockchain

Whenever someone sends money (creates a transaction) and their send is truthful, there's a hash created. A hash is like a secret word that you can only remember if you combine a few other words you always know. By combining some of the information about a recently solved math problem and some information about the current transaction, you can ensure that no one can fake our transaction again - not even yourself. Each transaction is contained within some notes about that recently solved math problem - these notes are called "blocks".
When we hash the blocks and the transactions together, it creates a chain with links that are impossible to replace without going back and doing all of the math problems again and convincing all of the other people that your new, replacement work is the real work. This is virtually impossible, so transactions and blocks are not able to be faked or undone.

Getting Bitcoin

When someone new wants to join in so that they can use the money too, they request a copy of all of the notes -- the blockchain -- and thus have their own copy of the public ledger. They can look at everyone's transactions and add their own to the mix, as long as they adhere to the rules we've all agreed on.

If they want to get some money to spend for themselves, they can try to solve math problems (mining), exchange some valuable item to someone already in the group in exchange for some money (exchanges or stores), or do something to help someone in the group so that they can earn the money, like mowing their lawn or cleaning their room.
It used to be really easy to do math problems, but it's getting harder and harder daily. Soon, the amount given to someone when they solve a problem will halve! They'll get only 25 coins instead of 50 coins. So, people are looking to buy into the group's money or do something in exchange for money they can spend within the group.

Proposed Methodology

The proposed methodology is shown in the following diagram

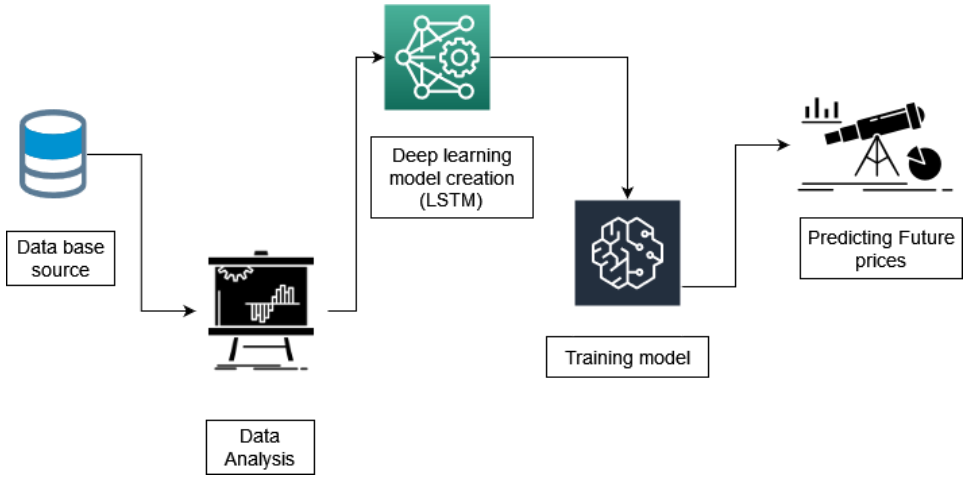

Fig 1.1: Proposed methodology

The methodology we used can be explained as following points: 1. Analyzing the data of the bitcoin prices by using graphs and plots. This helped us in observing the presence of the trends in the data present, if any. We observed a lot of randomness in the data.

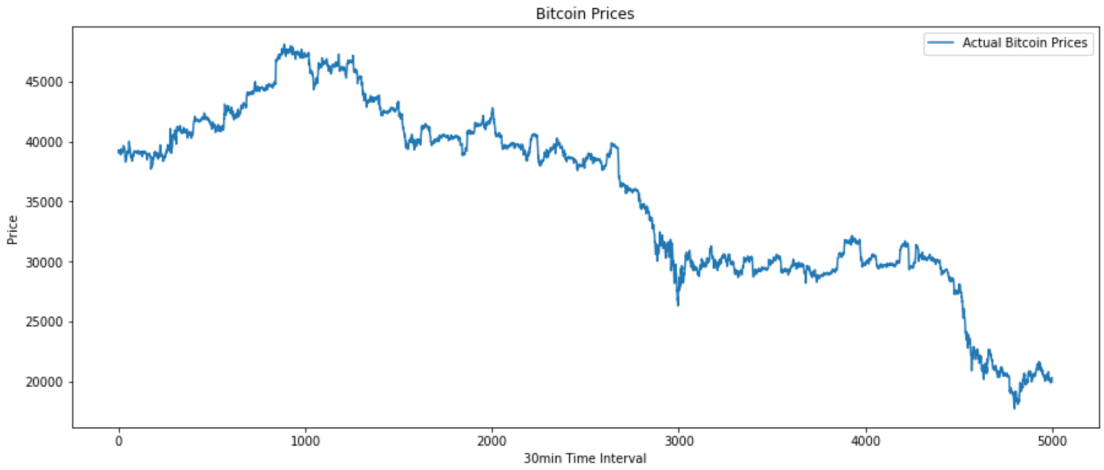

Fig 1.2: Analysis of Trends in Bitcoin Prices

2.Creation of the deep learning model: We observed that the model could be made on the basis that the previous 30 or so values contributed to the next value of the price, so we used the recurrent neural network LSTM. LSTM stands for Long Short Term Memory and this model was selected as the previous values affected the next values for short duration.

3.Then we trained the model on basis of our data by splitting it for testing and training.

4. On basis of our trained model we could predict the results of the future , unknown values if we knew previous 24 data values.

Principles

RNN

RNN is a deep neural network characterized as a recurrent connection between the input and output of its neurons or layers and capable of learning sequences designed to capture temporal contextual information along time series data. They have recently gained popularity in deep learning due to their ability to overcome the limitation of existing neural network architecture where it comes to learn over long sequences.
Two common RNN networks are LSTM and GRU and presented in the subsequent sections.

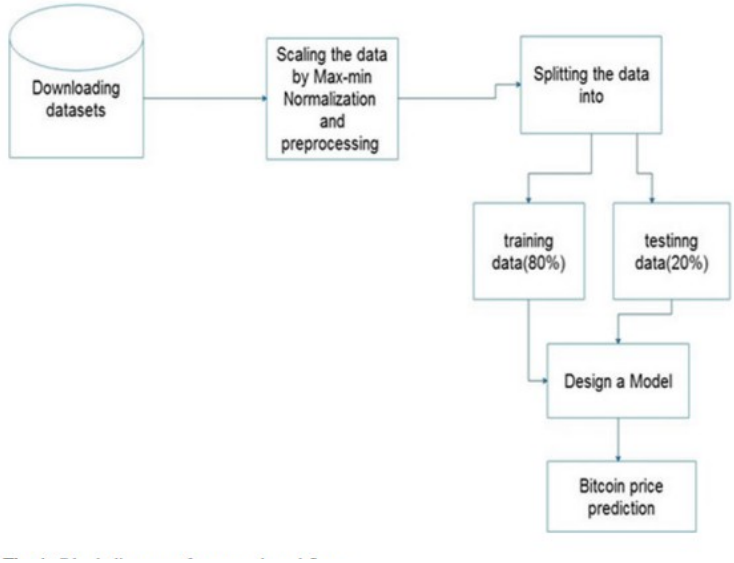

Fig 1.3: Standard deep learning price prediction problem workflow

LSTM

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. All recurrent neural networks that have the form of the module will have a very simple structure, such as a single *tan h* layer.
The deep learning LSTM neural networks overcome the problems with RNN related to vanishing gradients, by replacing nodes in the RNN with memory cells and gating mechanism. In this regard, it is an attractive deep learning neural architecture mostly on the account of its efficiency in memorizing long- and short-term temporal information simultaneously.

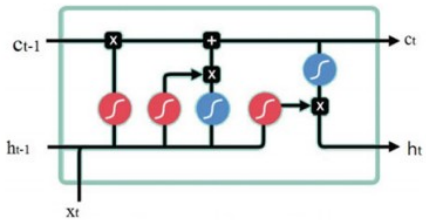

Fig 1.4: LSTM

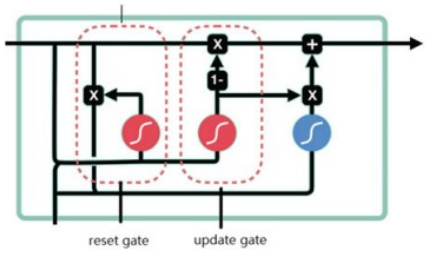

Fig 1.5: Modified LSTM

TensorFlow

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.
TensorFlow was developed by the Google Brain team for internal Google use in research and production. The initial version was released under the Apache License 2.0 in 2015. Google released the updated version of TensorFlow, named TensorFlow 2.0, in September 2019.
TensorFlow can be used in a wide variety of programming languages, most notably Python, as well as Javascript, C++, and Java. This flexibility lends itself to a range of applications in many different sectors.

Requests

Requests is a HTTP library for the Python programming language. The goal of the project is to make HTTP requests simpler and more human-friendly. The current version is 2.28.0. Requests is released under the Apache License 2.0.
Requests is one of the most popular Python libraries that is not included with Python. It has been proposed that Requests be distributed with Python by default.

Keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model.

Working

```
#importing all libraries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import requests
import tensorflow as tf
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.layers import Dense, Dropout, LSTM
from tensorflow.keras.models import Sequential

#obtaining market data from twelvedata

api_key = 'ae33d65c666843ec9ea83892f64671b4'
symbol = 'BTC/USD'
interval = '30min'
order = 'asc'
start_date = '2022-03-10 00:00:00'
end_date = '2022-06-23 00:00:00'

api_url=f'https://api.twelvedata.com/time_series?apikey={api_key}&interval=
{interval}&symbol={symbol}&order={order}&start_date=
{start_date}&end_date={end_date}'
tf.random.set_seed(7)
data = requests.get(api_url).json()
data_final = pd.DataFrame(data['values'])
data_final
```

	datetime	open	high	low	close
0	2022-03-10 07:30:00	39315.60156	39419.69141	39215.67188	39236.58984
1	2022-03-10 08:00:00	39248.55078	39248.55078	38868.89844	39053.23047
2	2022-03-10 08:30:00	39058.75000	39269.39844	39006.03125	39269.39844
3	2022-03-10 09:00:00	39235.32812	39297.55078	39104.00000	39169.82812
4	2022-03-10 09:30:00	39151.76172	39316.87109	39062.44141	39185.78125
...	...	...	...	...	...
4995	2022-06-22 22:00:00	19969.13086	20032.71094	19899.50977	19944.00977
4996	2022-06-22 22:30:00	19946.08008	20143.83008	19943.00000	20104.47070
4997	2022-06-22 23:00:00	20103.69922	20119.11914	20060.66016	20069.09961
4998	2022-06-22 23:30:00	20073.66016	20097.60938	19948.75977	19953.93945
4999	2022-06-23 00:00:00	19960.91016	20421.00000	19881.33984	20292.75000

5000 rows × 5 columns

```
#plotting the graph of bitcoin from the dataset

bitcoin_prices = pd.to_numeric(data_final['close'], errors = 'coerce').values

plt.figure(figsize=(15,6))
plt.plot(bitcoin_prices, label='Actual Bitcoin Prices')
plt.title('Bitcoin Prices')
plt.xlabel('30min Time Interval')
plt.ylabel('Price')
plt.legend()
plt.show()
```

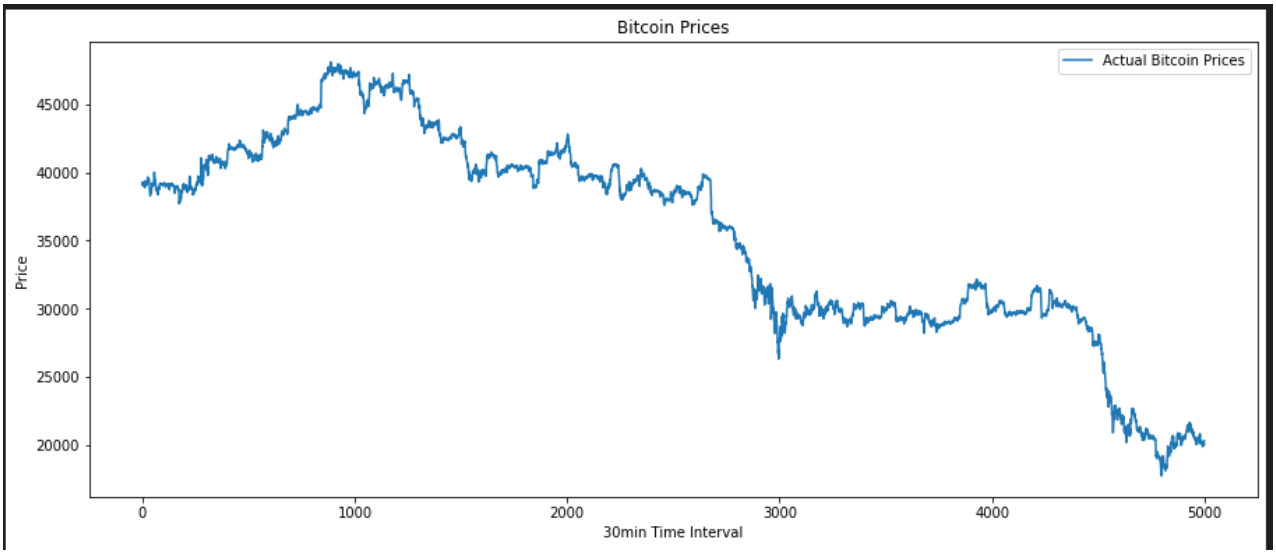

Fig 1.6: Plot for price trend analysis

```
#squishing our price data in the range [0, 1], this helps our optimization
algorithm converge faster

scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(data_final['close'].values.reshape(-1,1))
scaled_data
```

```
array([[0.70788167],
       [0.70185516],
       [0.70896   ],
       ...,
       [0.07789955],
       [0.07411455],
       [0.08525032]])
```

```
time_intervals_to_train = 24
prediction_interval = 12
x_train=[]
y_train=[]

for i in range(time_intervals_to_train, len(scaled_data)-prediction_interval):
  x_train.append(scaled_data[i - time_intervals_to_train: i, 0])
  y_train.append(scaled_data[i + prediction_interval, 0])

x_train = np.array(x_train)
y_train = np.array(y_train)

#reshaping x_train into 3d array because LSTM requires it

x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
x_train.shape
```

```
(4964, 24, 1)
```

```
model = Sequential()

model.add(LSTM(128, return_sequences = True, input_shape =(x_train.shape[1],1),
activation='relu'))
#Set return_sequences=True, so the output will still be a sequence.

model.add(Dropout(0.4))
#Dropout implemented in LSTM as it will improve performance and reduce
overfitting.

model.add(LSTM(64, return_sequences = True, activation='relu'))
model.add(Dropout(0.3))
model.add(LSTM(32, activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(1, activation = 'sigmoid'))
#Above is added at the end to get the output, where 1 is the number of features in the
input data.
```

```
model.compile(loss= 'mean_squared_error', optimizer='adam', metrics = ['accuracy'])
#using the loss function as mean_squared_error and optimizer as adam

model.fit(x_train, y_train, epochs = 10, batch_size=64)
```

```
Epoch 1/10
78/78 [==============================] - 19s 125ms/step - loss: 0.0303
Epoch 2/10
78/78 [==============================] - 10s 124ms/step - loss: 0.0025
Epoch 3/10
78/78 [==============================] - 10s 125ms/step - loss: 0.0018
Epoch 4/10
78/78 [==============================] - 10s 127ms/step - loss: 0.0016
Epoch 5/10
78/78 [==============================] - 11s 144ms/step - loss: 0.0016
Epoch 6/10
78/78 [==============================] - 10s 123ms/step - loss: 0.0015
Epoch 7/10
78/78 [==============================] - 10s 124ms/step - loss: 0.0015
Epoch 8/10
78/78 [==============================] - 10s 123ms/step - loss: 0.0015
Epoch 9/10
78/78 [==============================] - 11s 137ms/step - loss: 0.0014
Epoch 10/10
78/78 [==============================] - 10s 125ms/step - loss: 0.0014
```

```
#model has been trained with the given dataset
#now we will obtain a new dataset to test the accuracy of our model

test_start = '2022-06-23 00:00:00'
test_end = '2022-08-27 00:00:00'
test_api_url = f'https://api.twelvedata.com/time_series?apikey={api_key}&interval=
{interval}&symbol={symbol}&order={order}&start_date={test_start}&end_date={test_end}'
test_data = requests.get(test_api_url).json()
test_data_final = pd.DataFrame(test_data['values'])
test_data_final
```

	datetime	open	high	low	close
0	2022-06-23 00:00:00	19960.91016	20421.00000	19881.33984	20292.75000
1	2022-06-23 00:30:00	20322.49023	20478.52930	20217.11914	20409.18945
2	2022-06-23 01:00:00	20402.19922	20542.83984	20318.98047	20505.38086
3	2022-06-23 01:30:00	20490.02930	20576.08008	20332.23047	20361.96094
4	2022-06-23 02:00:00	20358.77930	20441.22070	20327.46094	20352.48047
...	...	...	...	...	...
3116	2022-08-26 22:00:00	20645.65039	20683.08008	20616.48047	20661.18945
3117	2022-08-26 22:30:00	20660.41992	20675.50977	20595.06055	20622.83008
3118	2022-08-26 23:00:00	20618.53906	20633.24023	20181.69922	20261.36914
3119	2022-08-26 23:30:00	20288.33008	20297.83008	20121.32031	20255.16992
3120	2022-08-27 00:00:00	20253.85938	20270.97070	20149.33984	20269.07031

3121 rows × 5 columns

```
bitcoin_prices = pd.to_numeric(test_data_final['close'], errors = 'coerce').values
test_inputs = test_data_final['close'].values
test_inputs = test_inputs.reshape(-1,1)
model_inputs = scaler.fit_transform(test_inputs)
x_test = []
for x in range(time_intervals_to_train, len(model_inputs)):
  x_test.append(model_inputs[x-time_intervals_to_train:x,0])

x_test = np.array(x_test)

#reshaping x_test into 3d array because LSTM requires it
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1))

x_test.shape
```

```
(3097, 24, 1)
```

```
#predicting prices of x_test using our model

prediction_prices = model.predict(x_test)
prediction_prices = scaler.inverse_transform(prediction_prices)
prediction_prices.shape
```

```
(3097, 1)
```

#plotting the graph of predicted prices by the model vs actual prices during the same time period

```
plt.figure(figsize=(15,6))
plt.plot(bitcoin_prices, label='Actual Bitcoin Prices')
plt.plot(prediction_prices, label='Prediction Prices')
plt.title('Predicting Bitcoin Prices')
plt.xlabel('30min Time Interval')
plt.ylabel('Price')
plt.legend()
plt.show()
```

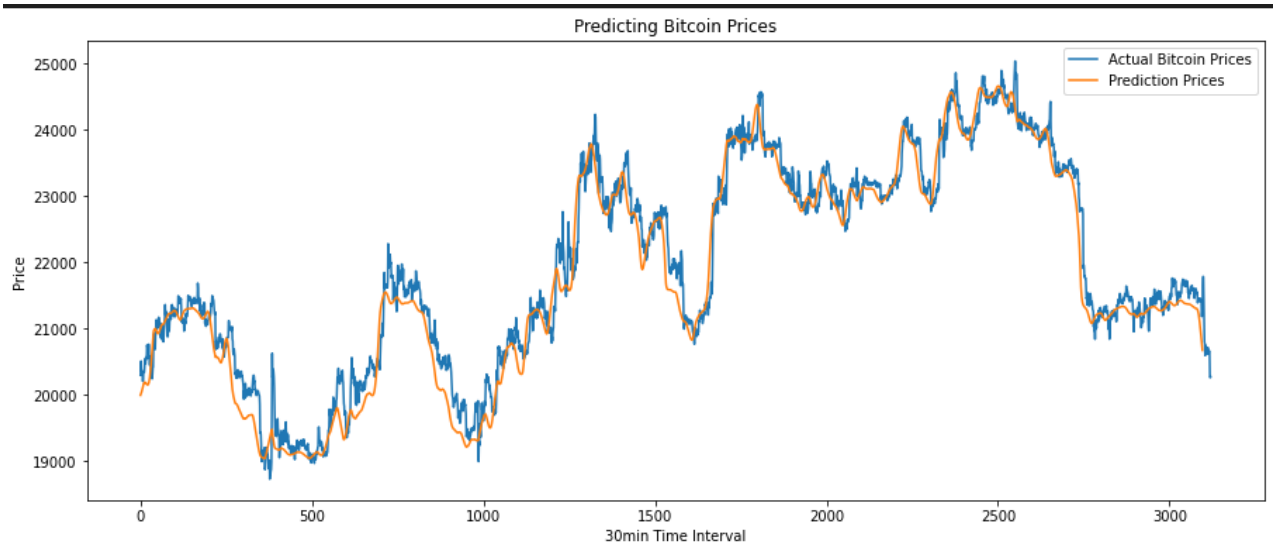

Fig 1.7: Plot of Predicted Prices and actual prices

#taking the last 24 values of the dataset to predict the value of Bitcoin after 6 hours

```
last_data = model_inputs[len(model_inputs) -
time_intervals_to_train:len(model_inputs)+1, 0]
last_data = np.array(last_data)
last_data
```

```
array([0.46198065, 0.48459583, 0.48336856, 0.43237082, 0.38673685,
       0.32890467, 0.32377691, 0.29576674, 0.31450679, 0.31984178,
       0.30606774, 0.29848388, 0.31195079, 0.30661889, 0.31093105,
       0.31456215, 0.30573061, 0.29913339, 0.3043258 , 0.30688798,
       0.30081347, 0.24357328, 0.24259159, 0.24479282])
```

#reshaping last_data so that it fits in our model

```
last_data = np.reshape(last_data, (1, last_data.shape[0], 1))
last_data.shape
```

```
(1, 24, 1)
```

```
prediction = model.predict(last_data)
prediction
```

```
array([[0.29890144]], dtype=float32)
```

```
prediction = scaler.inverse_transform(prediction)
print("Bitcoin Price after 6 hours will be :")
print(prediction)
```

```
Bitcoin Price after 6 hours will be :
[[20610.756]]
```

Simulation Results and Analysis

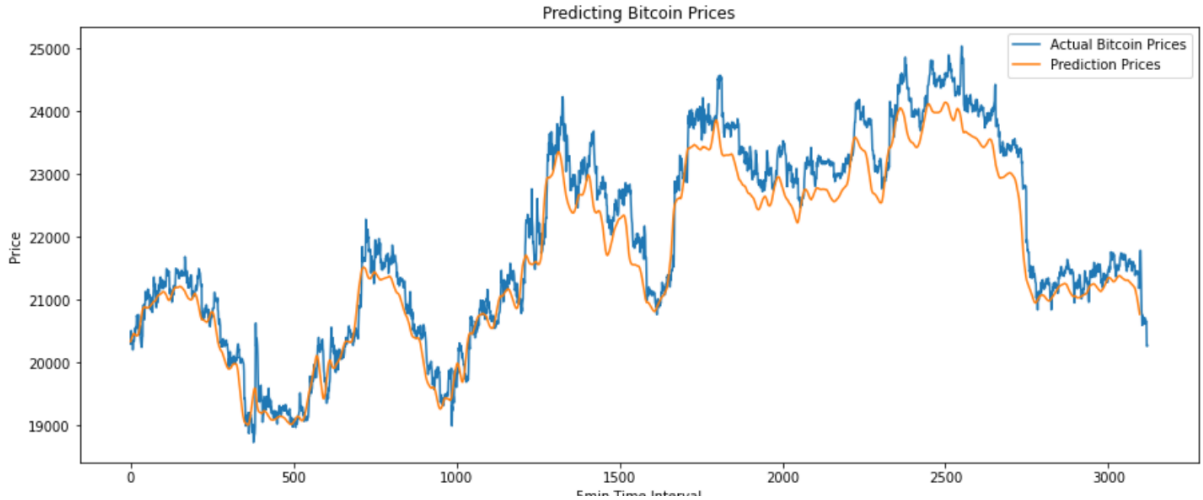

Fig 2.1: Plot of Predicted Prices and actual prices

Though not perfect, LSTMs seem to be able to predict bitcoin price behavior correctly most of the time. Note that you are making predictions of the trend based price, not the true prices. This is okay, because you're predicting the price movement, not the prices themselves.

CONCLUSION

- We obtained the dataset from “twelvedata” and processed it in between the range [0,1] using MinMaxScaler function of the sklearn module.
- We analyzed the trends and observed randomness in prices
- We used LSTM for Bitcoin Price behaviour prediction.
- We plotted the graphs of our predicted values vs actual values to compare the accuracy of our model.

FUTURE SCOPE

The future scope of this project is in

1. Improving the model for better prediction using sentiment analysis from twitter tweets and bitcoin news. Political and regulation related news may also effect bitcoin price

2. Using different learning rate would give us better result and this model could be connected with real time investment account for automation of investments.

3. Bot creation which would advice novice investors when to hold or when to sell bitcoins for maximum profits.

REFERENCES

[1] https://twelvedata.com/account/market-data
[2]https://changelly.com/blog/bitcoin-price-prediction/
[3]https://towardsdatascience.com/cryptocurrency-price-prediction-using-lstms-tensorflow-for-hackers-part-iii-264fcdbccd3f