Bitcoin Price Prediction using ARIMA Time Series model and Deep Learning LSTM Model

For Academic purpose only

What is Bitcoin ?

Bitcoin (₿) is a decentralized cryptocurrency that can be sent from one user to another through the bitcoin peer-to-peer network without intermediaries

Transactions are verified and recorded in a public distributed ledger called **Blockchain**.

Bitcoins are created as a reward for a process known as **mining** and can be exchanged for other currencies, products, and services.

Bitcoin Current Price : 20,389.64 USD

Problem Statement

This project is about building a model that is efficient in predicting Bitcoin Prices

The dataset for this project will be taken from the **yahoo finance API**

The dataset is Bitcoin vs USD (BTC-USD) prices

The dataset starts from **January 1, 2016 to June 26th ,2022**

Approach

In this project , the close price of Bitcoin is the category that is being focused on

The model is coded in Python and the tool used is Google Colab

The prediction has been done using 2 models

The first model is the ARIMA time series model where tests were conducted to check if data is stationary

The data in this case being non stationary , the data has been decomposed and ready to be passed to the Auto Arima model

Also an ARIMA model was build with one case where the values of p,d,q were assumed

Approach cond.

p is the number of autoregressive terms, **d** is the number of nonseasonal differences needed for stationarity, and. **q** is the number of lagged forecast errors in the prediction equation.

LSTM Model

The second model that was built for prediction is the **Recurrent Neural Network** using **LSTM(Long Short Term Memory)**

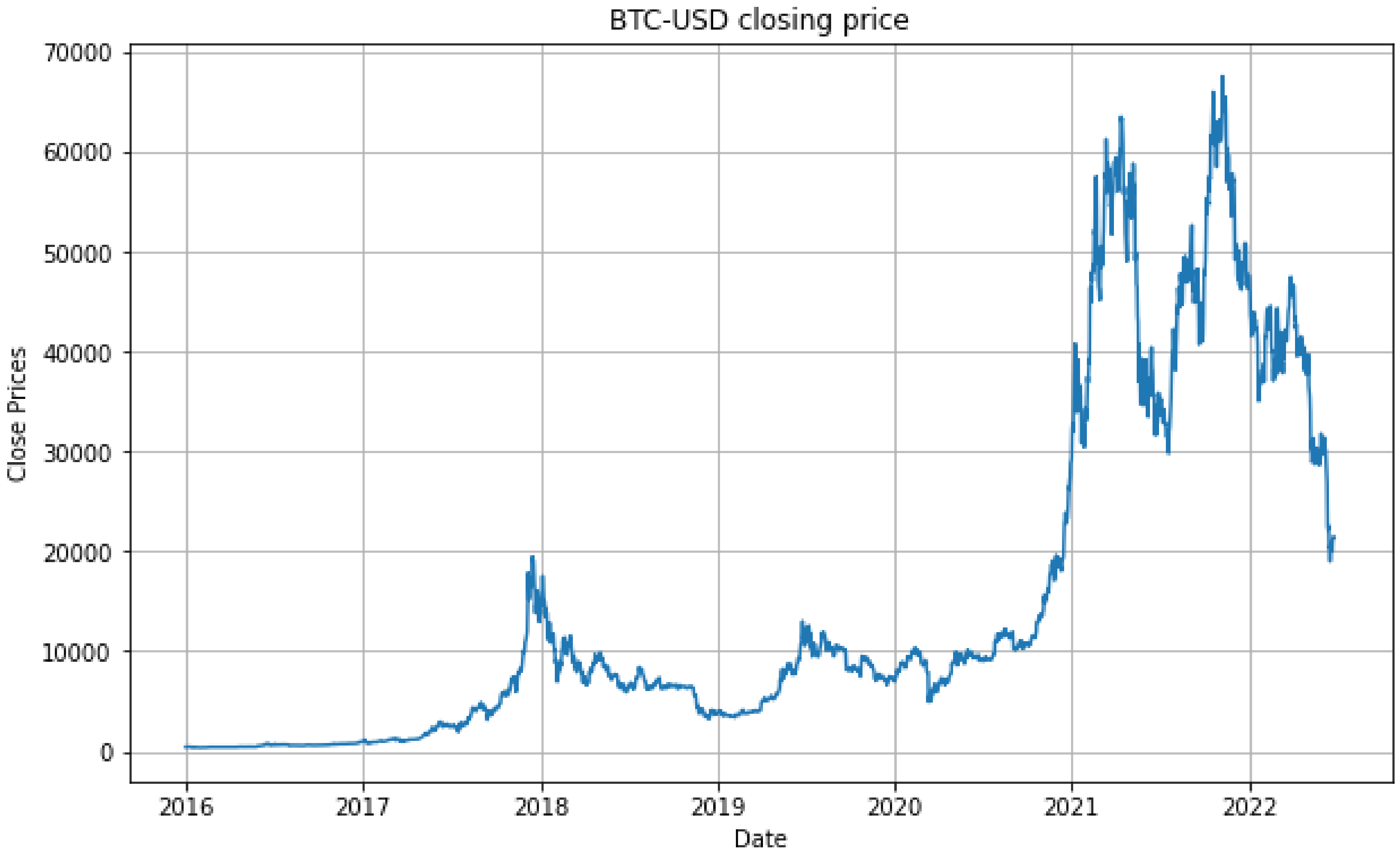
The necessary preprocessing steps were done on the data and trained and the model was built , trained and used for prediction

Summary of dataset

	High	Low	Open	Close	Volume	Adj Close
Date						
2016-01-01	436.246002	427.515015	430.721008	434.334015	36278900	434.334015
2016-01-02	436.062012	431.869995	434.622009	433.437988	30096600	433.437988
2016-01-03	433.743011	424.705994	433.578003	430.010986	39633800	430.010986
2016-01-04	434.516998	429.084015	430.061005	433.091003	38477500	433.091003
2016-01-05	434.182007	429.675995	433.069000	431.959991	34522600	431.959991

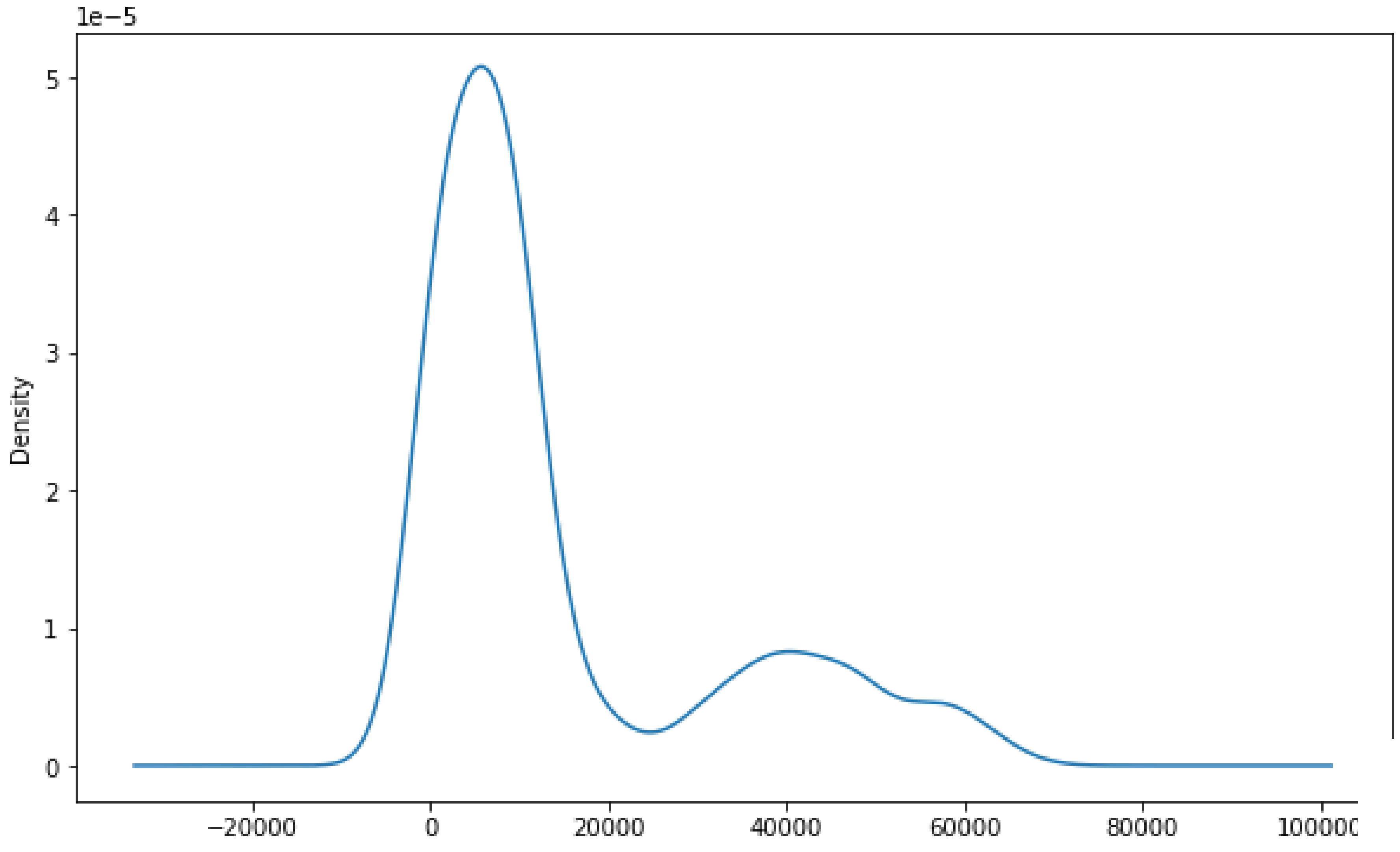
Open - The brice of the bitcoin at the beginning of the day
High - Maximum price at the given time period
Low - Minimum price at the given time period
Close - Price of the bitcoin in USD at the close of the day
Volume - how much - in monetary terms - a given cryptocurrency has traded over a period of time
Adj Close - the closing price after adjustments

Exploratory Data Analysis



Visualising the stock's daily closing price over the past 5 years

We clearly see that the price of Bitcoin rose dramatically in Feb 2021

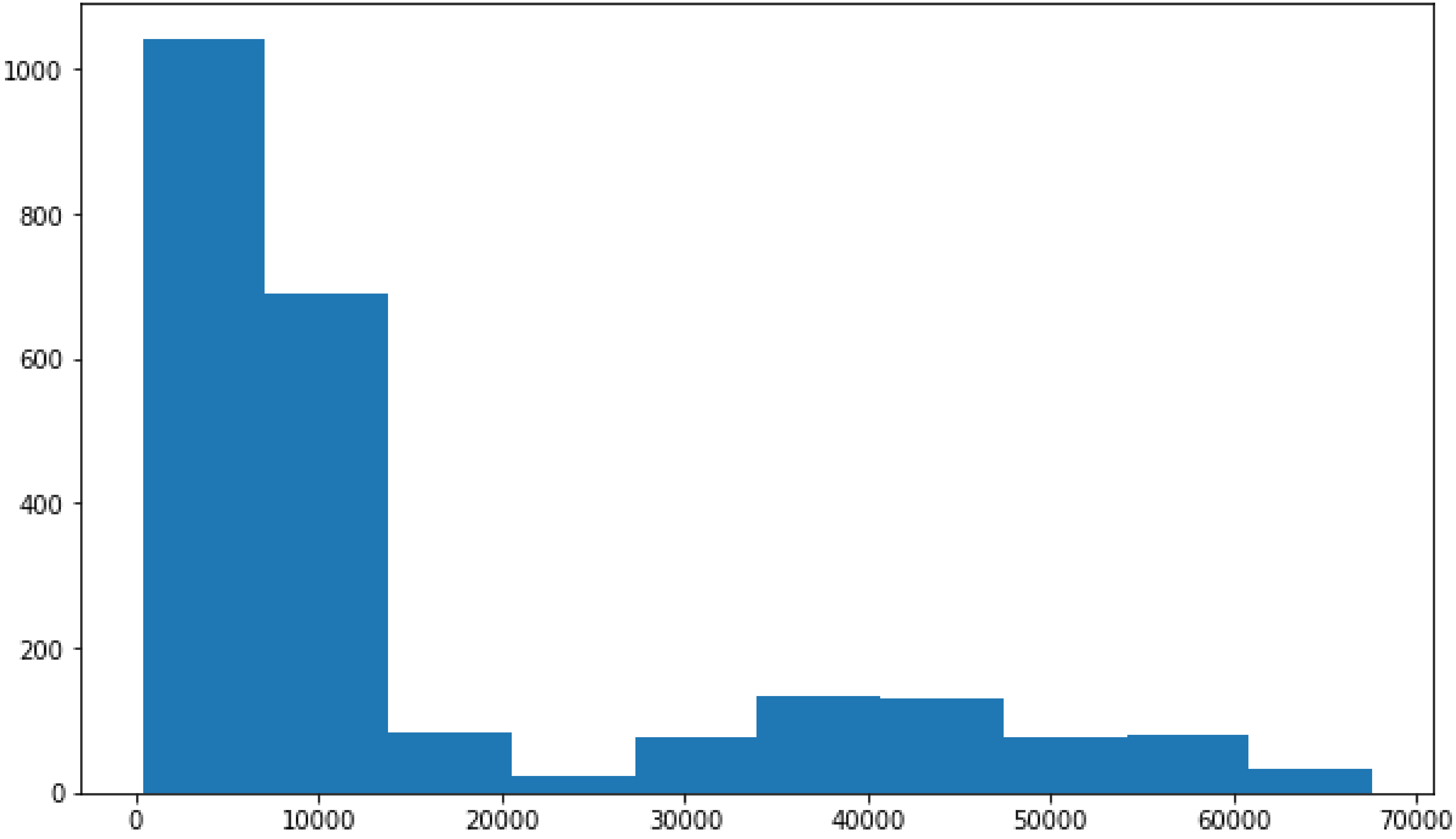


Visualising the data distribution

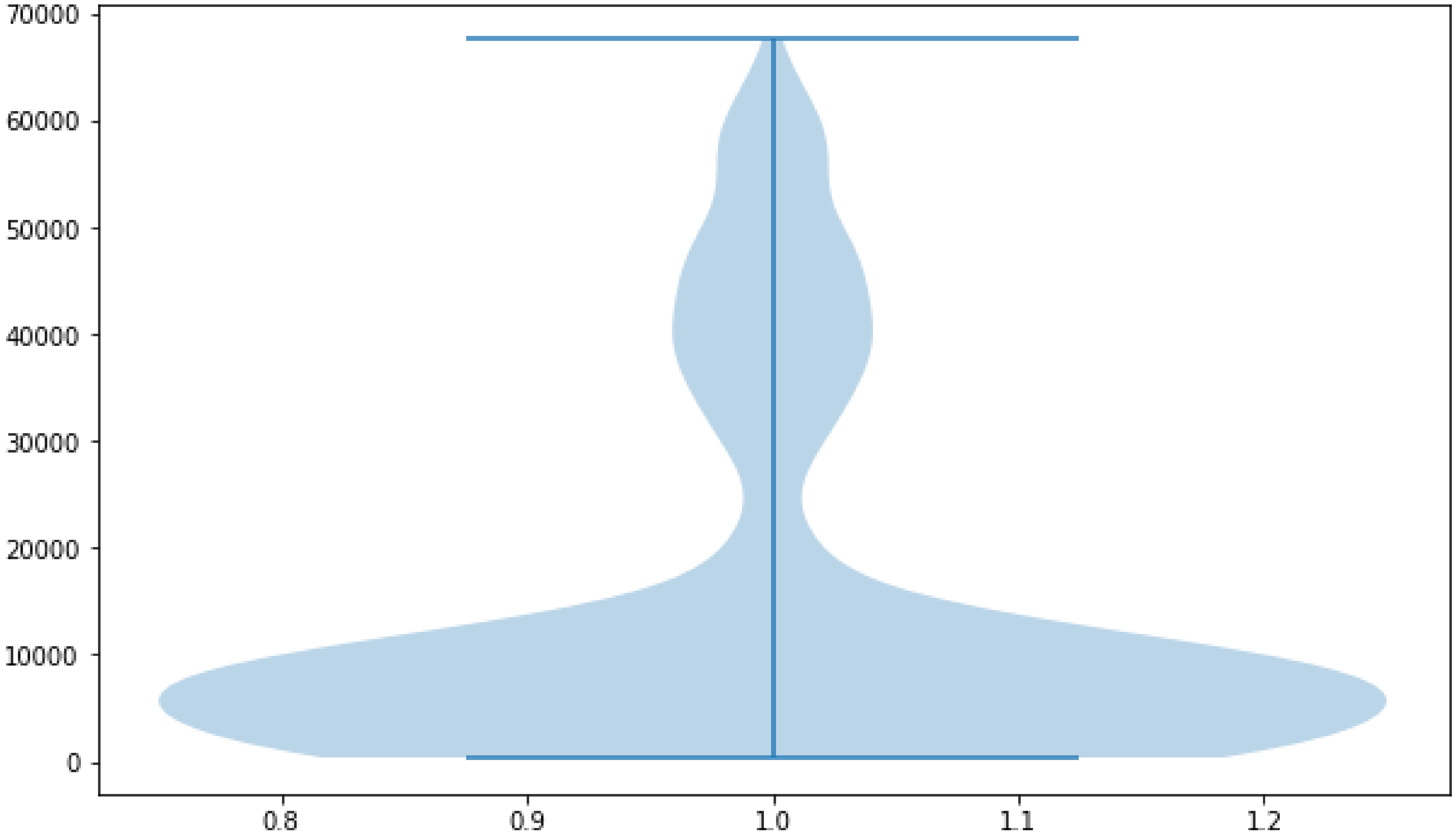
Violin plot to check for outliers

From the previous slide , below is the inference

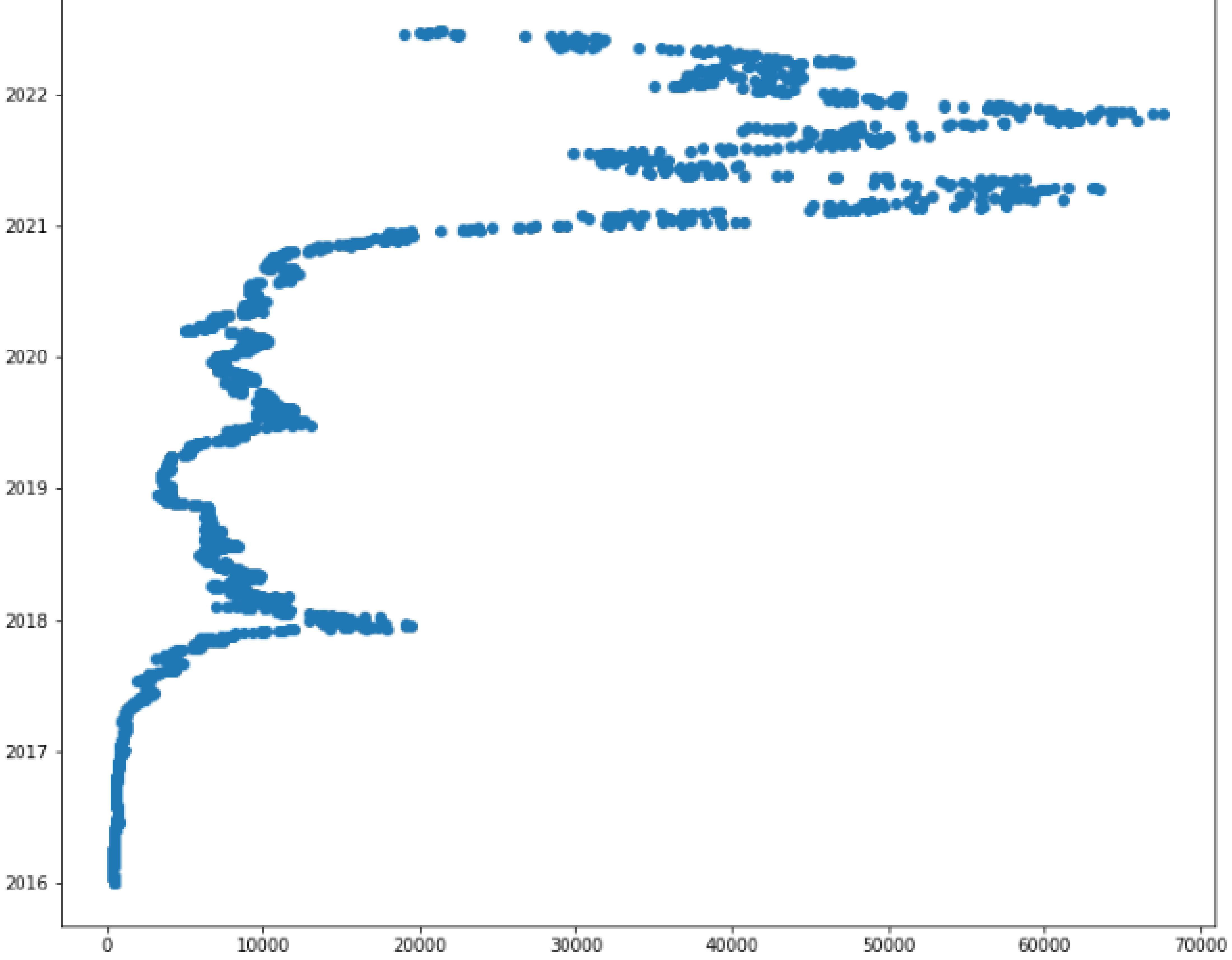
We see that distribution of the closing price is slightly right skewed
and also has several outliers because the sudden spike in the prices in 2021
reaching a maximum of 65000 USD while the previous 5 years had uniform
distribution in between 0 to 20000 USD


0
200
400
600
800
1000
0
10000
20000
30000
40000
50000
60000
70000

Box Plot

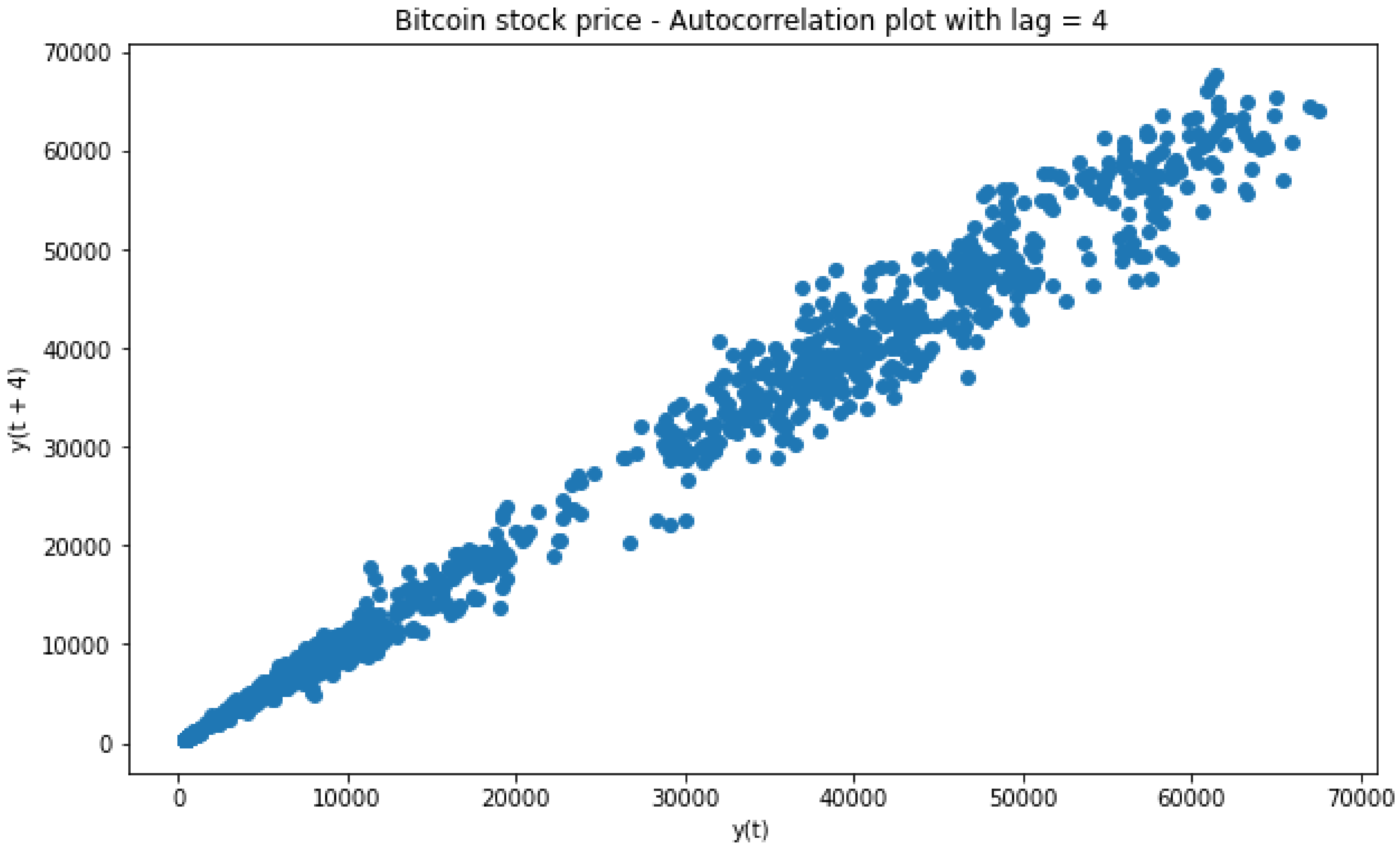


From the violin plot we see that the majority of the datapoints are within the range of 0-30000 USD while very few data points are above 30000 USD till 70000 USD .... we are analysing the last 5 years .... and a remarkable upward spike in prices in the last year(2021) ....



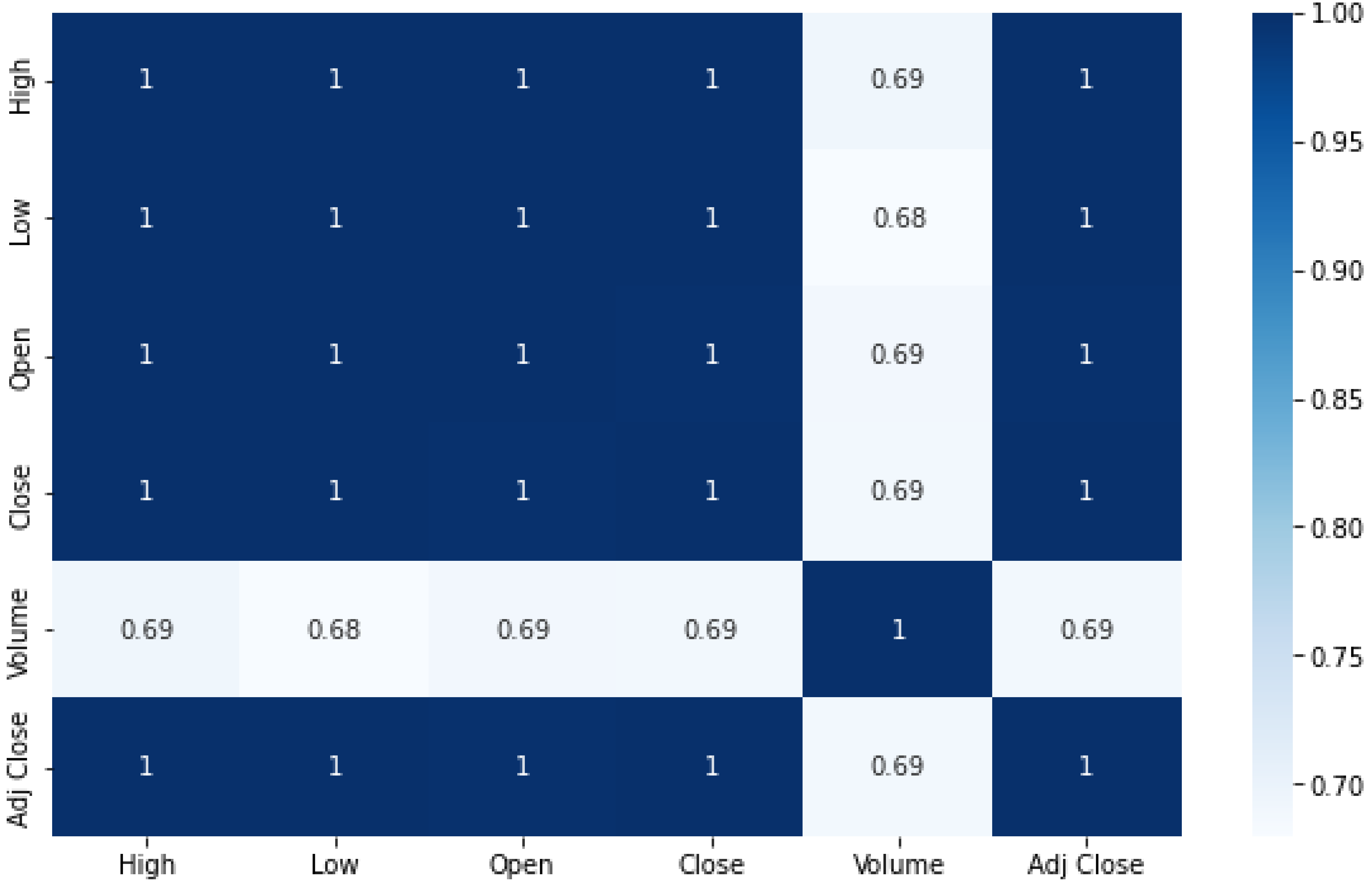
Scatter plot of the datapoints

We see most of the data points in the last five years in the range of 0-30000 USD



We see from the above lag plot the Close price data is linear , hence it is an auto regressive model and also there is no randomness in the data

<matplotlib.axes._subplots.AxesSubplot at 0x7fec7e8eb1d0>



Heat Map

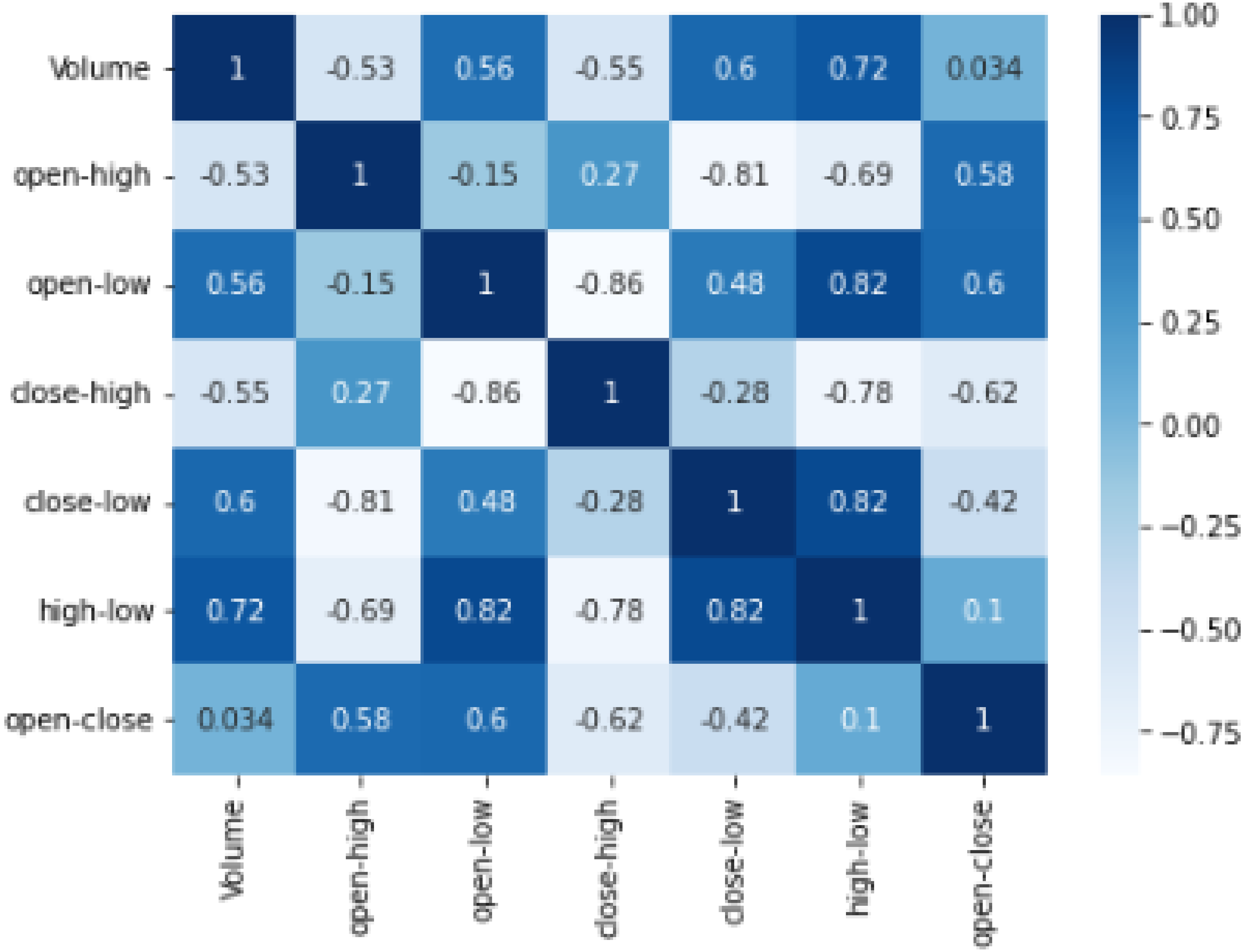
In the heat map in the previous slide , we see that most values are 1 or close to 1 which indicates high correlation . This happens because of very minute difference in the values of the dataset

However in the stock market , even those differences matter

So creating new features to understand more about the dataset

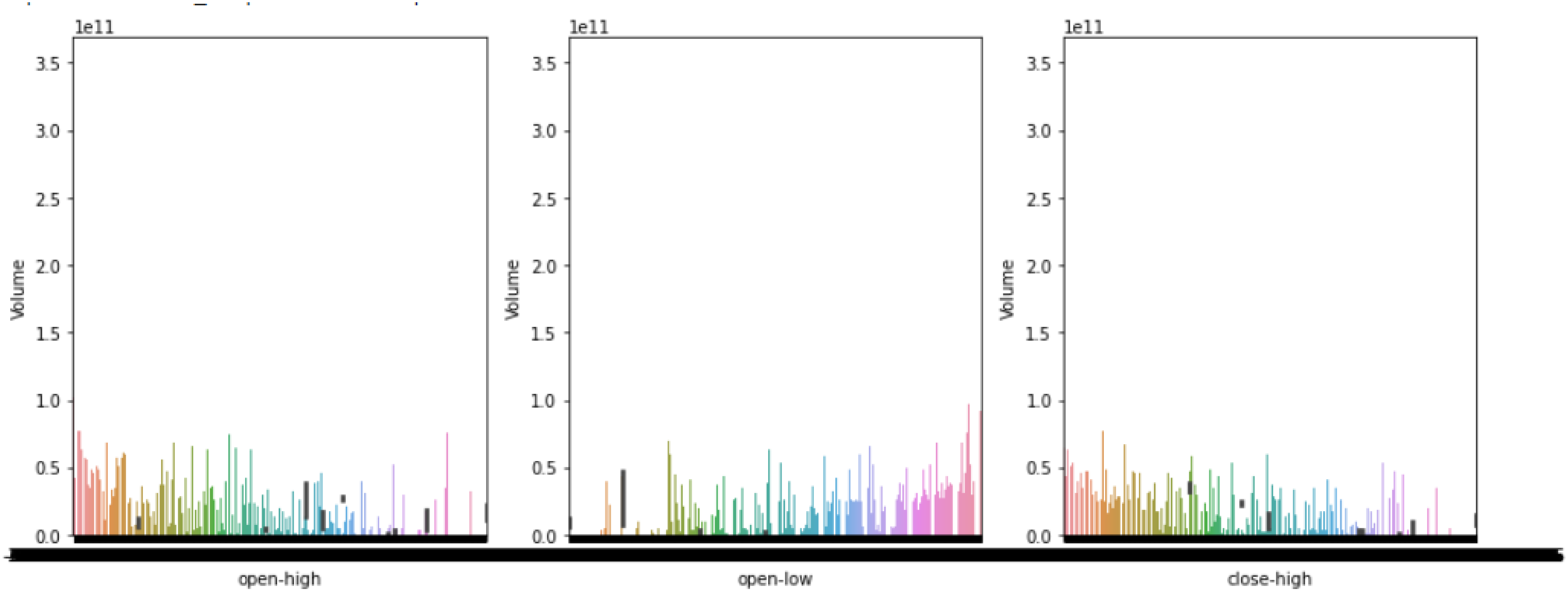
Date	High	Low	Open	Close	Volume	Adj Close	open-high	open-low	close-high	close-low	high-low	open-close
2016-01-01	436.246002	427.515015	430.721008	434.334015	36278900	434.334015	-5.524994	3.205994	-1.911987	6.819000	8.730988	-3.613007
2016-01-02	436.062012	431.869995	434.622009	433.437988	30096600	433.437988	-1.440002	2.752014	-2.624023	1.567993	4.192017	1.184021
2016-01-03	433.743011	424.705994	433.578003	430.010986	39633800	430.010986	-0.165009	8.872009	-3.732025	5.304993	9.037018	3.567017
2016-01-04	434.516998	429.084015	430.061005	433.091003	38477500	433.091003	-4.455994	0.976990	-1.425995	4.006989	5.432983	-3.029999
2016-01-05	434.182007	429.675995	433.069000	431.959991	34522600	431.959991	-1.113007	3.393005	-2.222015	2.283997	4.506012	1.109009

Heat Map with only new features

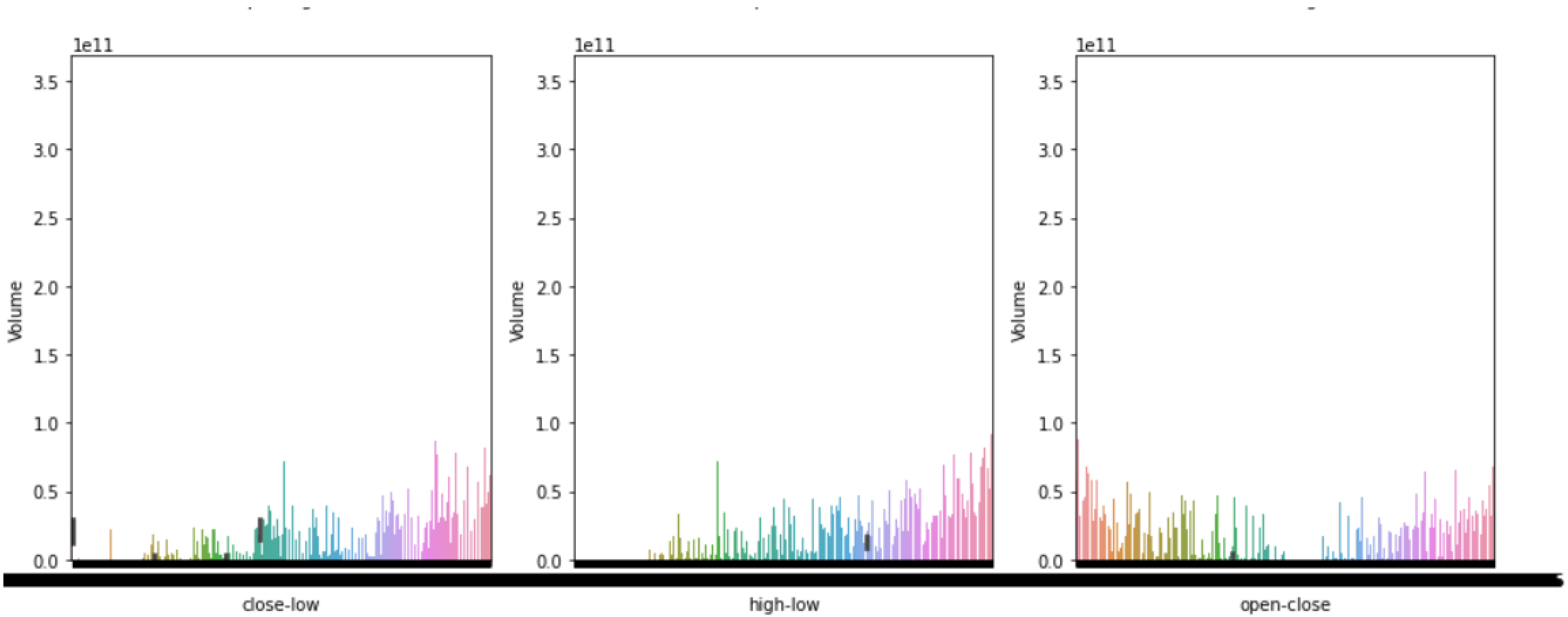


We check the correlation between the new features created and volume tells us how the change in feature impacts the number of stocks traded

visualizing the new features created with volume



visualize the new features created with volume



From the plots in the previous 2 slides , we can make the following inferences

"Open-High" and "Close-high" have a negative correlation

"Open-low" and "Close-low" have a postive correlation

and also "Open-High" and "Close-high" have higher volume for smaller values

while "Open-low" and "Close-low" have higher volume for higher values

We can start building the model now for predictions , as mentioned earlier we will be using both the **ARIMA model and LSTM network to make our predictions**

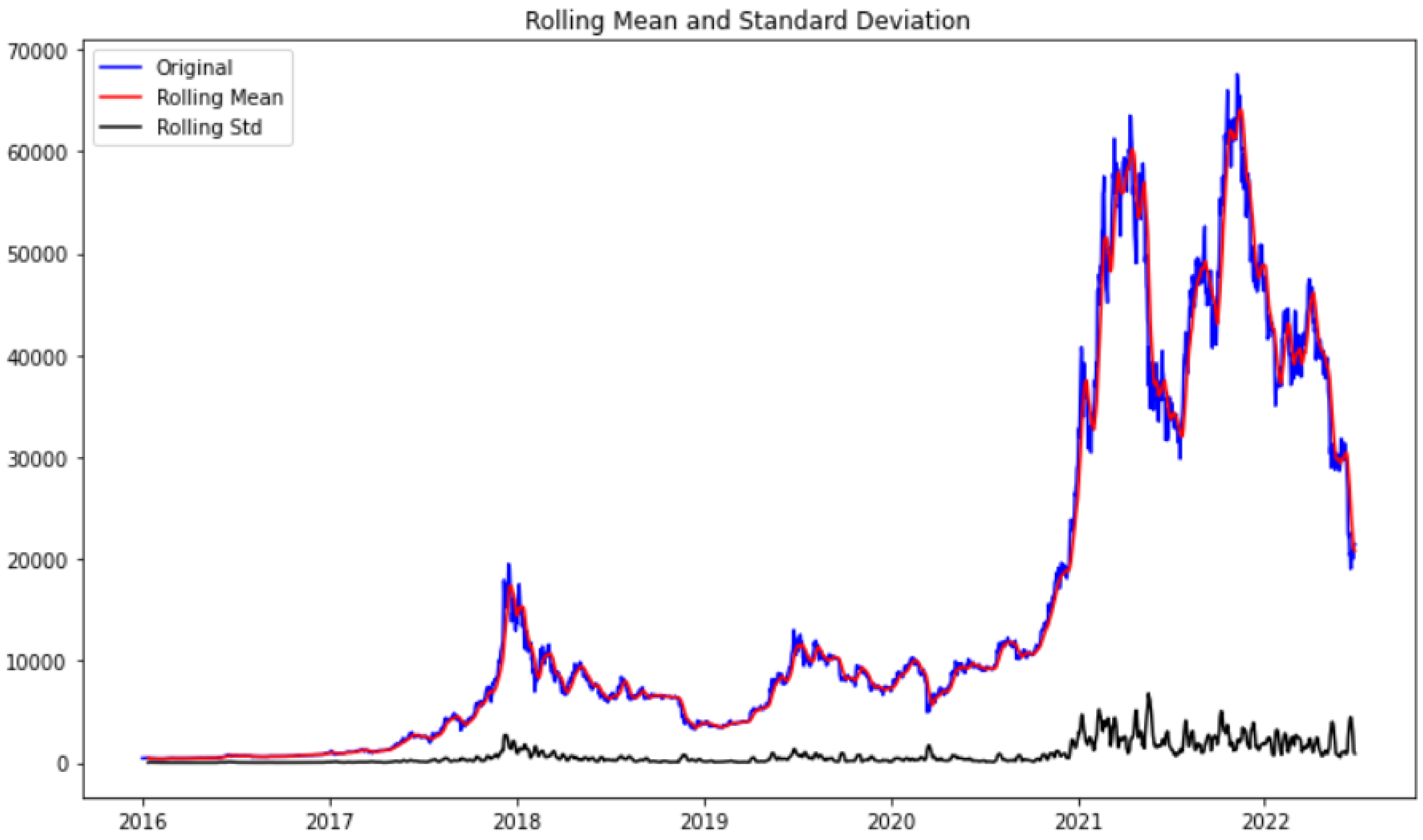
ARIMA MODEL

We will first check if the Bitcoin Close price dataset is stationary

Then we will find out if the data is stationary in two ways

Method1 --> Computing and plotting the moving average and moving standard Deviation with the original dataset with a rolling window of 12 months . if the graph isnt constant we will , data is not stationary

Method2 --> We will perform Dickey Fuller Hypothesis test



Computing and plotting the moving average and moving Standard Deviation with the original dataset with a rolling window of 12 months..... if the graph isn't constant the data is not stationary

Results of Dickey Fuller Test

```
Test Statistics                  -1.547372
p-value                           0.509961
No. of lags used                 27.000000
Number of observations used    2341.000000
critical value (1%)              -3.433146
dtype: float64
Test Statistics                  -1.547372
p-value                           0.509961
No. of lags used                 27.000000
Number of observations used    2341.000000
critical value (1%)              -3.433146
critical value (5%)              -2.862775
dtype: float64
Test Statistics                  -1.547372
p-value                           0.509961
No. of lags used                 27.000000
Number of observations used    2341.000000
critical value (1%)              -3.433146
critical value (5%)              -2.862775
critical value (10%)             -2.567428
dtype: float64
```

We see that the p-value = 0.510096 , which is greater than 0.05

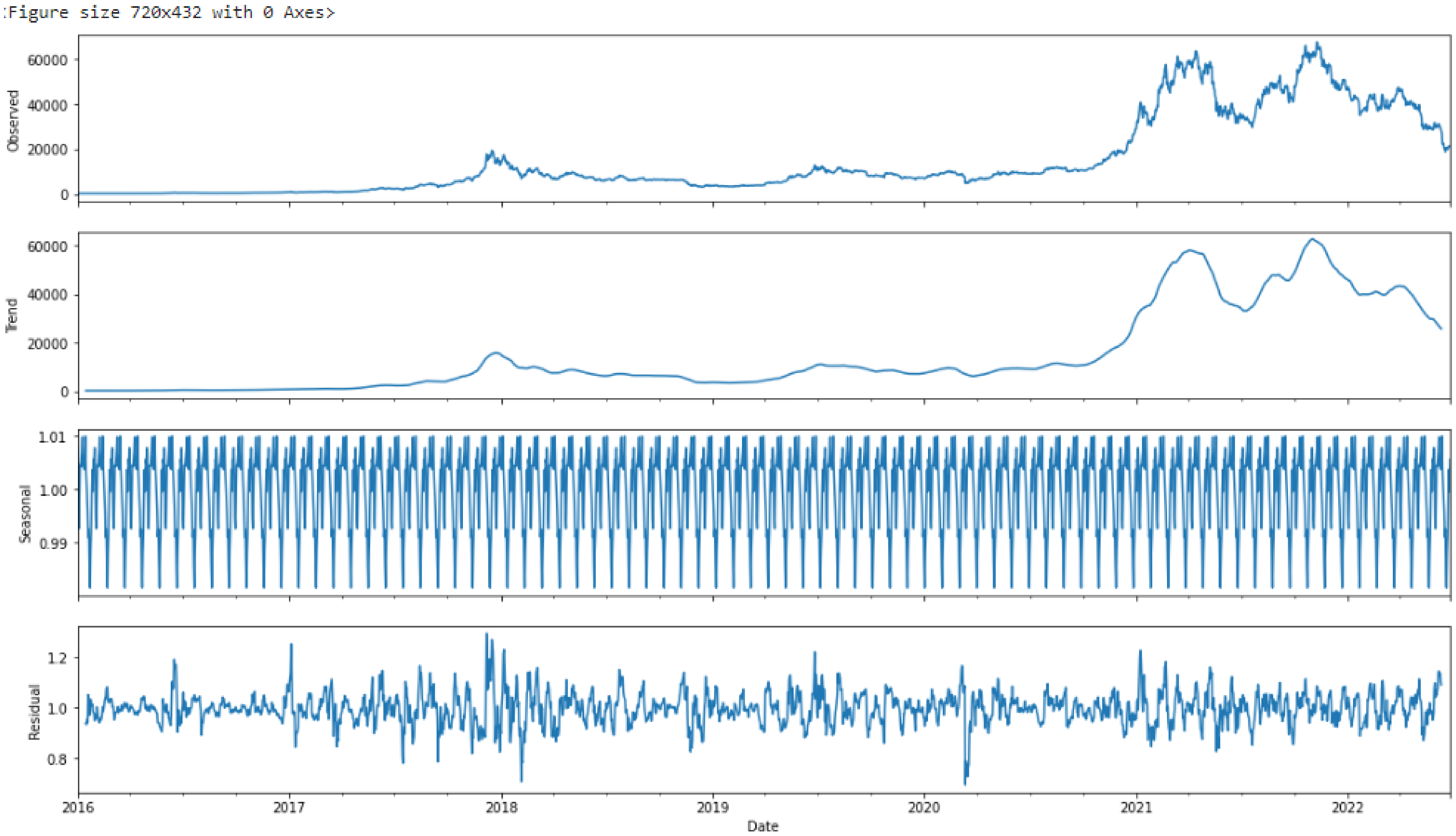
hence we fail to reject the null hypothesis ....

From both the moving average/standard deviation graph and the Dickey Fuller test ,

we can confirm the **data is not stationary**

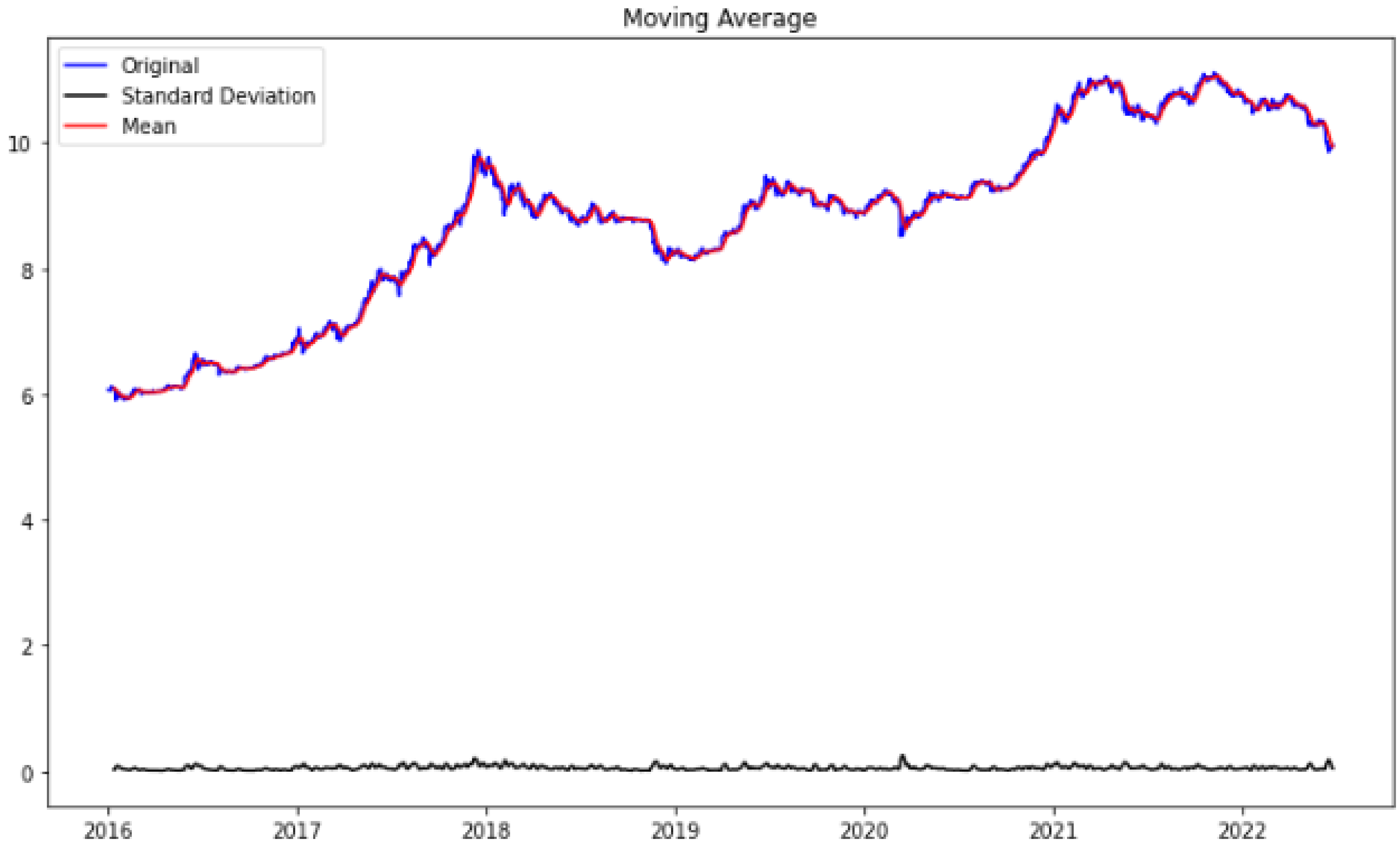
Hence to make it stationary we first decompose the seasonlity and trend using seasonal decompose

Decomposing the data to make it stationary



From the above graph we see that the trend is increasing

To reduce that we transform the data using log transformation and plot the rolling average/standard deviation and log transformned data

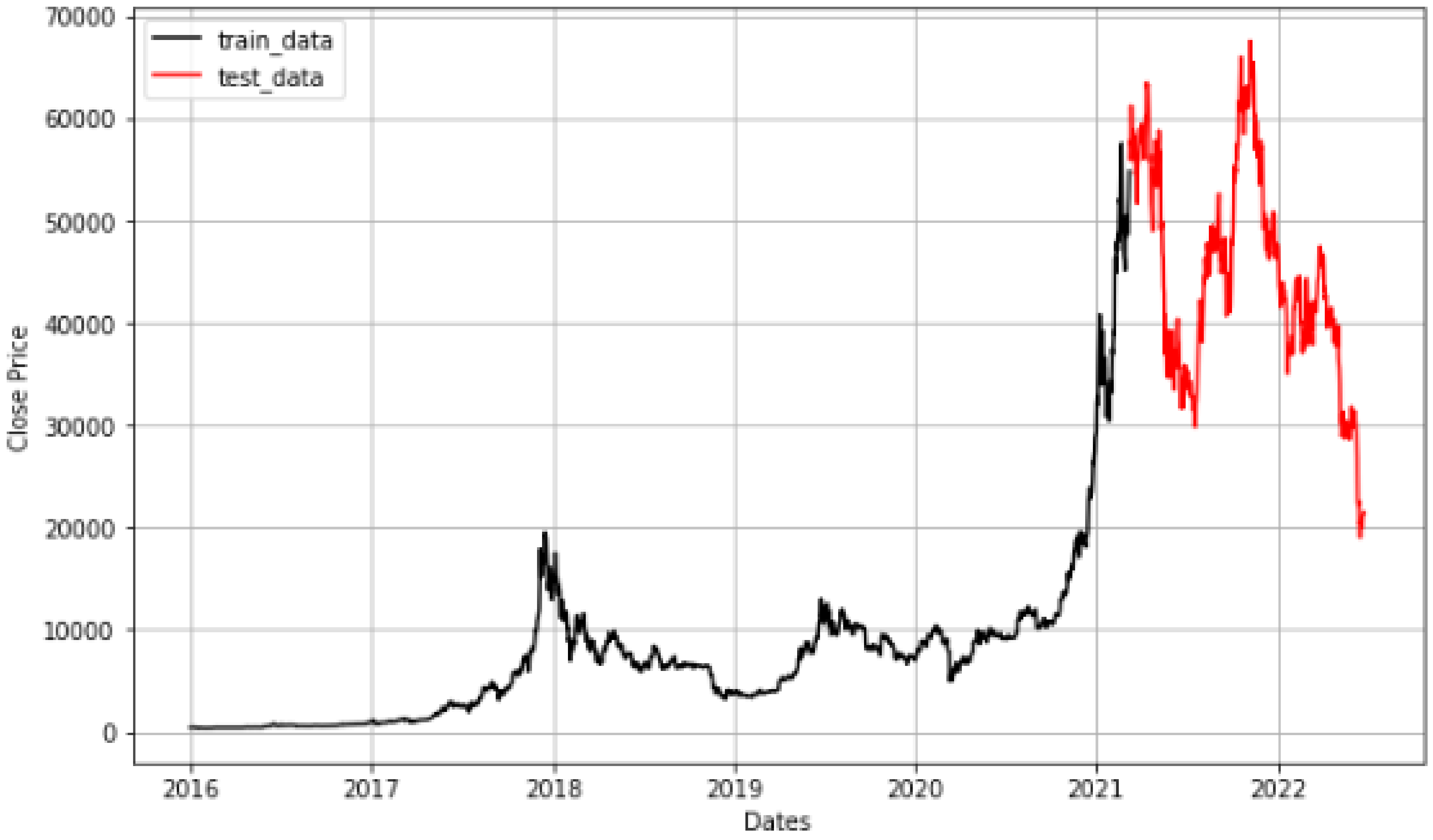


Now the data is stationary and the Auto Arima can be performed

But in this case we build an ARIMA model assuming values of p,d,q

for which we begin by performing train and test split on the original dataset

<matplotlib.legend.Legend at 0x7fec5f91e850>



Train Test split

ARIMA Model Results

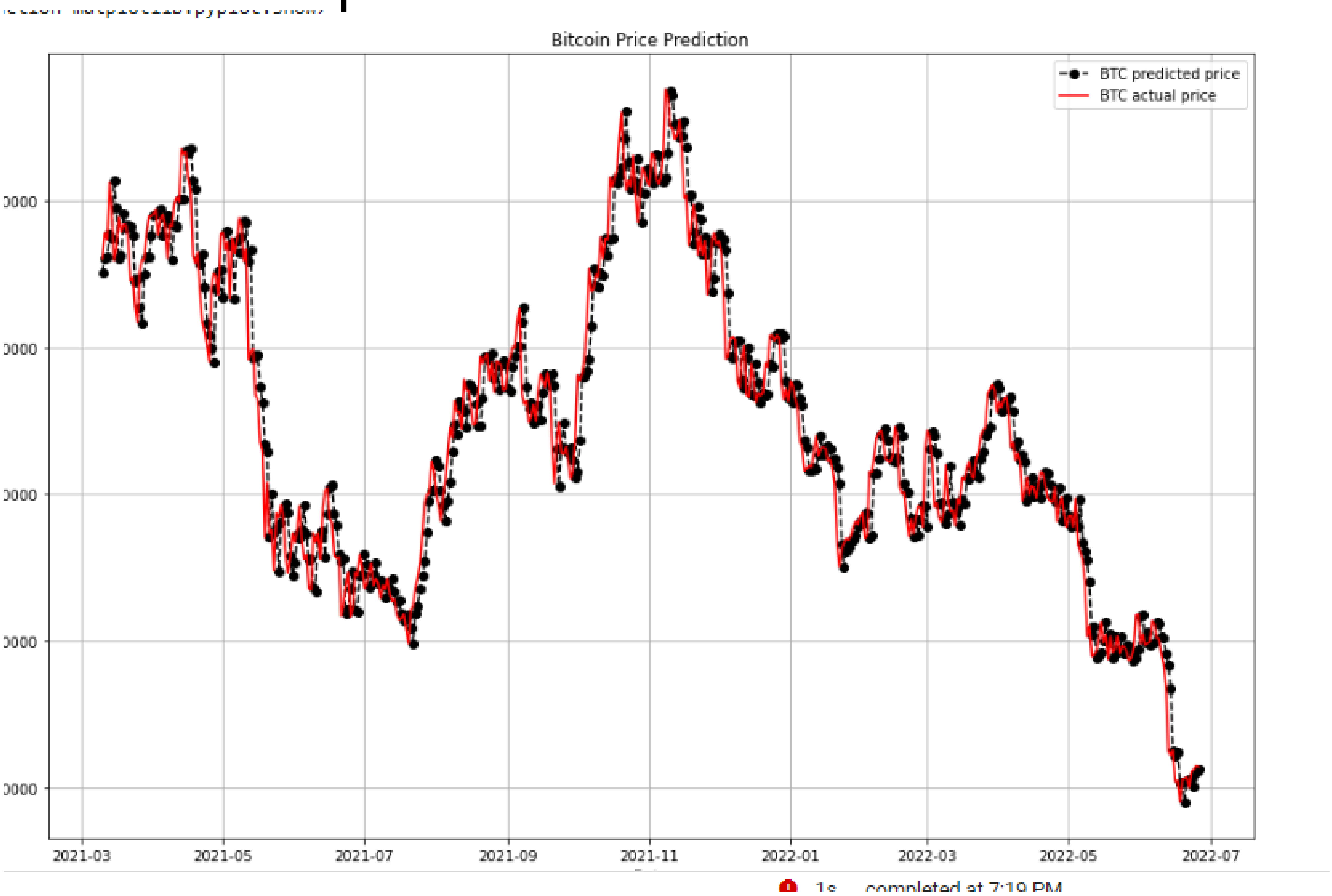
Dep. Variable:	D.y	No. Observations:	2368
Model:	ARIMA(4, 1, 1)	Log Likelihood	-19422.035
Method:	css-mle	S.D. of innovations	882.653
Date:	Sun, 26 Jun 2022	AIC	38858.070
Time:	10:27:47	BIC	38898.458
Sample:	1	HQIC	38872.773

	coef	std err	z	P>|z|	[0.025	0.975]
const	8.8979	19.262	0.462	0.644	-28.854	46.650
ar.L1.D.y	0.3823	0.309	1.238	0.216	-0.223	0.988
ar.L2.D.y	0.0157	0.024	0.660	0.509	-0.031	0.062
ar.L3.D.y	0.0189	0.022	0.857	0.391	-0.024	0.062
ar.L4.D.y	0.0301	0.024	1.272	0.204	-0.016	0.077
ma.L1.D.y	-0.4127	0.309	-1.337	0.181	-1.017	0.192

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	1.6662	-0.0000j	1.6662	-0.0000
AR.2	0.3687	-2.5375j	2.5641	-0.2270
AR.3	0.3687	+2.5375j	2.5641	0.2270
AR.4	-3.0303	-0.0000j	3.0303	-0.5000
MA.1	2.4233	+0.0000j	2.4233	0.0000

Bitcoin prediction vs actual chart



MAPE: 0.04132690290820174

This model is a very good represntation of how we can use a machine learning model to make some predictions as you can see that our model performed quite well , almost predicting the right Bitcoin Price

LSTM Model for Bitcoin Price prediction

We will be using the same dataset from the yahoo finance API using the pandas data reader library

For LSTM Neural Network we will have to scale the data to either range(0,1) or range(-1,1) here we will scale the Bitcoin close price to (0,1) using MixMaxScaler

Define the time frame for train data , first we define the learning days on which the model will train in this case 90 days

Since the learning days is defined as 90 , 90 will be the number of features in X_train dataset and the target variable y_train will have just one feature both with 2277 records

BUILDING NEURAL NETWORK

with 2 LSTM layers and then adding dropout layers to avoid overfitting and finally a Dense layer for the output

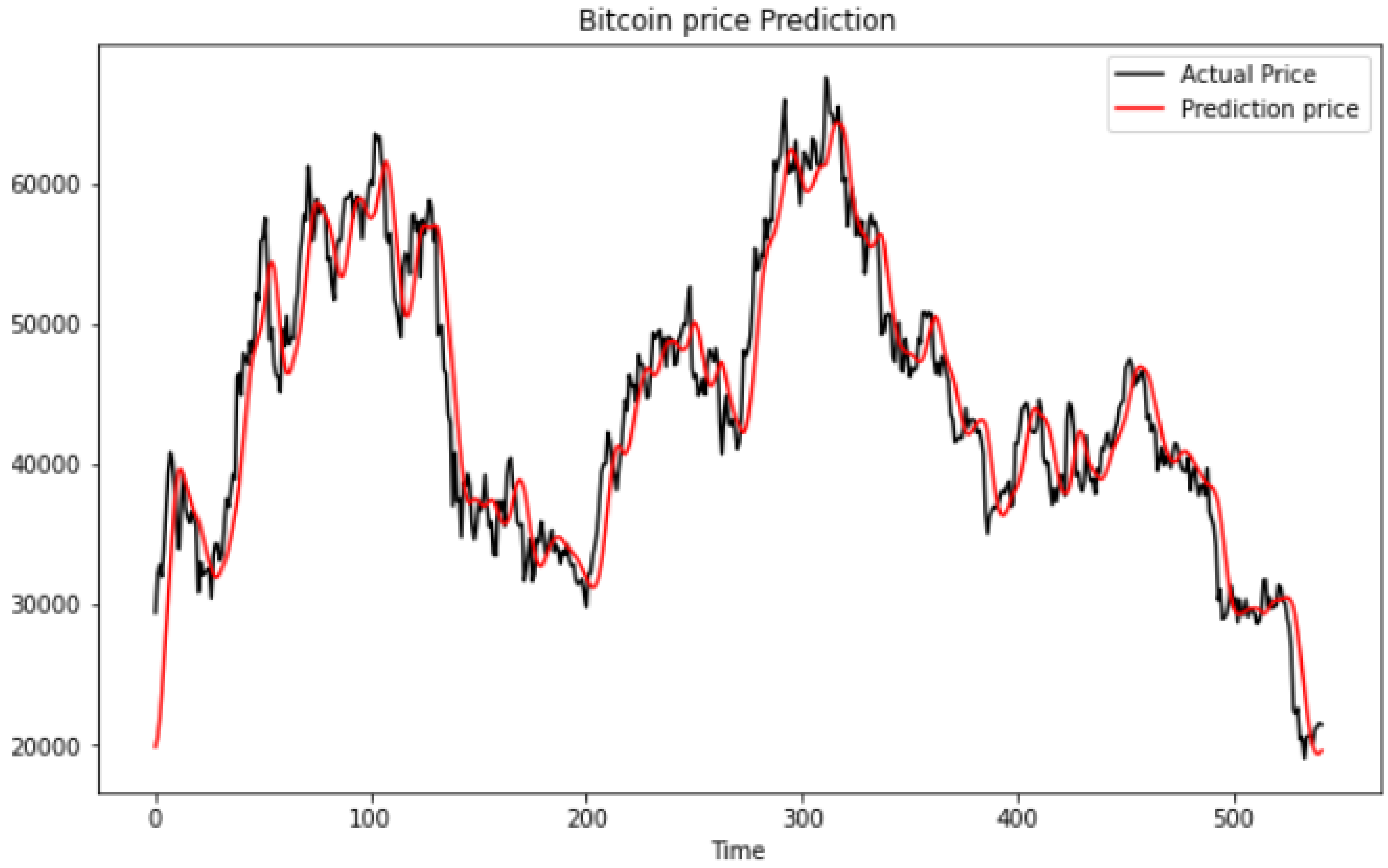
We then train the model with 20 epochs and a batch_size 32

The n we define the time frame for the test data in this case from January 1, 2021 till Jun 26, 2022

the input data for the model will be the total dataset minus (test data minus 90 days)

model_input = total_data-[test_data-90]

We finally make predictions and plot the actual vs predicted values graph and check the predicted value for the next day


Bitcoin price Prediction
Actual Price
Prediction price
60000
50000
40000
30000
20000
0
100
200
300
400
500
Time

Predicted next day price

```
#predict next day

real_data = [input_for_model[len(input_for_model) - learning_days:len(input_for_model) + 1 , 0]]
real_data = np.array(real_data)
real_data = np.reshape(real_data, (real_data.shape[0] , real_data.shape[1] , 1))
```

```
prediction = model.predict(real_data)
prediction = scaler.inverse_transform(prediction)
print(prediction,'USD')
```

```
[[19823.361]] USD
```

Date	High	Low	Open	Close	Volume	Adj Close
2021-01-01	29600.626953	28803.585938	28994.009766	29374.152344	40730301359	29374.152344
2021-01-02	33155.117188	29091.181641	29376.455078	32127.267578	67865420765	32127.267578
2021-01-03	34608.558594	32052.316406	32129.408203	32782.023438	78665235202	32782.023438
2021-01-04	33440.218750	28722.755859	32810.949219	31971.914062	81163475344	31971.914062
2021-01-05	34437.589844	30221.187500	31977.041016	33992.429688	67547324782	33992.429688

The close price during 2021 January

We see that the model has predicted well because the Bitcoin price is decreasing and from the previous slide's graph , our model predicted around 19800 USD which is a decline from the prices during 2021 January

However our model does predict a slightly lower amount during a certain period and a slightly higher amount the rest of the time