TIME SERIES ANALYSIS FOR BITCOIN PRICE PREDICTION USING PROPHET

A UG PROJECT PHASE-1 REPORT

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

EMME ANIL	**19UK1A05N7**
SAMALA HIMAVANTHSAI	**20UK1A0522**
SHAIK SHAREEF	**20UK1A0523**

Under the esteemed guidance of

Mr. A.ASHOK KUMAR

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE
(Affiliated to JNTUH, Hyderabad)
Bollikunta, Warangal – 506005

2019– 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE
BOLLIKUNTA, WARANGAL – 506005 2019 – 2023

CERTIFICATE OF COMPLETION UG PROJECT PHASE-1

This is to certify that the UG Project Phase-1 entitled "**TIME SERIES ANALYSIS FOR BITCOIN PRICE PREDICTION USING PROPHET** ''is being submitted by ***E.ANIL(H.NO:19UK1A05N7),S.HIMAVANTHSAI(H.NO:20UK1A0522),S.SHAREEF(H.NO:20 UK1A0523),***in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** to **Jawaharlal Nehru Technological University** Hyderabad during the academic year **2022-23**, is a record of work carried out by them under the guidance and supervision.

Project Guide
Mr. A.ASHOK KUMAR
(Assistant Professor)

Head of the Department
Dr. R. Naveen Kumar
(Professor)

EXTERNAL

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.P.PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.R.NAVEEN KUMAR**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **Mr. A.ASHOK KUMAR** Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

E.ANIL (19UK1A05N7)
S.HIMAVANTHSAI (20UK1A0522)
S.SHAREEF (20UK1A0523)

CONTENTS

1.ABSTRACT

Bitcoin is a decentralized digital currency created in January 2009..We will go deep into the datasets, do an EDA, feature extraction and predict the price of bitcoin using Stochastic, Machine Learning and Deep Learning models. In this particular work we will be visualising the Time Series data, handling the missing values with various imputation techniques. Feature extraction is performed before model building, the focus of study will be these four models ARIMA, Facebook prophet, XG boost and LSTM. Comparing the models and their evaluation metrics to see how each model have performed

Keywords: Blockchain, Bitcoin, Cryptocurrency, Neural Network.

2.INTRODUCTION

Bitcoin is a cryptographic currency which is utilized worldwide for advanced installment or basically for speculation purposes. Bitcoin is decentralized, for example it isn't possessed by anybody. Exchanges made by Bitcoins are simple as they are not attached to any nation. Speculation should be possible through different commercial centers known as "bitcoin trades". These enable individuals to sell/purchase Bitcoins utilizing various monetary forms.

Bitcoin emerged out of the 2008 global economic crisis when big banks were caught misusing borrowers' money, manipulating the system, and charging exorbitant fees. To address such issues, Bitcoin creators wanted to put the owners of bitcoins in-charge of the transactions, eliminate the middleman, cut high interest rates and transaction fees, and make transactions transparent. They created a distributed network system, where people could control their funds in a transparent way.

The stock market is one of the most volatile data available in terms of Machine learning datasets. Researchers have been long trying to predict the stock market and any breakthrough in this field would result in, literally, the people being able to mint money. Cryptocurrencies, to be specific, have gained a lot of traction in recent years from investors across the globe. There are several reasons why Bitcoin's price history has been so volatile. It is important to learn the factors that affect its market price so you can decide whether to invest, trade, or keep an eye on its development. As with most commodities, investments, assets, or other products, Bitcoin's price is heavily influenced by supply and demand. Since Bitcoin is a rapidly adopted asset by investors and traders, speculation about price movements plays a crucial role in its value at any given point in time.

However, there are issues with bitcoins such as hackers breaking into accounts, high volatility of bitcoins, and long transaction delays. Considering the volatility it's always challenging to predict the bitcoin price.

3.LITERATURE SURVEY

Here the data of BTC per minute was gathered and it was rearranged so that to reflect BTC price in hours, and a total of 56,832 points.24 hrs of data was taken as input and output for the BTC price of the next hour. For this the Multi-Layer Perceptron (MLP) was the unsuitable case for predicting price based on current trends whereas Long Short-Term Memory
(LSTM) gave the best prediction when the past memory and Gated Recurrent Network (GRU) was included.

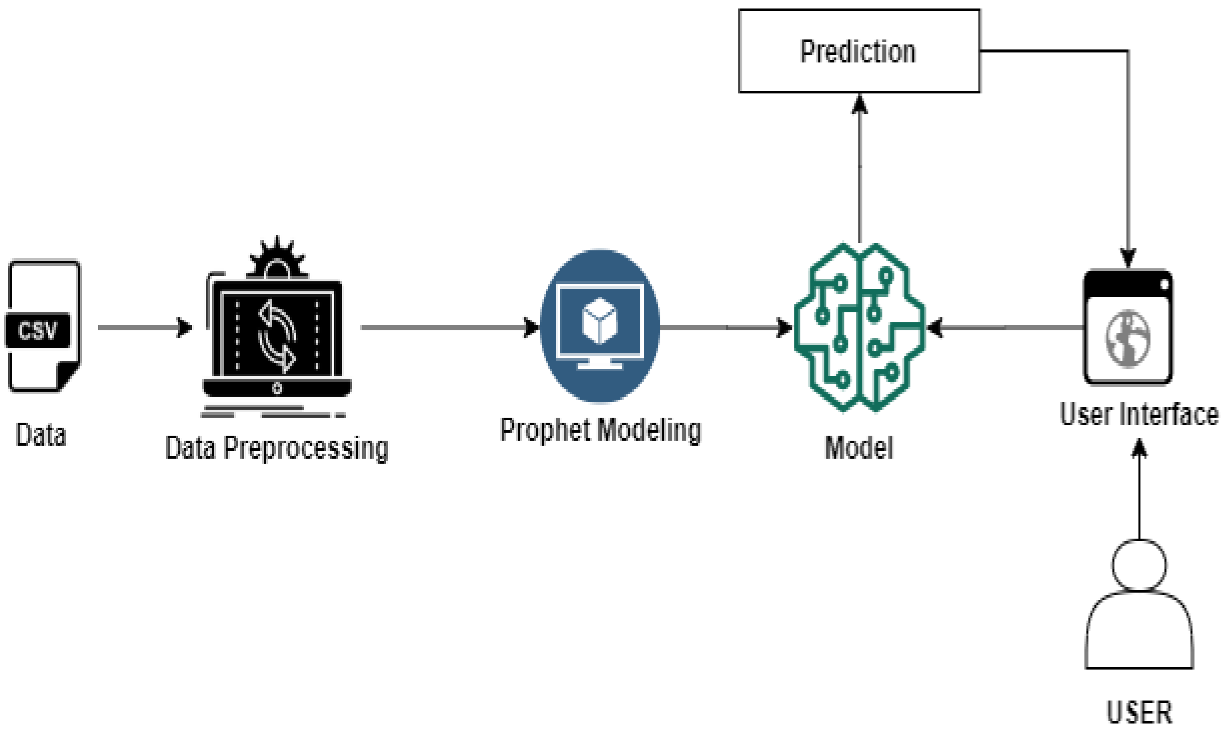
So here to predict BTC price movement and prices in short and medium terms, High performance ML-based classification and regressions are demonstrated. In this the work goes beyond that by using machine learning-based models for one, seven, thirty and ninety days compared to the previous ML-based classification which has only one-day time frames. The models which are developed have 65% accuracy for the next-day forecast and 62-64% for seventh-ninetieth day forecast, while the error percentage is as low as 1.44% while it varies from 2.88-4.10% for horizons of seven to ninety days.

So here firstly what affects the BTC value is taken into consideration. Here there are two phases in the first phase is to understand and identify the daily trends in the BTC market while gaining insight where the data sets consists of various features relating to the BTC price and payment network over the course of five years daily records and in the second phase using the available information it will be possible to predict the sign of daily price change with highest possible accuracy.

This paper looks over user comments in online communities to predict the transactions taking place in crypto. It was predicted that the prices would fluctuate at low costs. The method used approved buying crypto currencies, and gave information on aspects influencing user decisions. Moreover, the simulated investment demonstrated the methods used are applicable while trading in crypto.

In this paper a tree-based classifier is used to perform the predictionDecision Tree Classification method.In their analysis they implement decision tree learning based on cross-entropy impurity function optimization. They perform 1,000 independent rounds of training and prediction.In each of the rounds they select the last 510 sample points as test data and return the win ratio of each type of prediction and they try to clip the trading cutoff to check the real trading performance of the predictive model.
The study revolves around forecasts of crypto currency particular bitcoin price using algorithms like ARIMA-autoregressive integrated moving average and NEAR neural network autoregressive models.
Using the static forecast approach they have tried to forecast next day bitcoin prices both with and without re-

4.BLOCK DIAGRAM



3.1 HARDWARE / SOFTWARE DESIGNING

The hardware required for the development of this

project is: Processor: CoreTMi59300H

Processor speed :2.4GHz RAM Size :8 GB DDR

System Type : X64-based processor

SOFTWARE DESIGNING:

The software required for the development of this project is:

Desktop GUI: Anaconda Navigator

Operating system : Windows 10

Front end : HTML, CSS,JAVASCRIPT
Programming : PYTHON

Cloud Computing Service : IBM Cloud Services

5.EXPERIMENTAL INVESTIGATION

IMPORTING AND READING THE DATASET

Import Libraries

import the necessary libraries for data pre-processing, forecasting using FbProphet, etc.

- It is important to import all the necessary libraries such as pandas,plotly, yahoo finance & Fbprophet.
- Pandas- It is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool, built on top of the Python programming language.
- Plotly- The Plotly Python library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases. Built on top of the Plotly JavaScript library.
- Yahoo Finance- Download Market data from the yfinance module.

Import Dataset

Download the real-time data from the Yahoo Finance library where we need to pass three parameters in the yahoo finance download function i.e. abbreviation name of the cryptocurrency, start date, and today date then we stored it into a variable called df.

Check the entire dataset.

1. Date:- Datewise Information related to the quote currency.
2. Open:- The opening price of the time interval in the quote currency (For BTC/USD, the price would be USD).
3. High: Highest price reached during the time interval, in the quote currency.
4. Low: Lowest price reached during the time interval, in the quote currency.
5. Close:- The closing price of the time interval, in the quote currency.
6. Adj Close:- Final prices of the time interval, in the quote currency.
7. Volume: Quantity of assets bought or sold, displayed in base currency.

Analyse The Data

- the head() method is used to return the top n (5 by default) rows of a Data.
- List the Last five-row of the dataset using the tail function.
- describe() method computes a summary of statistics like count, mean, standard deviation, min, max, and quartile values.

- info() method prints information about the DataFrame. The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).

Handling Missing Values, Reset The Index & Renaming The Column

1. After loading it is important to check the complete information of data as it can indicate many of the hidden information such as null values in a column or a row

2. Check whether any null values are there or not. if it is present then the following can be done,

a.Imputing data using the Imputation method in sklearn

b.Filling NaN values with mean, median, and mode using fillna() method.

3. isnull()- Generate boolean mask indicating missing values.

4. We don't have any missing values present in our dataframe.

- Check the total no of missing values presented in the dataset

Visualize Time Series Plot

- Now, let’s visualize the data using the Plotly library for Time Series plot of Bitcoin Open Price.

Fitting The Prophet Library

- Create the instance of the prophet and fit it to the dataset.

By default Prophet fits additive seasonalities, meaning the effect of the seasonality is added to the trend to get the forecast. This time series of the price of Bitcoin where additive seasonality does not work. This time series has a clear yearly cycle, but the seasonality in the forecast is too large at the start of the time series and too small at the end. In this time series, the seasonality is not a constant additive factor as assumed by Prophet, rather it grows with the trend. This is multiplicative seasonality.

the prophet can model multiplicative seasonality by setting seasonality_mode='multiplicative' in the input arguments:

Making Future Predictions

The next step is to prepare our model to make future predictions. This is achieved using the Prophet.make_future_dataframe method and passing the number of days we’d like to predict in the future. We use the periods attribute to specify this. This also includes the historical dates. We’ll use these historical dates to compare the predictions with the actual values in the ds column.

Evaluate The Model

We use the predict method to make future predictions. This will generate a dataframe with an yhat column that will contain the predictions.

If we check the head for our forecast dataframe we'll notice that it has very many columns. However, we are mainly interested in ds, yhat, yhat_lower and yhat_upper. yhat is our predicted forecast, yhat_lower is the lower bound for our predictions and yhat_upper is the upper bound for our predictions.

This is the final activity of this milestone, here you will be saving the model to integrate to the web application.

Save The Model

If the dataset in same directory ofyour program, you can directly read it, without any path. After the next Steps we made following bellow:

1.Datavisualization

2.Collabrative and filtering

3.Creating the Model

4.Test and save the model

5.Build Python Code

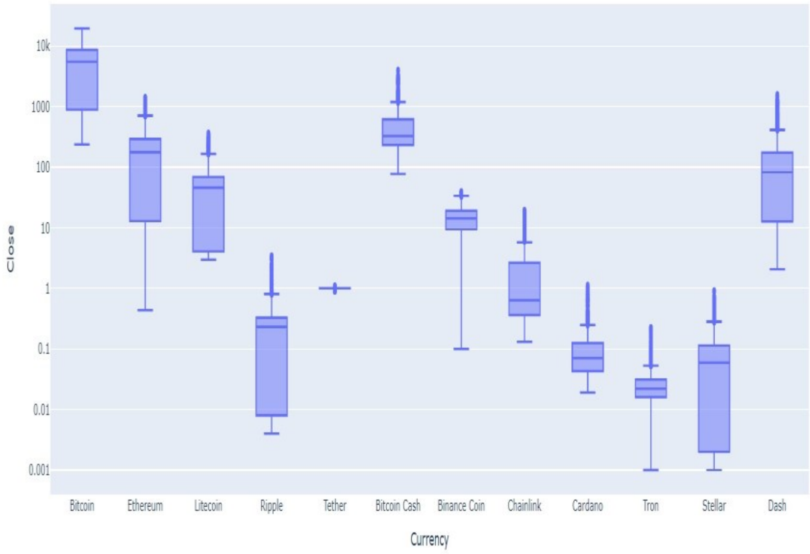
6.Build HTML Code

7.Run the Application

8.We are the following above sections we did and investigate it.

6.PROPOSED WORK

In the system that we have implemented, we have taken the data from January, 2012. This data is within 1 minute intervals. Four types of ML models have been used: ARIMA, Facebook prophet, XG boost and LSTM.



Comparison of various Cryptos

At first we compared various cryptos using a dataset that consisted of various coins like Bitcoin, Dash, Etherium, Cardano, etc which were mapped with each other alongside columns such as Date, Open, High, Close, Adj Close, Volume, etc. We then plotted various pyplots, box plots, pie charts, violin plots in order to analyze all the currencies and figured out that bitcoin would be ideal to work upon because of the volume that it is traded in and its stability.

Methodology:

Data Collection:

In this study, we are focusing on the time-series forecast of BTC prices using machine learning. A time-series is a set of data values with respect to successive moments in time. Time-series forecast is the forecast of future behavior by analyzing time-series data. First we collect all the Data from the "Bitcoin Historical Data" dataset which is available on Kaggle. Included here is historical bitcoin market data at 1-min intervals for select bitcoin exchanges where trading takes place. It consists of a time period of Jan 2012 to September 2020, with minute to minute updates of OHLC (Open, High, Low, Close), Volume in BTC and indicated currency, and weighted bitcoin price.The Open and Close columns indicate the opening and closing price on a particular day. The High and Low columns provide the highest and the lowest price on a particular day, respectively. The Volume column tells us the total volume of traded on a particular day.The Weighted price is a trading benchmark used by traders that gives the weighted price a security has traded at throughout the day, based on both volume and price. It is important because it provides traders with insight into both the trend and value of a security.

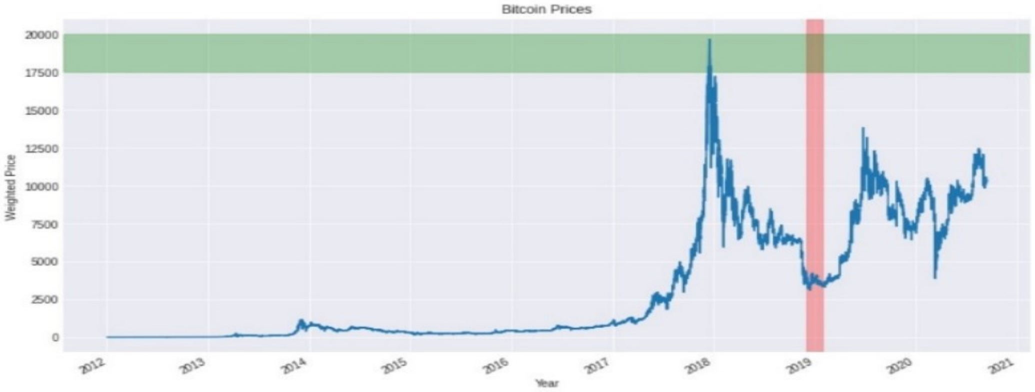
Missing Values:

We replaced the missing values by using three imputation techniques which are 'ffill' or 'pad', 'bfill' or 'backfill' and the linear interpolation method. In the 'ffill' method, the NaNs are replaced with the observed value. In the 'bfill' method, the NaNs are replaced with the next observed value. By using these two methods, a fair portion of missing values are filled. In order to fill the remaining values, we use the linear interpolation method. It is an imputation technique that assumes a linear relationship between data points and utilises non-missing values from adjacent data points to compute a value for a missing data point. Thus, using these three methods, we observe no null values in our dataset.

Exploratory Data Analysis:

1.Visualizing the weighted price using markers:

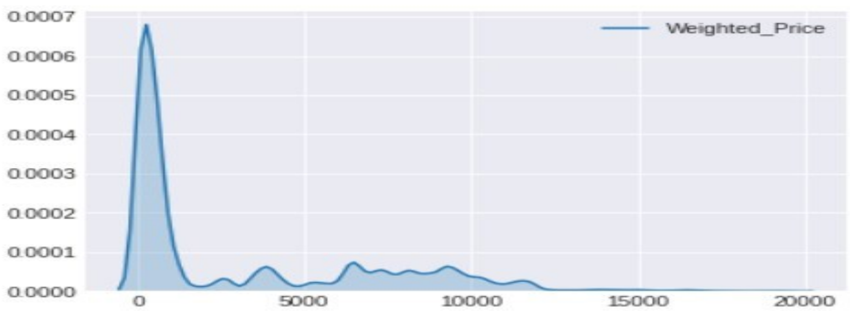
When working with time-series data, a lot can be revealed through visualizing it. It is possible to add markers in the plot to help emphasize the specific observations or specific events in the time series.



Weighted prices v years

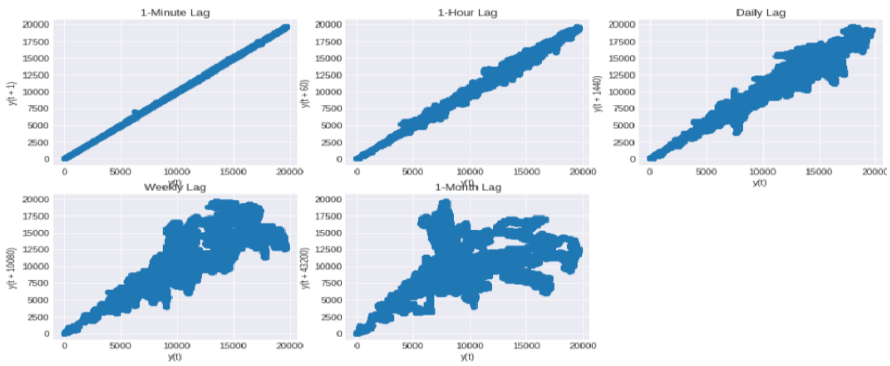
2.Visualizing using KEDs:

Summarizing the data with Density plots to see where the mass of the data is located.



3.Visualizing using Lag Plots:

Lag plots are used to observe the autocorrelation. These are crucial when we try to correct the trend and stationarity and we have to use smoothing functions. Lag plot helps us to understand the data better.



4.Time resampling:

As we noticed the correlation in the above diagram, we decided to resample the data in an Hourly format. Thus we replaced the price of the particular hour by the mean of all the minute prices in that hour.

5.Time Series Decomposition:

We can decompose a time series into trend, seasonal and remainder components. The series can be decomposed as an additive or multiplicative combination of the base level, trend, seasonal index and the residual.Then,we performed some statistical tests like KPSS and Augmented Dickey-Fuller tests to check stationarity.

Time series Decomposition

Post time series decomposition we don't observe any seasonality. Also, there is no constant mean, variance and covariance, hence the series is Non Stationary.

6.KPSS test:

The KPSS test, short for Kwiatkowski-Phillips-Schmidt-Shin (KPSS), is a type of Unit root test that tests for the stationarity of a given series around a deterministic trend.

Time series data can be noisy due to high fluctuations in the market. As a result, it becomes difficult to gauge a trend or pattern in the data.As we're looking at daily data, there's quite a bit of noise present. It would be nice if we could average this out by a week, which is where a rolling mean comes in.A rolling mean, or moving average, is a transformation method which helps average out noise from data. It works by simply splitting and aggregating the data into windows according to function, such as mean(), median(), count(), etc. For this example, we'll use a rolling mean for 3, 7 and 30 days.Our data becomes a lot less noisy and more reflective of the trend than the data itself.

Model building:

To measure the performance of our forecasting model, We typically want to split the time series into a training period and a validation period. This is called fixed partitioning.

We'll train our model during the training period, we'll evaluate it during the validation period. Here's where you can experiment to find the right architecture for training. And work on it and your hyper parameters, until you get the desired performance, measured using the validation set. Often, once you've done that, you can retrain using both the training and validation data.And then test during the test(or forecast) period to see if your model will perform just as well.

And if it does, then you could take the unusual step of retraining again, using also the test data. The test data is the closest data you have to the current point in time. And as such it's often the strongest signal in determining future values. If your model is not trained using that data, too, then it may not be optimal.

1. ARIMA model:

An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that uses time series data to either better understand the data set or to predict future trends[7].ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a class of models that captures a suite of different standard temporal structures in time series data.This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are:

- AR: Autoregression A model that uses the dependent relationship between an observation and some number of lagged observations.
- I: Integrated The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
- MA: Moving Average A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

2. Facebook Prophet:

Facebook Prophet is an open-source algorithm for generating time-series models that uses a few old ideas with some new twists. It is particularly good at modeling time series that have multiple seasonalities.[8]Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects.

It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well

.

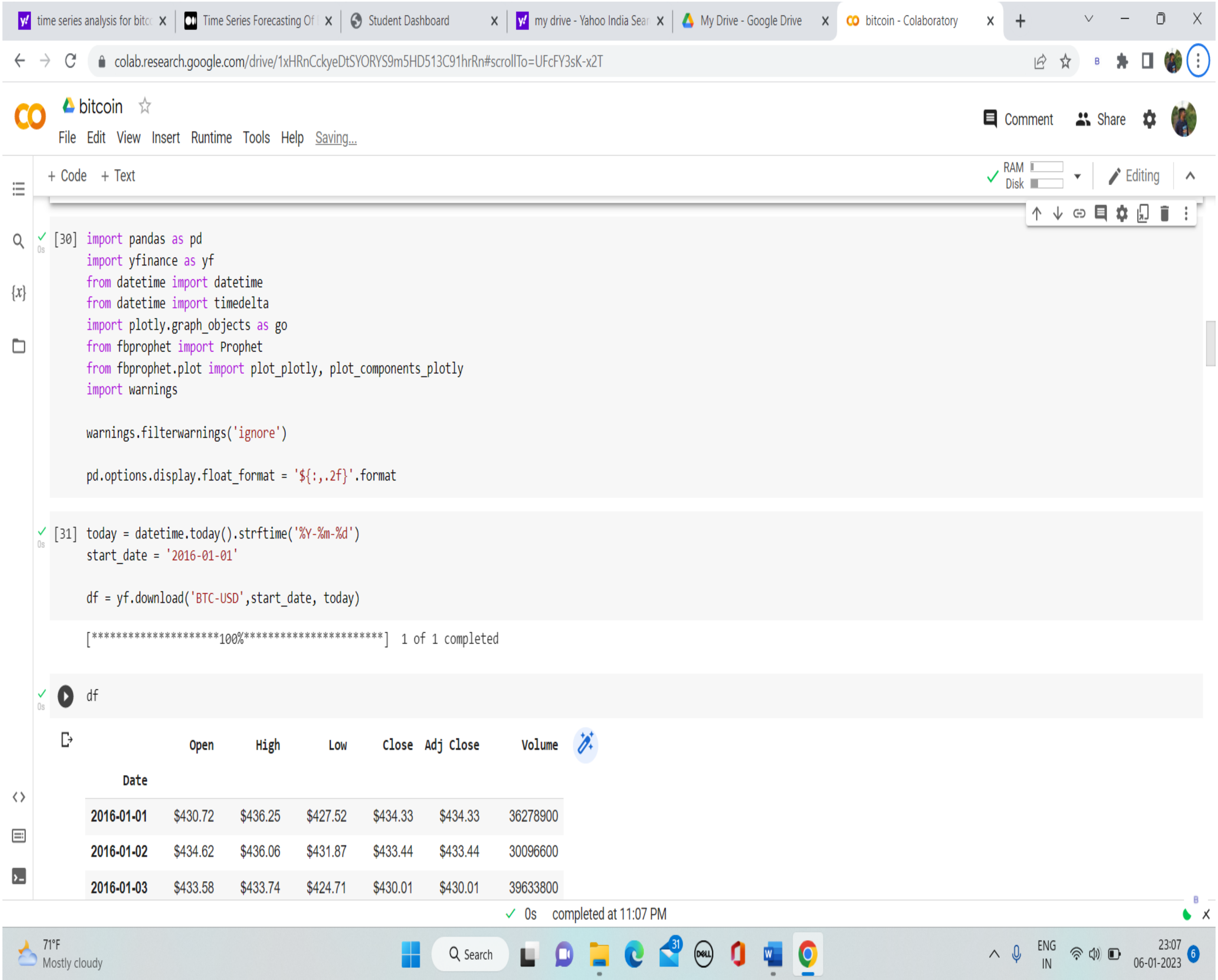
3. XG Boost:

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks.

4. LSTM:

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) 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 (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition and anomaly detection in network traffic or IDSs .

7. OUTPUTS



```
[30] import pandas as pd
     import yfinance as yf
     from datetime import datetime
     from datetime import timedelta
     import plotly.graph_objects as go
     from fbprophet import Prophet
     from fbprophet.plot import plot_plotly, plot_components_plotly
     import warnings

     warnings.filterwarnings('ignore')

     pd.options.display.float_format = '${:,.2f}'.format
```

```
[31] today = datetime.today().strftime('%Y-%m-%d')
     start_date = '2016-01-01'

     df = yf.download('BTC-USD',start_date, today)
```

```
[*********************100%***********************]  1 of 1 completed
```

```
df
```

Date	Open	High	Low	Close	Adj Close	Volume
2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600
2016-01-03	$433.58	$433.74	$424.71	$430.01	$430.01	39633800

```
[33] df.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600
2016-01-03	$433.58	$433.74	$424.71	$430.01	$430.01	39633800
2016-01-04	$430.06	$434.52	$429.08	$433.09	$433.09	38477500
2016-01-05	$433.07	$434.18	$429.68	$431.96	$431.96	34522600

```
df.tail()
```

Date	Open	High	Low	Close	Adj Close	Volume
2022-12-31	$16,603.67	$16,628.99	$16,517.52	$16,547.50	$16,547.50	11239186456
2023-01-01	$16,547.91	$16,630.44	$16,521.23	$16,625.08	$16,625.08	9244361700
2023-01-02	$16,625.51	$16,759.34	$16,572.23	$16,688.47	$16,688.47	12097775227
2023-01-03	$16,688.85	$16,760.45	$16,622.37	$16,679.86	$16,679.86	13903079207
2023-01-04	$16,680.21	$16,964.59	$16,667.76	$16,863.24	$16,863.24	18421743322

```
[35] df.describe()
```

0s completed at 11:07 PM

```
[35] df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	$2,561.00	$2,561.00	$2,561.00	$2,561.00	$2,561.00	$2,561.00
mean	$15,175.19	$15,557.53	$14,747.13	$15,179.83	$15,179.83	$19,339,278,838.71
std	$16,599.61	$17,024.97	$16,105.34	$16,593.25	$16,593.25	$20,344,193,178.66
min	$365.07	$374.95	$354.91	$364.33	$364.33	$28,514,000.00
25%	$3,671.59	$3,758.27	$3,619.95	$3,673.84	$3,673.84	$2,219,589,888.00
50%	$8,667.58	$8,836.84	$8,395.11	$8,668.12	$8,668.12	$15,844,731,575.00
75%	$19,970.47	$20,343.75	$19,523.84	$19,970.56	$19,970.56	$30,966,005,122.00
max	$67,549.73	$68,789.62	$66,382.06	$67,566.83	$67,566.83	$350,967,941,479.00

```
[36] df.info()

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2561 entries, 2016-01-01 to 2023-01-04
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       2561 non-null   float64
 1   High       2561 non-null   float64
 2   Low        2561 non-null   float64
 3   Close      2561 non-null   float64
 4   Adj Close  2561 non-null   float64
 5   Volume     2561 non-null   int64
dtypes: float64(5), int64(1)
```

0s completed at 11:07 PM

```
[37] df.isnull().any()
```

```
Open         False
High         False
Low          False
Close        False
Adj Close    False
Volume       False
dtype: bool
```

```
[38] df.isnull().sum()
```

```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

```
[39] df.reset_index(inplace=True)
     df.columns
```

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

```
[40] df.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
1	2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600

```
[43] new_names = {"Date": "ds","Open": "y"}
     df1.rename(columns=new_names, inplace=True)
```

```
[44] df1.head()
```

	ds	y
0	2016-01-01	$430.72
1	2016-01-02	$434.62
2	2016-01-03	$433.58
3	2016-01-04	$430.06
4	2016-01-05	$433.07

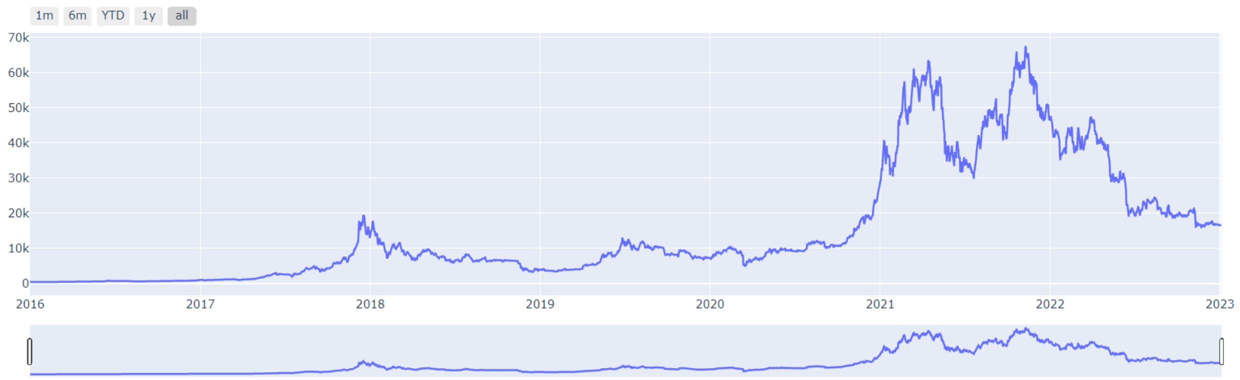
```
[45] x = df1["ds"]
     y = df1["y"]

     fig = go.Figure()

     fig.add_trace(go.Scatter(x=x, y=y))

     fig.update_layout(title_text="Time series plot Bitcoin Open Price",)

     fig.update_layout(xaxis=dict(rangeselector=dict(buttons=list([dict(count=1, label="1m", step="month", stepmode="backward"),
                                                                   dict(count=6, label="6m", step="month", stepmode="backward"),
                                                                   dict(count=1, label="YTD", step="year", stepmode="todate"),
                                                                   dict(count=1, label="1y", step="year", stepmode="backward"),
                                                                   dict(step="all"),
```

bitcoin

File Edit View Insert Runtime Tools Help All changes saved

[45]

Time series plot Bitcoin Open Price




```
[46] m = Prophet(seasonality_mode="multiplicative")
     m.fit(df1)
```

✓ 0s completed at 11:07 PM

bitcoin

File Edit View Insert Runtime Tools Help All changes saved

```
[46] m = Prophet(seasonality_mode="multiplicative")
     m.fit(df1)
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
<fbprophet.forecaster.Prophet at 0x7fd7c6c70760>
```

```
[47] future = m.make_future_dataframe(periods = 365)
     future.tail()
```

	ds
2921	2023-12-31
2922	2024-01-01
2923	2024-01-02
2924	2024-01-03
2925	2024-01-04

```
[48] forecast = m.predict(future)
     forecast
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	multiplicative_terms	multiplicative_terms_lower	multiplicative_terms_upper	weekly	weekly_lower	weekly_uppe
0	2016-01-01	$249.06	$-2,285.10	$3,182.89	$249.06	$249.06	$0.14	$0.14	$0.14	$-0.00	$-0.00	$-0.(
1	2016-01-02	$250.65	$-2,653.79	$3,154.41	$250.65	$250.65	$0.15	$0.15	$0.15	$-0.00	$-0.00	$-0.(

✓ 0s completed at 11:07 PM

[Type here] [Type here] [Type here]

bitcoin

File Edit View Insert Runtime Tools Help All changes saved

2921	2023-12-31	$-16,897.83	$-49,782.76	$11,137.19	$-43,126.36	$9,744.63	$0.14	$0.14	$0.14	$-0.00	$-0.00	$-0.(
2922	2024-01-01	$-16,980.36	$-49,381.37	$10,920.10	$-43,250.42	$9,779.58	$0.15	$0.15	$0.15	$0.00	$0.00	$0.(
2923	2024-01-02	$-17,062.88	$-50,751.27	$11,584.90	$-43,374.48	$9,773.55	$0.15	$0.15	$0.15	$0.00	$0.00	$0.(
2924	2024-01-03	$-17,145.41	$-51,201.54	$11,641.07	$-43,498.54	$9,787.20	$0.15	$0.15	$0.15	$-0.00	$-0.00	$-0.(
2925	2024-01-04	$-17,227.93	$-49,988.41	$10,865.93	$-43,702.06	$9,826.07	$0.15	$0.15	$0.15	$0.00	$0.00	$0.(

2926 rows × 19 columns

```
[49] forecast[['ds','yhat','yhat_lower','yhat_upper']].tail()
```

	ds	yhat	yhat_lower	yhat_upper
2921	2023-12-31	$-19,308.71	$-49,782.76	$11,137.19
2922	2024-01-01	$-19,470.69	$-49,381.37	$10,920.10
2923	2024-01-02	$-19,607.79	$-50,751.27	$11,584.90
2924	2024-01-03	$-19,689.08	$-51,201.54	$11,641.07
2925	2024-01-04	$-19,821.69	$-49,988.41	$10,865.93

0s completed at 11:07 PM

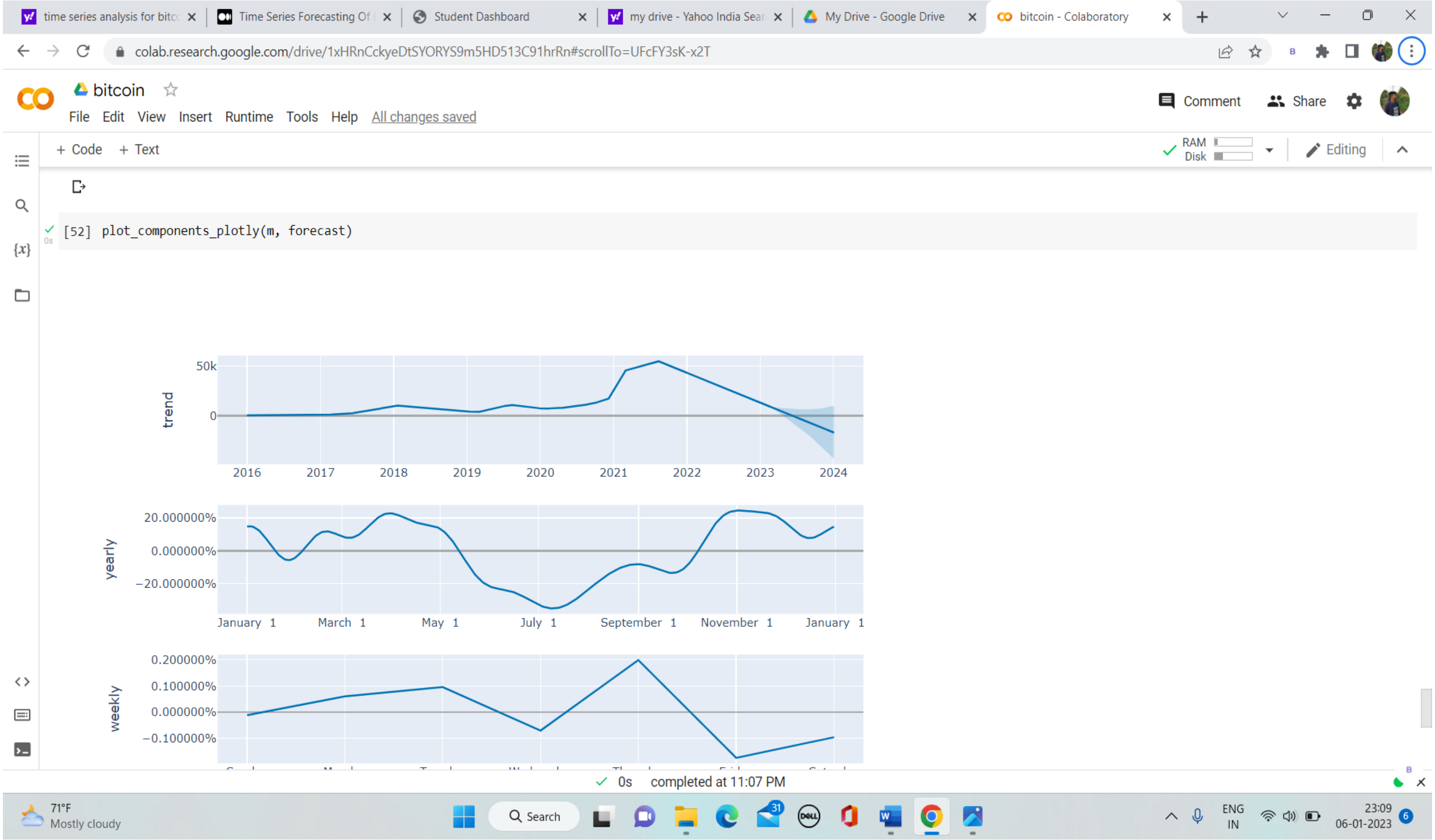
bitcoin

File Edit View Insert Runtime Tools Help All changes saved

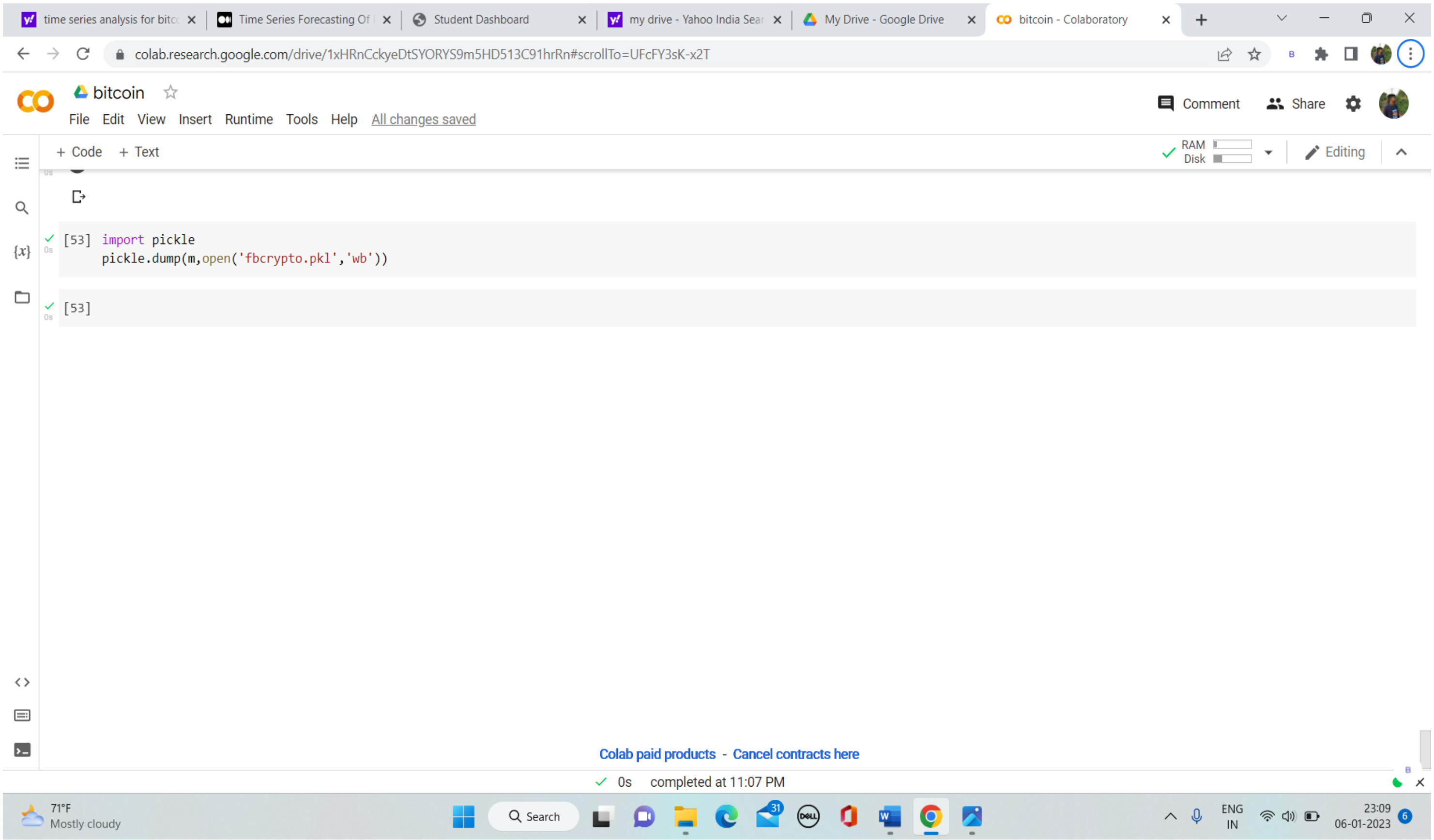
```
[50] next_day = (datetime.today() + timedelta(days=1)).strftime('%Y-%m-%d')

     forecast[forecast['ds'] == next_day]['yhat'].item()

14364.46424244345
```

```
[51] plot_plotly(m, forecast)
```

0s completed at 11:07 PM



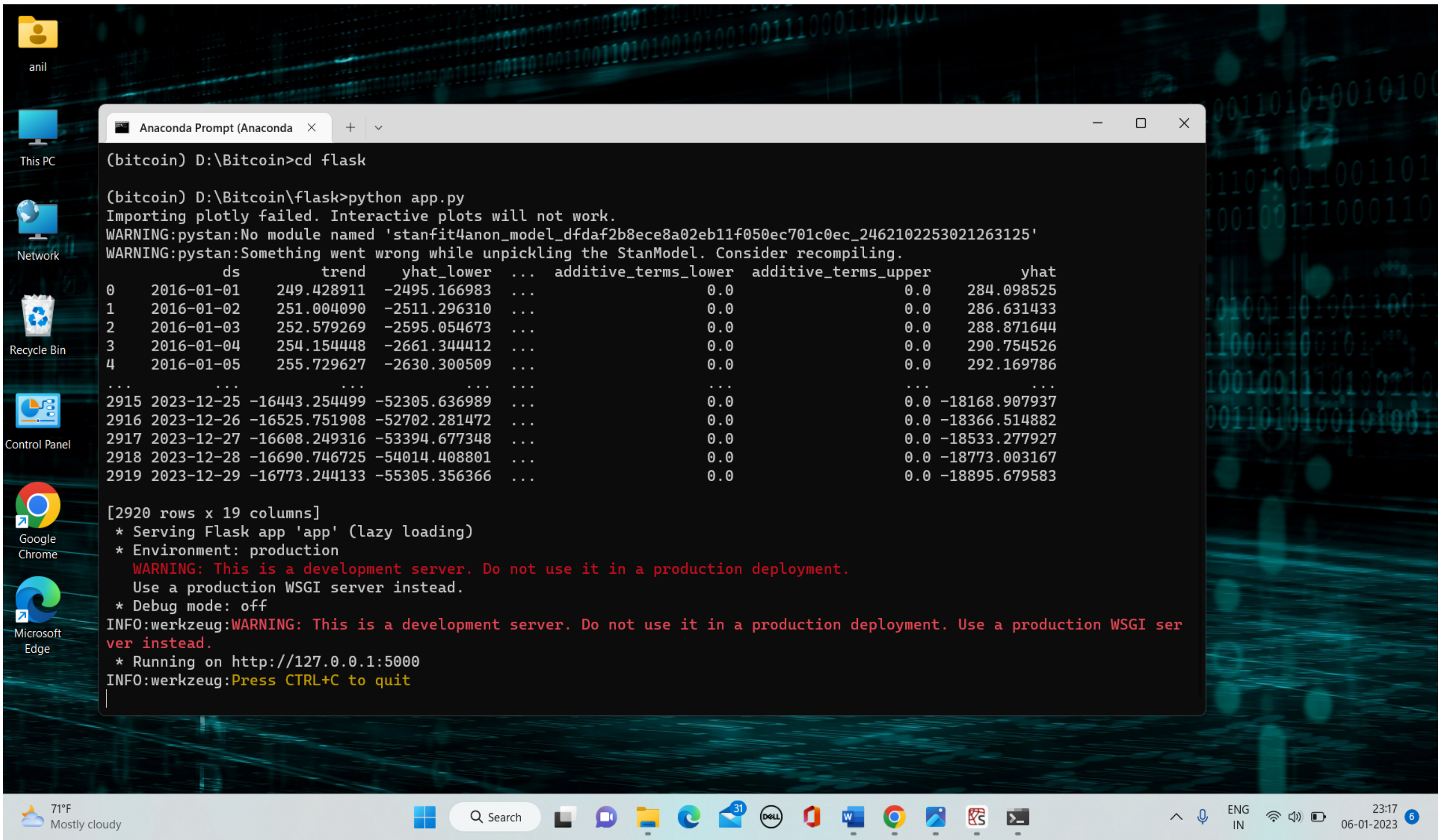
bitcoin
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[52] plot_components_plotly(m, forecast)
trend
50k
0
2016 2017 2018 2019 2020 2021 2022 2023 2024
yearly
20.000000%
0.000000%
−20.000000%
January 1 March 1 May 1 July 1 September 1 November 1 January 1
weekly
0.200000%
0.100000%
0.000000%
−0.100000%
0s completed at 11:07 PM



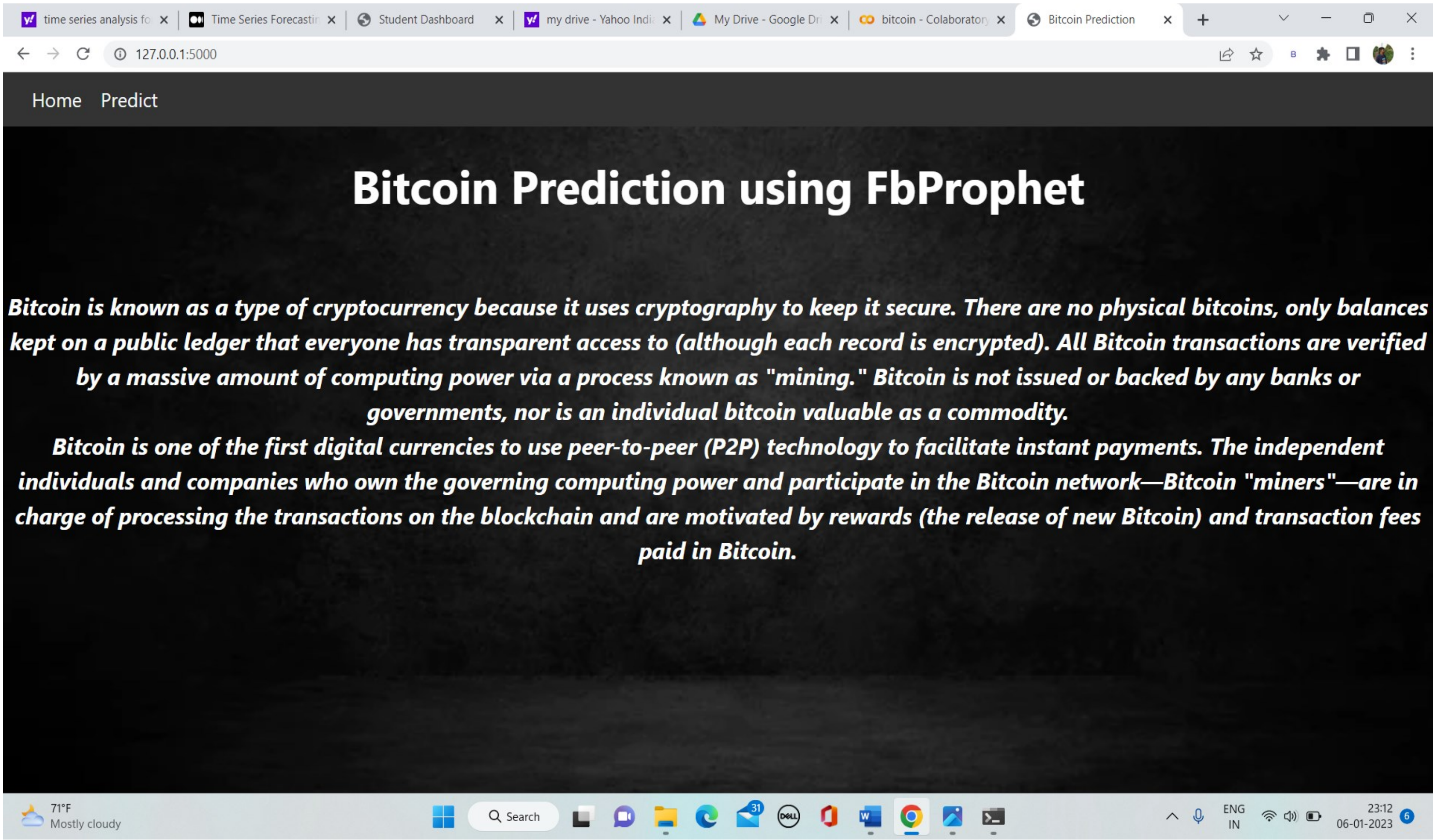
bitcoin
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[53] import pickle
pickle.dump(m,open('fbcrypto.pkl','wb'))
[53]
Colab paid products - Cancel contracts here
0s completed at 11:07 PM

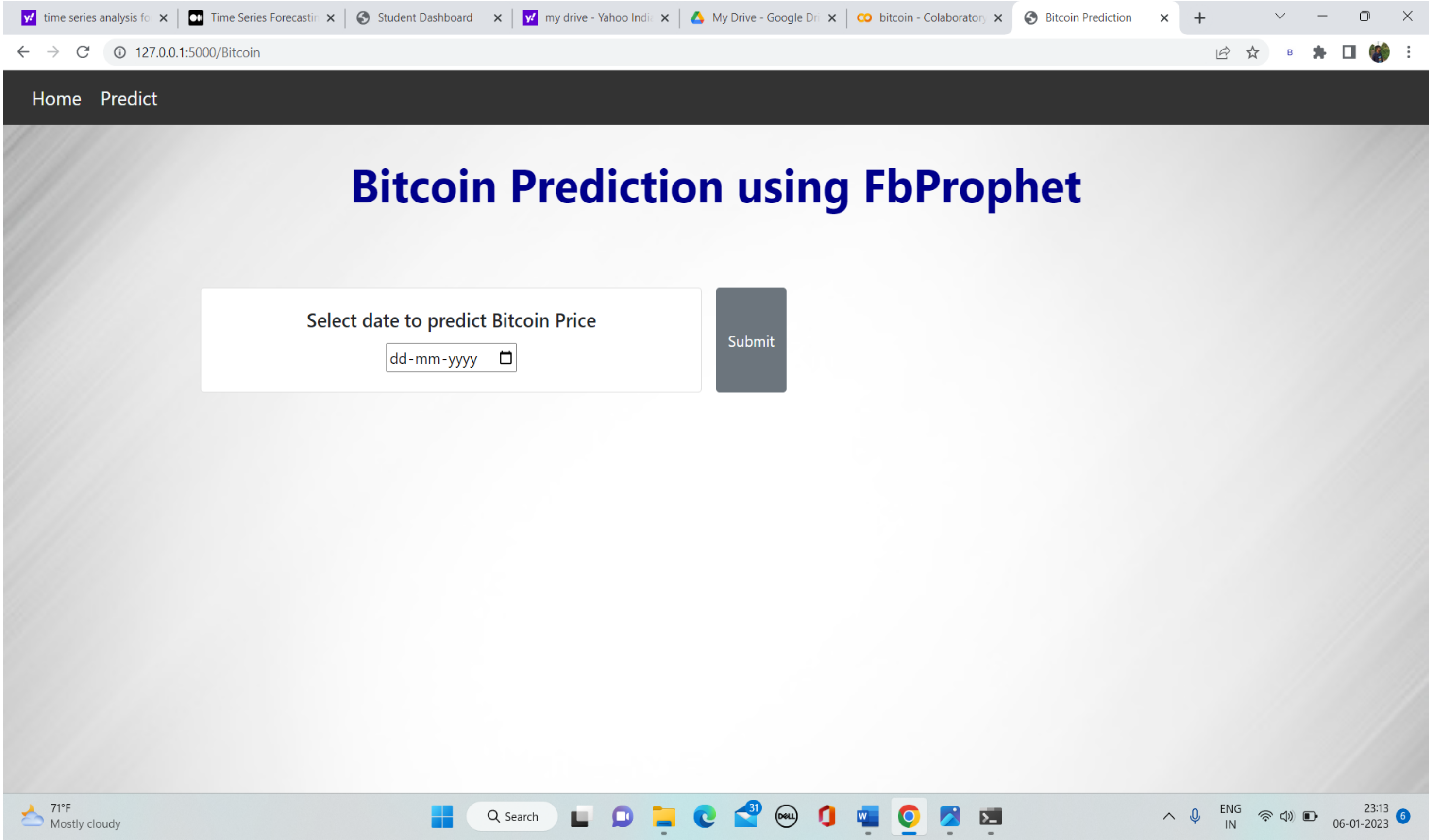
FLASK APP:

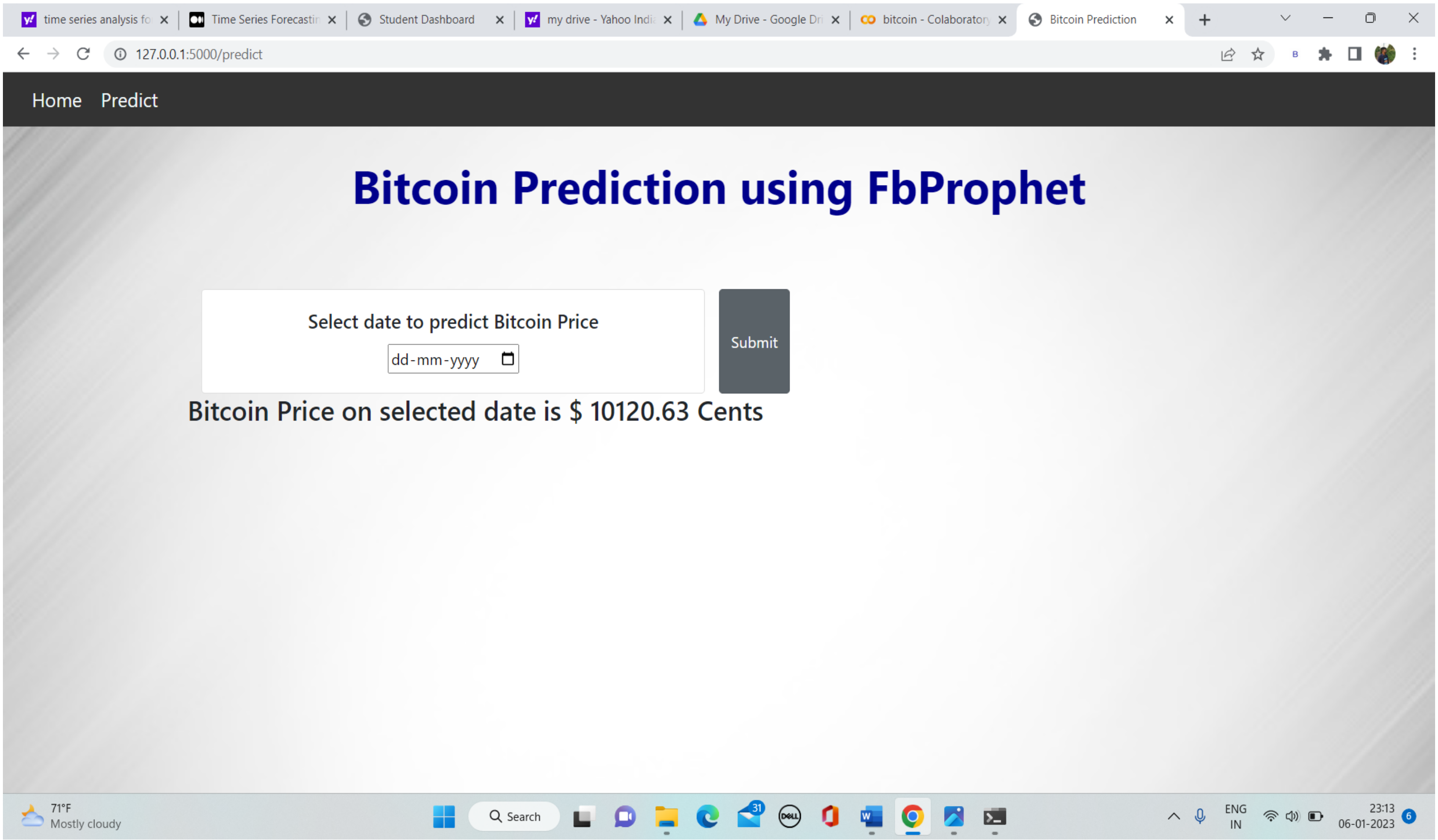
```python
import pandas as pd

from flask import Flask, request, jsonify, render_template
import pickle

#flask app
app = Flask(__name__)
#loading the saved model
m = pickle.load(open('fbcrypto.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/Bitcoin',methods=['POST','GET'])
def prediction(): # route which will take you to the prediction page
    return render_template('predict.html')

future = m.make_future_dataframe(periods = 365)
forecast = m.predict(future)
print(forecast)


@app.route('/predict',methods=['POST'])
def y_predict():
    if request.method == "POST":
        ds = request.form["Date"]
        print(ds)
        ds=str(ds)
        print(ds)
        next_day=ds
        print(next_day)
        prediction=forecast[forecast['ds'] == next_day]['yhat'].item()
        prediction=round(prediction,2)
        print(prediction)
        return render_template('predict.html',prediction_text="Bitcoin Price on selected date is $ {} Cents".format(predict
    return render_template("predict.html")

if __name__ == "__main__":
```


Anaconda Prompt (Anaconda
(bitcoin) D:\Bitcoin>cd flask
(bitcoin) D:\Bitcoin\flask>python app.py
Importing plotly failed. Interactive plots will not work.
WARNING:pystan:No module named 'stanfit4anon_model_dfdaf2b8ece8a02eb11f050ec701c0ec_2462102253021263125'
WARNING:pystan:Something went wrong while unpickling the StanModel. Consider recompiling.
ds trend yhat_lower ... additive_terms_lower additive_terms_upper yhat
0 2016-01-01 249.428911 -2495.166983 ... 0.0 0.0 284.098525
1 2016-01-02 251.004090 -2511.296310 ... 0.0 0.0 286.631433
2 2016-01-03 252.579269 -2595.054673 ... 0.0 0.0 288.871644
3 2016-01-04 254.154448 -2661.344412 ... 0.0 0.0 290.754526
4 2016-01-05 255.729627 -2630.300509 ... 0.0 0.0 292.169786
... ...
2915 2023-12-25 -16443.254499 -52305.636989 ... 0.0 0.0 -18168.907937
2916 2023-12-26 -16525.751908 -52702.281472 ... 0.0 0.0 -18366.514882
2917 2023-12-27 -16608.249316 -53394.677348 ... 0.0 0.0 -18533.277927
2918 2023-12-28 -16690.746725 -54014.408801 ... 0.0 0.0 -18773.003167
2919 2023-12-29 -16773.244133 -55305.356366 ... 0.0 0.0 -18895.679583
[2920 rows x 19 columns]
* Serving Flask app 'app' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
71°F
Mostly cloudy
23:17
06-01-2023

HTML:


time series analysis fo
Time Series Forecastir
Student Dashboard
my drive - Yahoo Indi
My Drive - Google Dr
bitcoin - Colaborator
Bitcoin Prediction
127.0.0.1:5000
Home Predict
Bitcoin Prediction using FbProphet
Bitcoin is known as a type of cryptocurrency because it uses cryptography to keep it secure. There are no physical bitcoins, only balances kept on a public ledger that everyone has transparent access to (although each record is encrypted). All Bitcoin transactions are verified by a massive amount of computing power via a process known as "mining." Bitcoin is not issued or backed by any banks or governments, nor is an individual bitcoin valuable as a commodity.
Bitcoin is one of the first digital currencies to use peer-to-peer (P2P) technology to facilitate instant payments. The independent individuals and companies who own the governing computing power and participate in the Bitcoin network—Bitcoin "miners"—are in charge of processing the transactions on the blockchain and are motivated by rewards (the release of new Bitcoin) and transaction fees paid in Bitcoin.
71°F
Mostly cloudy
23:12
06-01-2023


time series analysis fo
Time Series Forecastir
Student Dashboard
my drive - Yahoo Indi
My Drive - Google Dr
bitcoin - Colaborator
Bitcoin Prediction
127.0.0.1:5000/Bitcoin
Home
Predict
Bitcoin Prediction using FbProphet
Select date to predict Bitcoin Price
dd-mm-yyyy
Submit
71°F
Mostly cloudy
Search
ENG
IN
23:13
06-01-2023


time series analysis fo
Time Series Forecastir
Student Dashboard
my drive - Yahoo Indi
My Drive - Google Dr
bitcoin - Colaborator
Bitcoin Prediction
127.0.0.1:5000/predict
Home
Predict
Bitcoin Prediction using FbProphet
Select date to predict Bitcoin Price
dd-mm-yyyy
Submit
Bitcoin Price on selected date is $ 10120.63 Cents
71°F
Mostly cloudy
Search
ENG
IN
23:13
06-01-2023

8. CONCLUSION & FUTURE WORK

We observed remarkable results using LSTMs. They really work a lot better for most sequence tasks. Also the other model's performance was not as good as LSTM which is evident from the RMSE score.

We can increase the number of epochs to refine our model performance, we can increase epochs to 100 and see the results. Also, the number of lag features can be increased beyond 100 to help in model learning.

9. REFERENCES

[1] Bitcoin Price Prediction Based on Deep Learning Methods Xiangxi Jiang Published: February 11, 2020

[2] Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach Mohammed Mudassir, Shada Bennbaia, Devrim Unal &Mohammad Hammoudeh Published: 04 July 2020

[3] Bitcoin Price Prediction using Machine Learning Siddhi Velankar, Sakshi Valecha, Shreya Maji Published: 11 February 2018

[4] Predicting Fluctuations in Cryptocurrency Transactions Based on User Comments and Replies Young Bin Kim,Jun Gi Kim,Wook Kim,Jae Ho Im,Tae Hyeong Kim,Shin Jin Kang,Chang Hun Kim Published: August 17, 2016

[5] Predicting Bitcoin Returns Using High-Dimensional Technical Indicators Jing-Zhi Huang, William Huang, Jun Ni Received Date: 12 October 2018Accepted Date: 29 October 2018

[6] Next-Day Bitcoin Price Forecast Ziaul Haque Munim ,Mohammad Hassan Shakil and Ilan Alon Published: 20 June 2019

[7] Investopedia/ARIMA

[8] Towards data science

SOURCE CODE(FLASK):

```
import numpy as np
import pandas as pd
from flask import Flask, request, jsonify, render_template
import pickle

#flask app
app = Flask(__name__)
#loading the saved model
m = pickle.load(open('D:/Bitcoin/fbcrypto.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/Bitcoin',methods=['POST','GET'])
def prediction(): # route which will take you to the prediction page
    return render_template('predict.html')

future = m.make_future_dataframe(periods = 365)
forecast = m.predict(future)
print(forecast)
@app.route('/predict',methods=['POST'])
def y_predict():
    if request.method == "POST":
        ds = request.form["Date"]
        print(ds)
        ds=str(ds)
        print(ds)
        next_day=ds
        print(next_day)
        prediction=forecast[forecast['ds'] == next_day]['yhat'].item()
        prediction=round(prediction,2)
        print(prediction)
        return render_template('predict.html',prediction_text="Bitcoin Price on selected date is $ {}
Cents".format(prediction))
    return render_template("predict.html")

if __name__ == "__main__":
    app.run(debug=False)
```