Crypto price prediction using FbProphet.

Project Report

Source Code
GitHub & Project Demo Link

1. Project Introduction

The core idea of this project is to be able to predict the price of Bitcoin which is the most valuable cryptocurrency in the world with reasonable accuracy, using time series forecasting libraries like FbProbhet.

This will not only enable us to take trades on the basis of future price but on a larger scale if the model is accurate enough it can also be used to make crypto market more predictable and hence more acceptable in general public who usually wants to avoid high risk environments.

Bitcoin prediction can be useful in various ways, offering valuable insights to investors, traders, and the broader cryptocurrency community. By analyzing historical data and market trends, these predictions can help individuals make informed investment decisions, manage risk, and optimize their portfolios. They can also serve as a foundation for trading strategies and automated algorithms, enabling users to capitalize on price movements.

Additionally, Bitcoin predictions can foster awareness, education, and market sentiment analysis, contributing to a better understanding of the cryptocurrency market's dynamics and its potential for both gains and losses. However, it's essential to remember that cryptocurrency markets are highly volatile, and predictions are never fool proof, so they should be used in conjunction with other research and risk management strategies.

2. LITERATURE SURVEY

Time Series Forecasting in Finance: Explore research papers and articles on the application of time series forecasting in financial markets. Understand different methods and algorithms used for predicting financial time series data.

Cryptocurrency Price Prediction: Look into studies specifically focused on predicting cryptocurrency prices. Analyse the challenges and methodologies employed in predicting the volatile nature of crypto assets.

FbProphet and Time Series Forecasting: Investigate academic papers or documentation related to FbProphet to understand its underlying principles. Examine case studies where FbProphet has been successfully applied in various domains.

Web-based Machine Learning Model Deployment: Explore literature on deploying machine learning models on the web, especially using Flask. Understand the best practices, challenges, and security considerations in deploying predictive models online.

User Acceptance of Predictive Models: Search for literature discussing user perceptions and acceptance of predictive models, especially in the context of financial markets. Consider ethical implications and transparency in deploying predictive models to a wider audience.

Integration of Data Sources (Yahoo Finance): Investigate how financial data from sources like Yahoo Finance is used in academic and industry research. Understand the challenges and best practices in collecting and pre-processing financial time series data.

Evaluation Metrics for Predictive Models: Explore literature on metrics used to evaluate the performance of time series forecasting models. Understand the limitations of common metrics and any proposed alternatives.

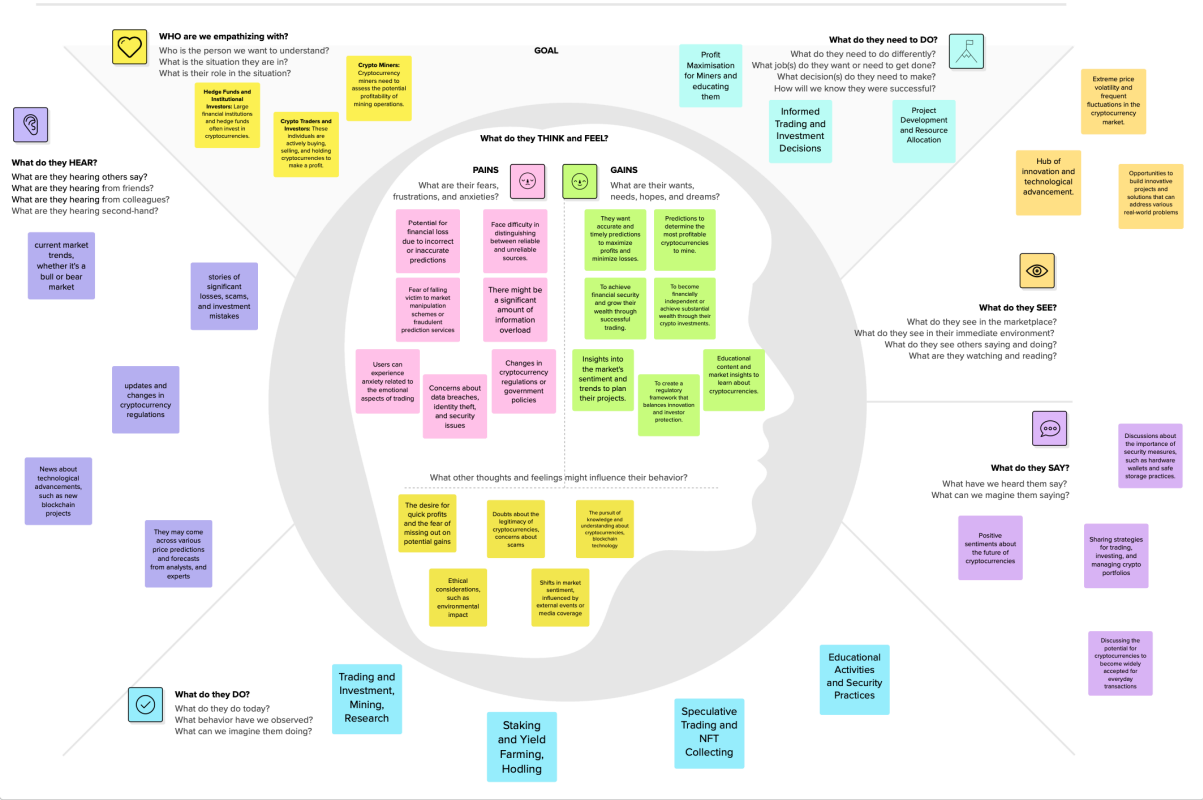
Impact of Predictive Models on Financial Markets: Research articles discussing the potential impact of accurate predictive models on financial markets. Consider both positive and negative implications and the overall market dynamics.

Machine Learning Model Interpretability: Look into research on making machine learning models more interpretable, especially in the context of financial predictions. Understand the importance of model interpretability for user trust and acceptance.

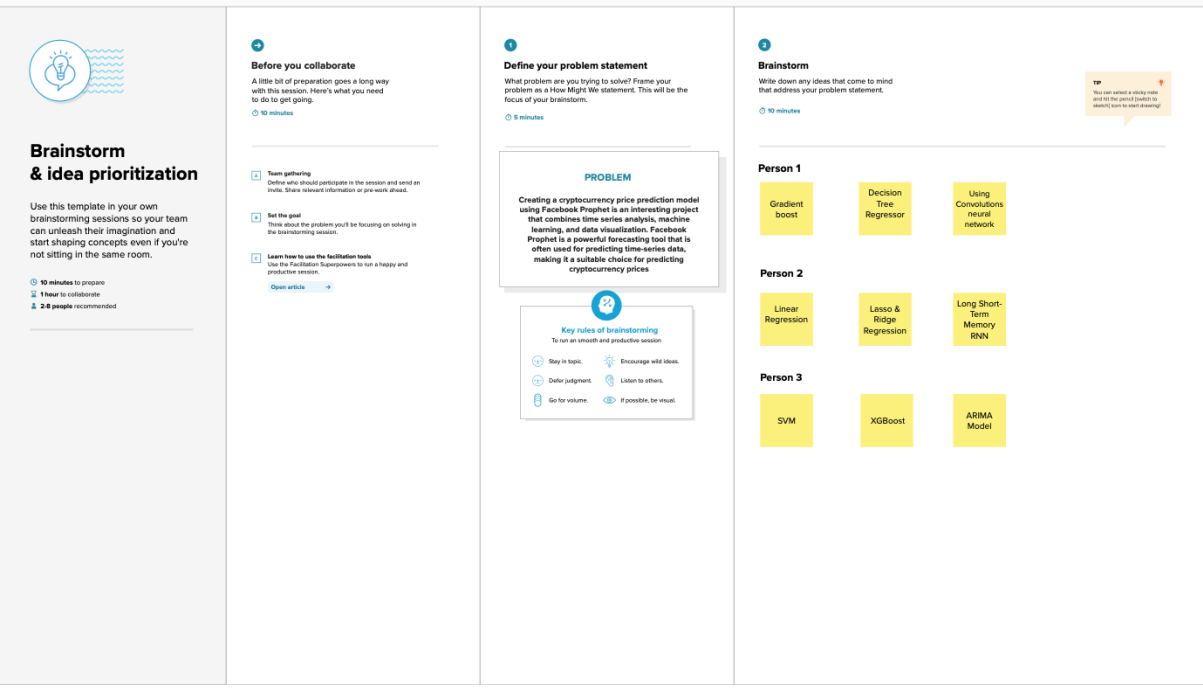
Future Trends in Crypto and Finance Prediction: Explore recent publications discussing emerging trends and advancements in the field of cryptocurrency prediction and financial forecasting.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map



3.2 Brain Storming



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Group 1

- Data Collection
- see the outliers and remove them
- visualise the data
- Organize your data, ensuring you have a timestamp
- Install and import the necessary Python libraries

Group 2

- preprocess data and see the skewness of each parameter
- handling missing values using mean, etc.
- split the data into test and train data sets
- min-max scaler can be used for scaling

Group 3

- make a 3 layer network
- Utilize Facebook Prophet to create a time series forecasting model
- Calculate metrics to evaluate the performance of your model
- you can deploy it as a web application

4

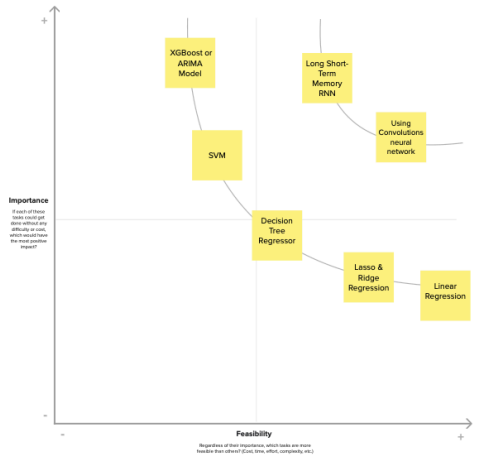
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

Share the mural
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

Export the mural
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

4. Requirement Analysis

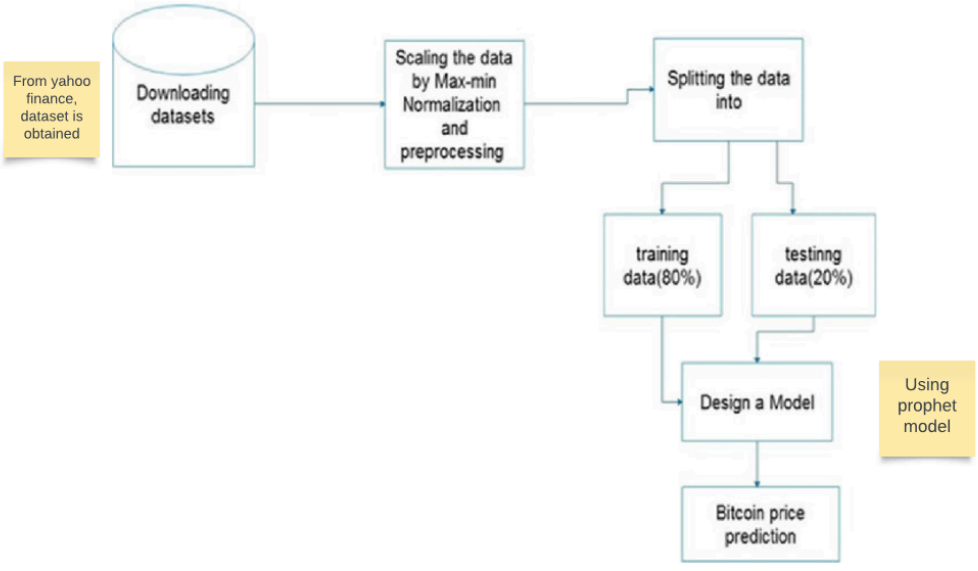
Table-1 : Components & Technologies:

Component	Description	Technology
User Interface	How user interacts with application	HTML, CSS, Flask
Application Logic-1	Predicting the results in the background at cloud or local system and letting user interact with it on the web	Flask frame work
Cloud Database	Database Service on Cloud	Flask
File Storage	File storage requirements	Local file
External API-1	The purpose of API is to fetch accurate data of bitcoin for our model to train and make predictions.	yfinance
Machine Learning Model	The model is built to be able to predict bitcoin values using available data	Prophet
Infrastructure (Server / Cloud)	Local server configuration and to deploy an application to the cloud	Flask, Google

Table-2: Application Characteristics:

Characteristics	Description	Technology
Open-Source Frameworks	The framework should be open-source, which means that the underlying code and algorithms are freely available to the public. This encourages collaboration, transparency, and a wider user base.	Flask
Security Implementations	Cryptocurrency-related projects need to be mindful of security issues, especially when dealing with financial data. The framework should incorporate best practices for data security and user authentication.	https
Scalable Architecture	Cryptocurrency markets can be highly volatile and require real-time analysis. Therefore, the framework should be scalable to handle large volumes of data efficiently.	google cloud
Availability	To ensure the availability of the system, it's crucial to use a reliable and redundant infrastructure.	google cloud
Performance	Continuously fine-tune the Prophet models to adapt to changing market conditions and improve prediction accuracy.	metrics

Technical Architecture:



Proposed Solution Template:

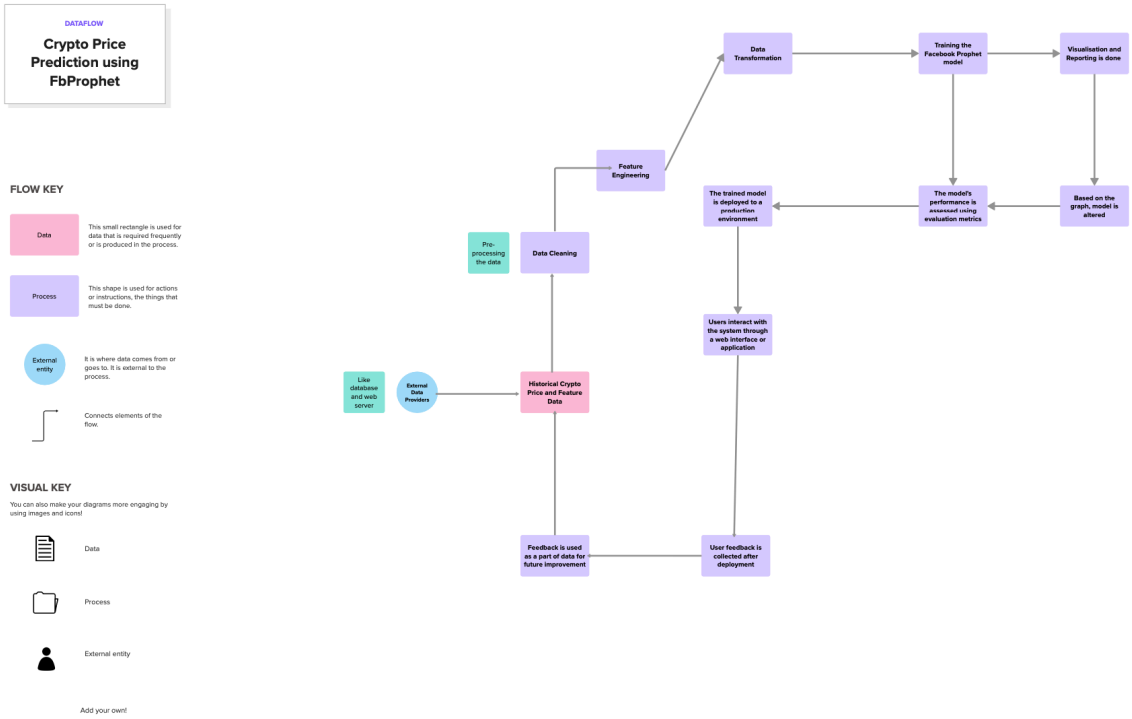
Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	One of the biggest things stopping widespread adoption of Crypto is it's high and unpredictable volatility, when you own a dollar or an rupee you know it is going to be valued same at any time after a month, or a year, there is an expected loss due to inflation of 4-8% but that is acceptable to most of people as long as it stays in that range, but with crypto a dollar can be worth 10X or 0.1X after a month.
2.	Idea / Solution description	The idea is to be able to predict the price of crypto currency specifically bit coins to a certain margin of error, if people know that okay after two months BTC would be worth say 40,000 USD 5-10% more or less, then they would feel much more comfortable to hold an invest in them, and that is exactly our idea, to give people an educated estimate on what will be the future value of the crypto they are holding or are planning to buy.
3.	Novelty / Uniqueness	Though there are many websites who do all sorts of predictions they hardly have interests of retail customers in mind, as they are mostly operated by some crypto exchange, who is not an neutral party in this game, as exchange will always want people to buy crypto, that is how they earn, our product will be consumer centric and therefore will try to provide conservative estimates and predictions won't be rigid by any incentives to sell the same crypto whose price we are trying to predict.
4.	Social Impact / Customer Satisfaction	If we are able to make an model that is able to predict the value of BTC even with inaccuracy of 5-10% then it would encourage widespread adoption of crypto currency and therefore fast forward the inevitable revolution of Web 3.

5.	Business Model (Revenue Model)	There are multiple revenue models here, first and most obvious being that if someone can predict price of BTC with any reasonable accuracy than he/she can leverage that to trade according to those prices and make money out of it, the second revenue model consists of a subscription based model, where general public will have access to normal predictions and will be able to view a bit into future, but the traders who want the most accurate information fastest will have to pay a premium price to get predictions of BTC future into the future.
6.	Scalability of the Solution	Once the model has been made and it is able to make reasonable predictions, it can easily be scaled to unimaginable levels, as all it would require is a cloud to host a website and to a bit of computation power to run the algorithm, and then virtually anyone from any part of the world can simply create an account on website and see what will be the price of the BTC in future according to the algorithm.

5. Project Design

Data flow diagram:

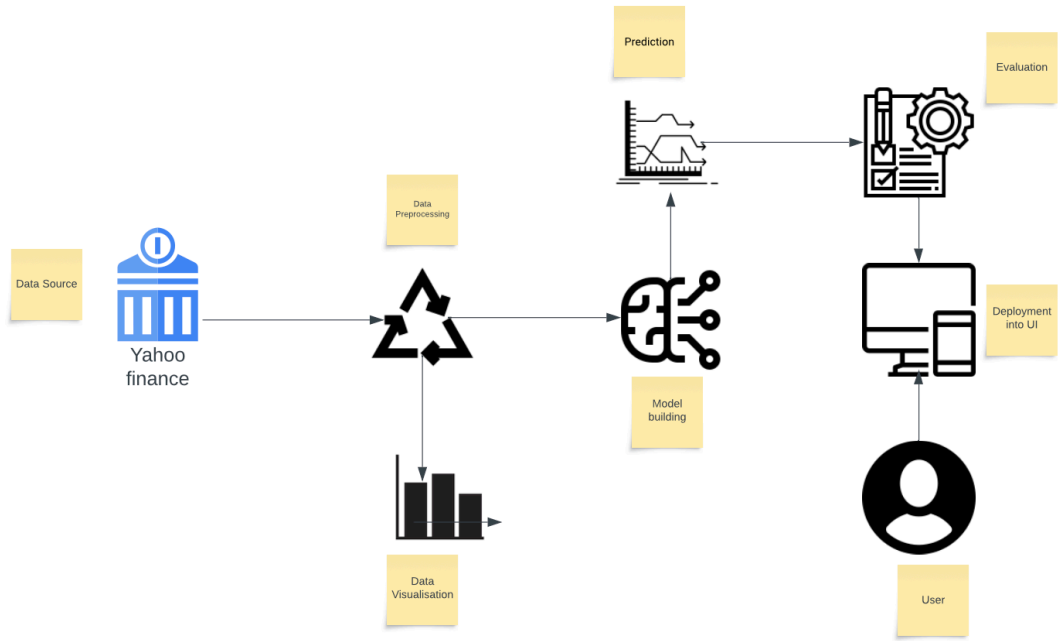


User stories:

User Stories

User Type	Functional Requirement	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Investor	Display daily price predictions for a selected cryptocurrency.	USN-1	As a crypto investor, I want to see a daily price prediction for a specific cryptocurrency, so I can make informed investment decisions.	The system should allow the user to input the cryptocurrency symbol and select a date range. - The system should provide daily price predictions for the selected cryptocurrency within the specified date range. - The predictions should be displayed in an easy-to-read format (e.g., chart or table).	HIGH	SPRINT-1
Trader	Set price threshold alerts. - Send email notifications when the predicted price crosses the set threshold.	USN-2	As a trader, I want to receive email notifications when the predicted price of a cryptocurrency crosses a certain threshold, so I can take action accordingly.	The user should be able to set price threshold alerts. - When the predicted price crosses the set threshold, an email notification should be sent.	HIGH	SPRINT-1
Researcher	Provide historical price data in a downloadable format. - Allow users to specify the date range for historical data.	USN-3	As a researcher, I want to access historical crypto price data for analysis, so I can study market trends.	The system should provide historical price data in a downloadable format (e.g., CSV). - The user should be able to specify the date range for the historical data.	LOW	SPRINT-2
Developer	Implement FbProphet for time series forecasting. - Regularly update the model with new data.	USN-4	As a developer, I want the system to use FbProphet to perform accurate price predictions for cryptocurrencies, so the predictions are reliable and up-to-date.	The system should implement FbProphet for time series forecasting. - The model should be regularly updated with new data.	HIGH	SPRINT-1
General User	Create a user-friendly and intuitive web interface. - Allow users to input parameters and view predictions	USN-5	As a user, I want to have an easy-to-use web interface for accessing crypto price predictions, so I can navigate and use the system without technical expertise.	The web interface should be user-friendly and intuitive. - Users should be able to input parameters easily and view predictions.	MEDIUM	SPRINT-1

Solution Architecture:



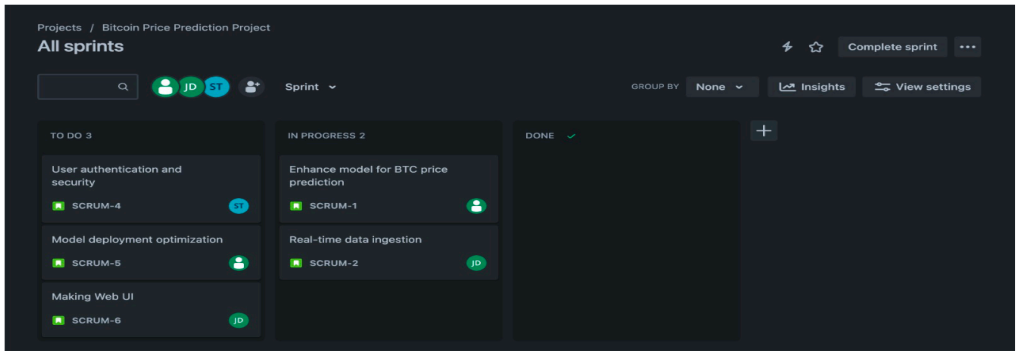
6. Project Planning and Scheduling

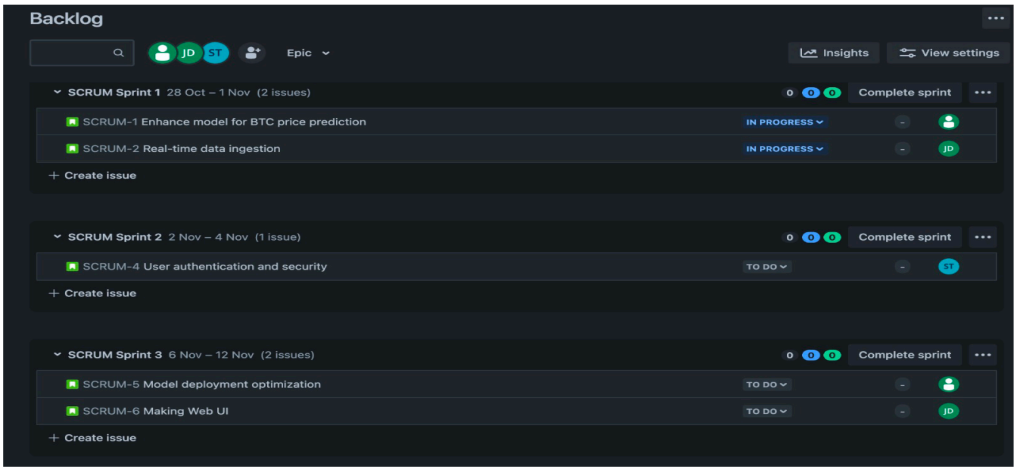
Product Backlog, Sprint Schedule, and Estimation

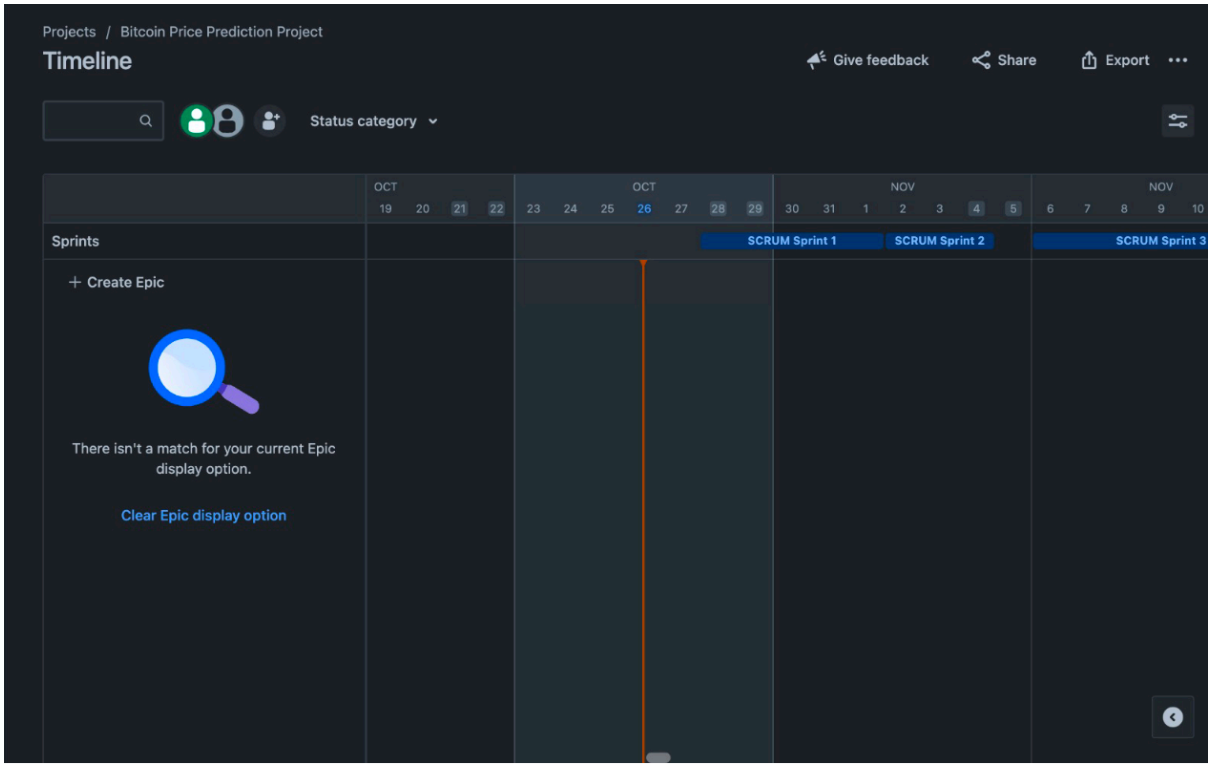
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task Story	Points	Priority
Sprint 1	Prophet	USN-1	Enhance Model for BTC Price Prediction	8	HIGH
Sprint 1		USN-2	Real-time data ingestion	5	HIGH
Sprint 2		USN-3	User authentication and security	5	MEDIUM
Sprint 3	Prophet	USN-4	Model deployment optimization	13	HIGH
Sprint 3	Flask,HTML,CSS	USN-5	Making the UI for web	7	LOW

Project Tracker, Velocity & Burndown Chart:

Sprint	Total story point	Duration	Sprint Start Date	Sprint End Date (Planned)
Sprint-1	13	5 days	28 October 2023	1 November 2023
Sprint-2	5	3 days	2 November 2023	4 November 2023
Sprint-3	20	6 days	6 November 2023	12 November 2023






Projects / Bitcoin Price Prediction Project
Timeline
Give feedback
Share
Export
Status category
OCT
19 20 21 22
OCT
23 24 25 26 27 28 29
NOV
30 31 1 2 3 4 5
NOV
6 7 8 9 10
Sprints
SCRUM Sprint 1
SCRUM Sprint 2
SCRUM Sprint 3
Create Epic
There isn't a match for your current Epic display option.
Clear Epic display option

7. Coding and Solutioning

→Phase one – Setting up the environment

- In this phase we created an separate anaconda environment for the project, in-order to avoid any clashes in requirements, then we installed the FbProphet, Yfinnance, PlotLy, Flask and other libraries using 'pip install' command.

→Phase two – Data collection

- Once the environment was ready, our first task was to collect the historical data

→Phase three – Data pre-processing and data visualization

- In this phase we visualized the time series data and cleaned the data and made it ready for the algorithm

→Phase four – Model building

- Once the data was ready it was time to feed the data to our FbProphet model in order to train it, this is what was done in this phase

→Phase five – Deployment

- After the model was ready we finally deployed the model using HTML,CSS and Flask

Let's now look at these phases one by one in detail

Setting up the environment

→This phase was simple and straight forward, we started by downloading anaconda navigator, then installed the Jupyter notebook.

→Once this was done we began installing required libraries starting from FbProphet, then PlotLy, Flask and Yfinance, we accomplished using pip commands.

Data collection

→The data collected in this project comes directly from Yahoo Finance .
→To collect the data we have used Yfinance library of Python, the data is collected from 01/01/2016 till 02/11/23.

Loading the Dataset

```
[ ]  import yfinance as yf
```

```
[ ]  st_date = "2016-01-01"
     td_date = dt.today().strftime("%Y-%m-%d")

     print("Starting Date:", st_date)
     print("Today's Date :", td_date)
```

```
Starting Date: 2016-01-01
Today's Date : 2023-11-02
```

```
df = yf.download("BTC-USD", st_date, td_date)
df.head()
```

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

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

Data pre-processing and visualization

→This phase included understanding, visualizing and finally cleaning the data for model to use.
→First we checked weather there were any discrepancies.

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2862 entries, 2016-01-01 to 2023-11-01
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       2862 non-null   float64
 1   High       2862 non-null   float64
 2   Low        2862 non-null   float64
 3   Close      2862 non-null   float64
 4   Adj Close  2862 non-null   float64
 5   Volume     2862 non-null   int64
dtypes: float64(5), int64(1)
memory usage: 156.5 KB

[ ] df.shape

(2862, 6)

[ ] df.describe()
```

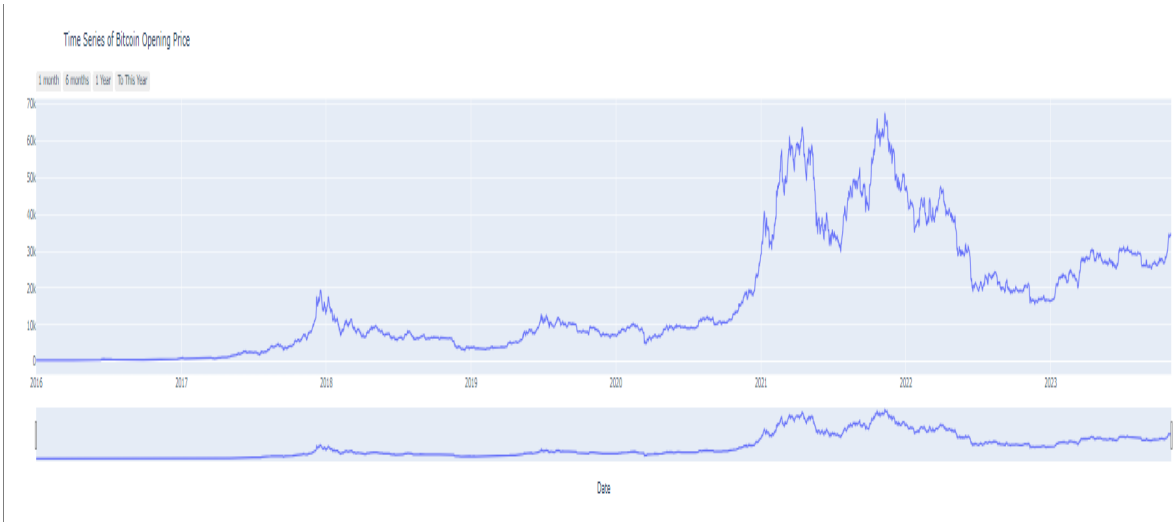
	Open	High	Low	Close	Adj Close	Volume
count	2862.000000	2862.000000	2862.000000	2862.000000	2862.000000	2.862000e+03
mean	16397.794126	16786.357058	15976.882811	16408.447428	16408.447428	1.917040e+10
std	16137.430435	16534.000945	15686.914678	16135.236348	16135.236348	1.945663e+10
min	365.072998	374.950012	354.914001	364.330994	364.330994	2.851400e+07
25%	4034.599426	4110.110962	3968.972778	4039.051270	4039.051270	3.671598e+09
50%	9484.943848	9671.880859	9282.042480	9487.736816	9487.736816	1.564545e+10
75%	26533.612305	26909.723145	26173.731934	26560.643066	26560.643066	2.992057e+10
max	67549.734375	68789.625000	66382.062500	67566.828125	67566.828125	3.509679e+11

```
Checking for NULL Values

[ ] df.isnull().sum()

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

→After checking it was reviled that the there were some discrepancies in indexing of data which we fixed using ‘reset index’ function of ‘pandas’ apart from that there wasn’t anything that could hamper our accuracy of our model, so we started visualizing data to understand it better.

Data Visualisation

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=df["Date"], y=df["Open"]))

fig.update_layout(
    title_text = "Time Series of Bitcoin Opening Price",
    xaxis = dict(

                rangeselector = dict(

                    buttons=list(
                        [
                            dict(count=1, label = "1 month",
                                 step="month", stepmode="backward"),
                            dict(count=6, label = "6 months",
                                 step="month", stepmode = "backward"),
                            dict(count=1, label = "1 Year",
                                step="year", stepmode="backward"),
                            dict(count=1, label = "To This Year",
                                step="year", stepmode="todate")
                        ]
                      )
                ),
                rangeslider = dict(visible = True),
                title = "Date",
    )
)
```



→In the final step of this phase we prepared the data for training and testing, we separated our target variable from rest of the dataset in-order for timeseries prediction to work.

Preparing Training Data

```
[ ] X_train = df.iloc[:,:2]
```

```
[ ] X_train.head()
```

	Date	Open
0	2016-01-01	430.721008
1	2016-01-02	434.622009
2	2016-01-03	433.578003
3	2016-01-04	430.061005
4	2016-01-05	433.069000

Renaming Columns for Model Training Purpose

```
[ ] X_train.columns = ["ds", "y"]
```

```
[ ] X_train.head()
```

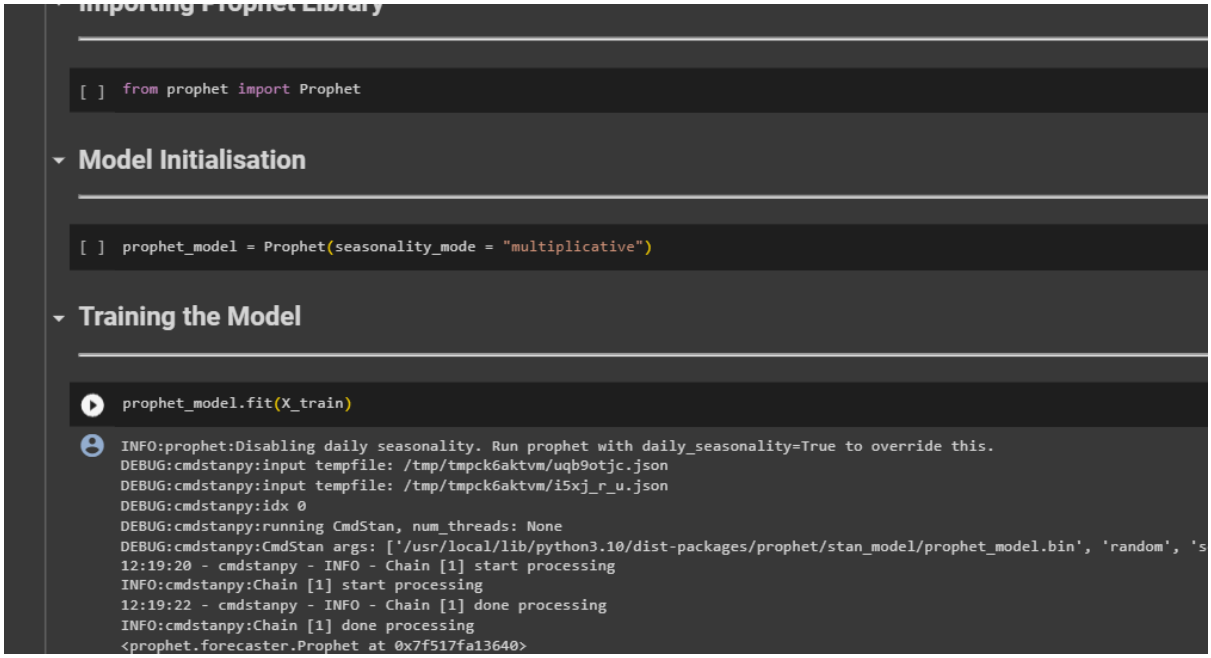
	ds	y
0	2016-01-01	430.721008
1	2016-01-02	434.622009
2	2016-01-03	433.578003
3	2016-01-04	430.061005
4	2016-01-05	433.069000

Model building

→Once we had understood the data, cleaned it and separated target variable, it was time for us to actually build the model

→So first we imported the model, then we created an ‘instance’ of that model.

→After which we called ‘fit’ object from the model to fit our dataset.

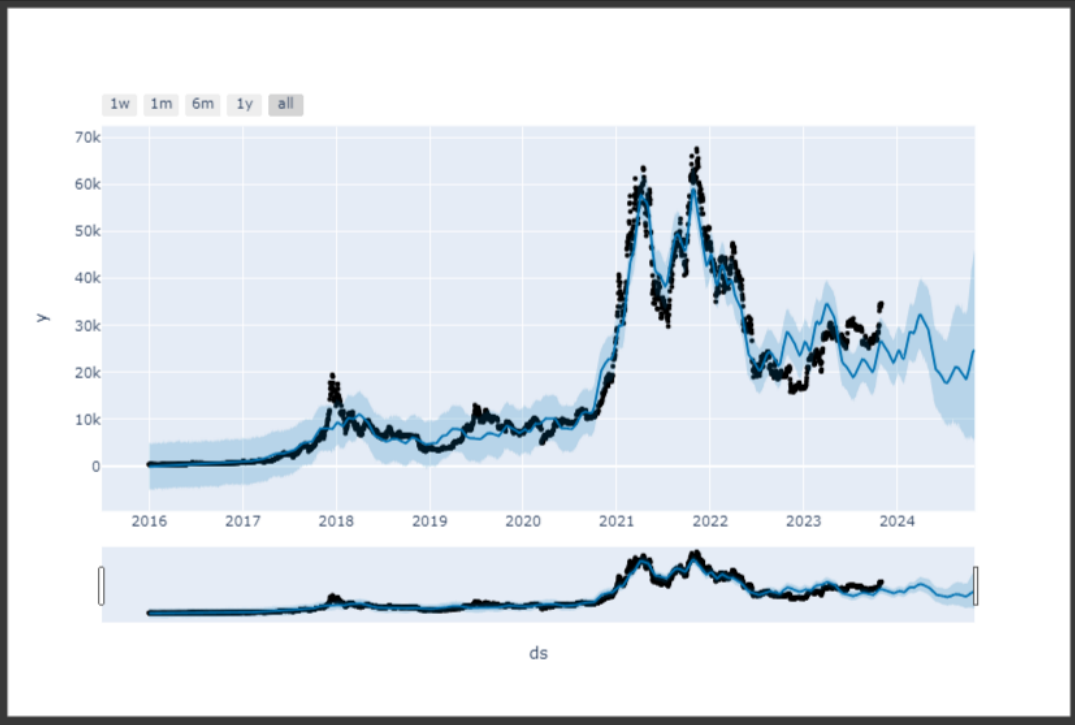


→Once the model was fitted it was time for us to test it and check if it gave reasonable predictions, we did that using 'predict' object of the model.

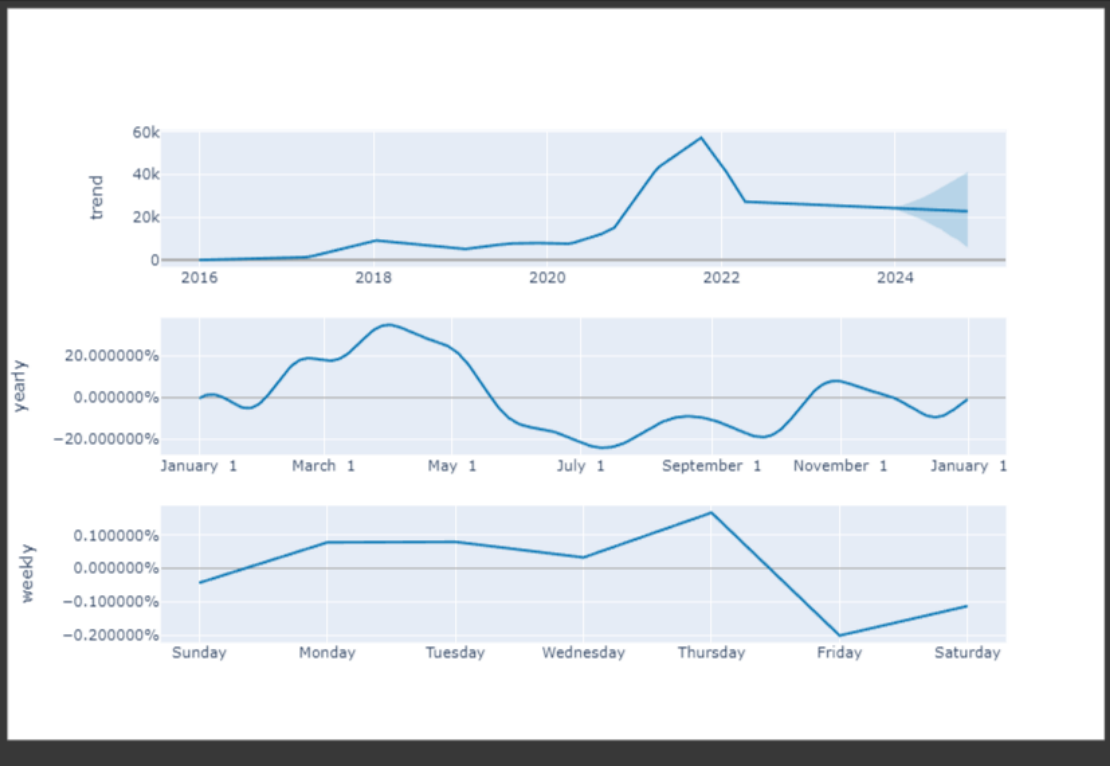
→After which we also plotted the future predictions of the model on the graph to understand it better.

```
[ ] from prophet.plot import plot_plotly, plot_components_plotly
```

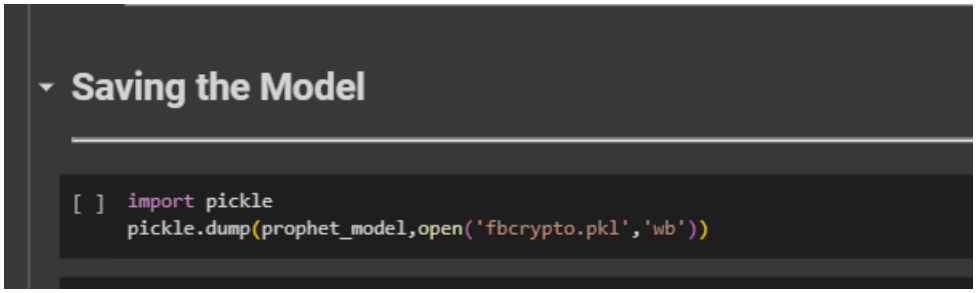
```
plot_plotly(prophet_model, y_preds)
```




```
plot_components_plotly(prophet_model, y_preds)
```



→As the last step of this model we saved the it using ‘pickel’



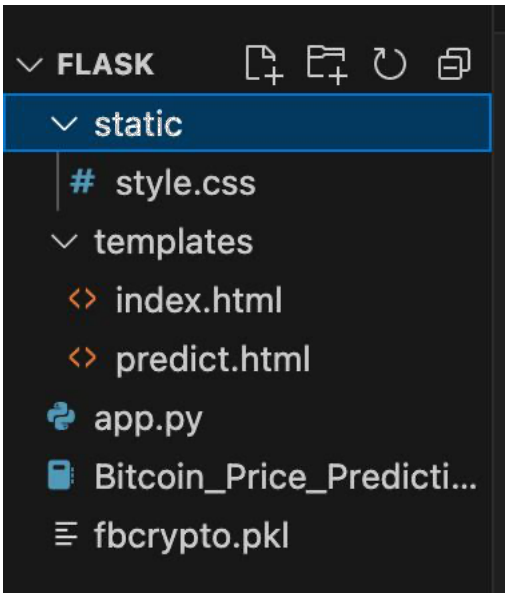
Model deployment

→ We are going to use Flask to deploy our machine learning model.

→ Flask requires a saved version of our ipynb where our model is being developed and we are using pickle to save our model in .pkl format.

→In the flask application, the input parameters are taken from the HTML page These factors are then given to the model to predict the price of bitcoin on a selected date and showcased on the HTML page to notify the user. Whenever the user interacts with the UI and selects the “predict” button, the next page is opened where the user selects the date and predicts the output.

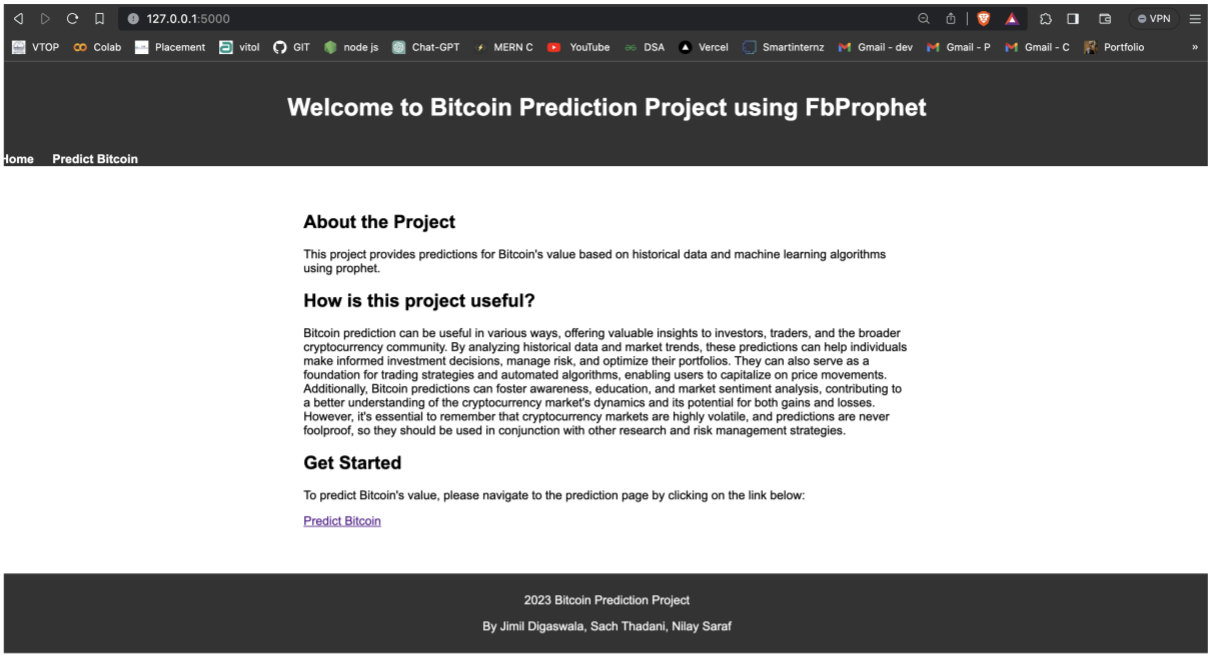
The file structure of our flask deployment looks like this:



→We have made two html files, index.html for the homepage and predict.html for the prediction page.

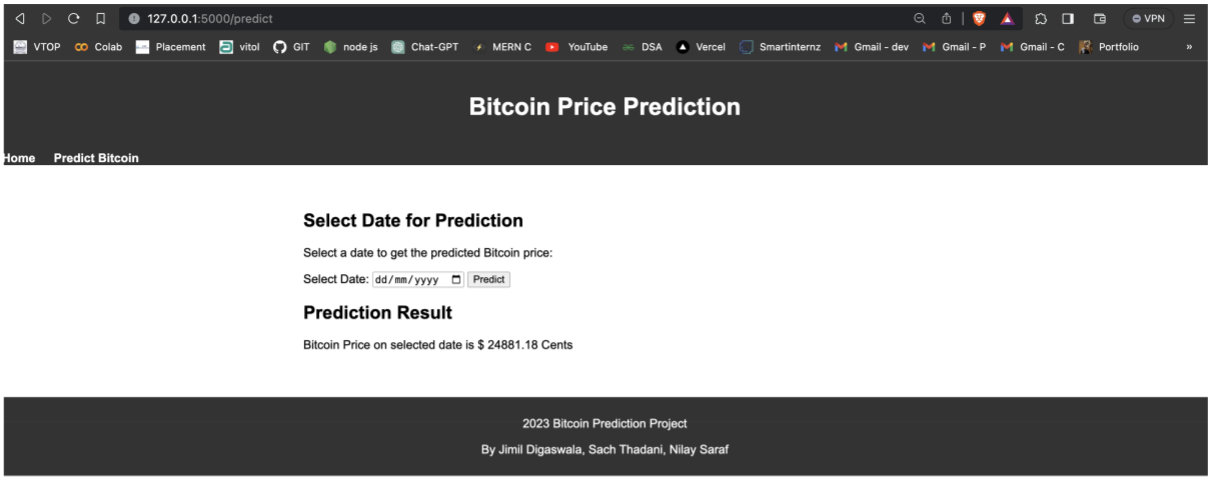
→ The code and deployment of the **index.html** looks like this:

```
tes > index.html > ...
<!DOCTYPE html>
<html>
<head>
    <title>Bitcoin Prediction Project</title>
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <header>
        <h1>Welcome to Bitcoin Prediction Project using FbProphet</h1>
    </header>
    <nav>
        <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="{{ url_for('prediction') }}">Predict Bitcoin</a></li>
        </ul>
    </nav>
    <main>
        <section>
            <h2>About the Project</h2>
            <p>This project provides predictions for Bitcoin's value based on historical data and machine learning algorithms using prophet.</p>
        </section>
        <section>
            <h2>How is this project useful?</h2>
            <p>Bitcoin prediction can be useful in various ways, offering valuable insights to investors, traders, and the broader cryptocurrency community.
                By analyzing historical data and market trends, these predictions can help individuals make informed investment decisions, manage risk, and
                optimize their portfolios. They can also serve as a foundation for trading strategies and automated algorithms, enabling users to capitalize
                on price movements. Additionally, Bitcoin predictions can foster awareness, education, and market sentiment analysis, contributing to a better
                understanding of the cryptocurrency market's dynamics and its potential for both gains and losses. However, it's essential to remember
                that cryptocurrency markets are highly volatile, and predictions are never foolproof, so they should be used in conjunction with other
                research and risk management strategies.</p>
        </section>
        <section>
            <h2>Get Started</h2>
            <p>To predict Bitcoin's value, please navigate to the prediction page by clicking on the link below:</p>
            <a href="{{ url_for('prediction') }}">Predict Bitcoin</a>
        </section>
    </main>
    <footer>
        <p> 2023 Bitcoin Prediction Project</p>
        <p> By Jimil Digaswala, Sach Thadani, Nilay Saraf </p>
    </footer>
</body>
</html>
```



→ The code and deployment of the **predict.html** looks like this:

```
<!DOCTYPE html>
<html>
<head>
    <title>Bitcoin Prediction - Select Date</title>
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <header>
        <h1>Bitcoin Price Prediction</h1>
    </header>
    <nav>
        <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="predict.html">Predict Bitcoin</a></li>
        </ul>
    </nav>
    <main>
        <section>
            <h2>Select Date for Prediction</h2>
            <p>Select a date to get the predicted Bitcoin price:</p>
            <form method="POST" action="/predict">
                <label for="date">Select Date:</label>
                <input type="date" id="date" name="Date">
                <input type="submit" value="Predict">
            </form>
        </section>
        <section>
            <h2>Prediction Result</h2>
            <p><span>{{ prediction_text }}</span></p>
        </section>
    </main>
    <footer>
        <p>2023 Bitcoin Prediction Project</p>
        <p> By Jimil Digaswala, Sach Thadani, Nilay Saraf </p>
    </footer>
</body>
</html>
```



Now, the main file of flask app.py is required to host the model as it contains the code which links the model, and the two html files responsible for ui and calling methods.

App.py code:

Task1: Importing libraries

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

Task2: Creating our model and loading out model using pandas.

```
app = Flask(__name__)
m = pd.read_pickle("fbcrypto.pkl")
```

Task3: Routing to HTML pages and making future predictions:

```
@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')

@app.route('/predict', methods=['POST', 'GET'])
def prediction():
    if request.method == 'POST':
        ds = request.form['Date']
        ds = str(ds)
        next_day = ds

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

        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)
```

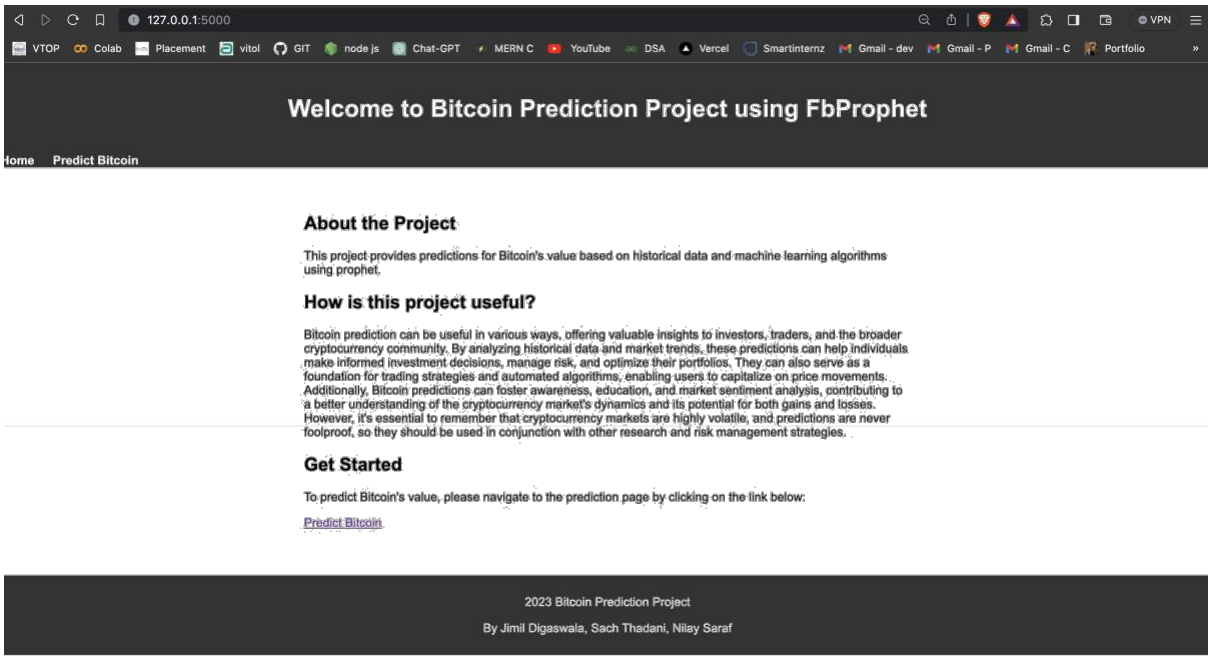
This is done using prophet model and the predictions made by the model is showed on the ui by routing in the html pages and then showing it in the deployment.

Steps to predict the bitcoin price using the ui:

1. Run the app.py file

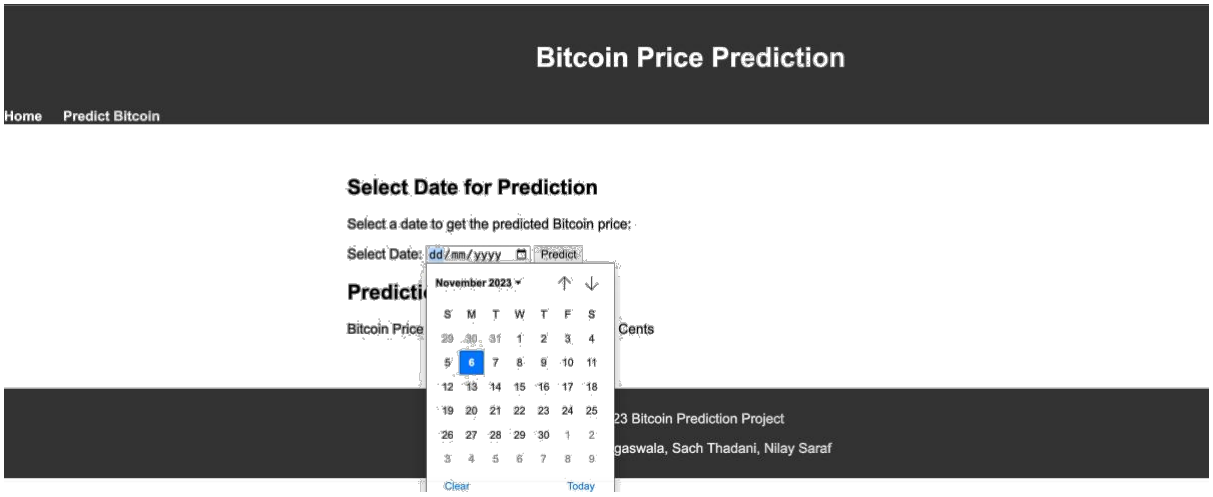
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

/usr/local/bin/python3 /Users/jimildigaswala/Desktop/flask/app.py
(base) jimildigaswala@Jimils-MacBook-Air flask % /usr/local/bin/python3 /Users/jimildigaswala/Desktop/flask/app.py
 * Serving Flask app 'app'
 * Debug mode: off
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
Press CTRL+C to quit
127.0.0.1 - - [06/Nov/2023 15:29:47] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 15:29:47] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 15:29:47] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [06/Nov/2023 15:30:43] "GET /predict HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 15:30:43] "GET /static/style.css HTTP/1.1" 304 -
24881.18
127.0.0.1 - - [06/Nov/2023 15:30:49] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 15:30:49] "GET /static/style.css HTTP/1.1" 304 -
```

2. A localhost link will be provided, navigate to the link and you will see the page given below:

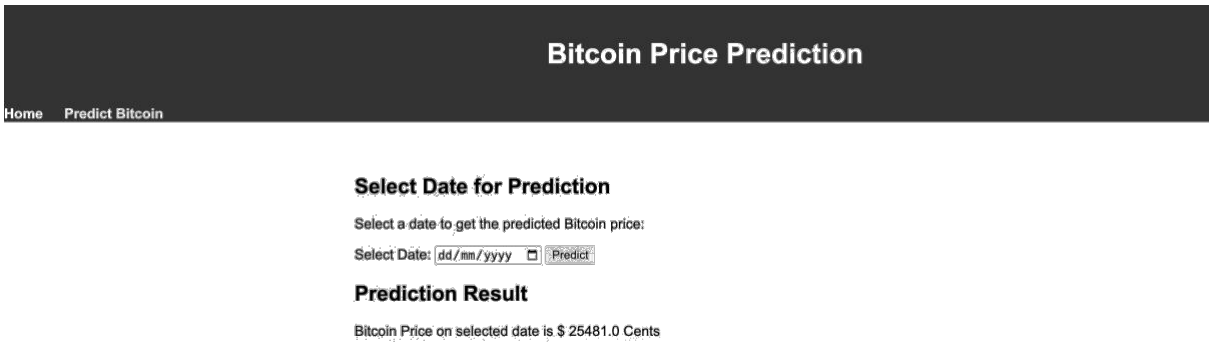


Now, click on the predict button and you will be navigated to the prediction page.

3. In the predict page, you will find a dialog box to select the date you want to predict the bitcoin price for:



4. After selecting the date, click on the predict button and the predicted price should be displayed on the screen.



8. Performance Testing:

Metrics:

MAE: 6366.8
MSE: 49493173.95
RMSE: 7835.14
R2: -2.78

```
[35]  1  error
```

	Date	Actual	Predicted	Difference
2557	2023-01-01	16547.914062	25727.026582	-9179.112519
2558	2023-01-02	16625.509766	25887.545653	-9262.035887
2559	2023-01-03	16688.847656	25992.850999	-9304.003343
2560	2023-01-04	16680.205078	26071.708875	-9391.503797
2561	2023-01-05	16863.472656	26200.157554	-9336.684898
...	...	...	...	...
2863	2023-11-03	34942.472656	24523.518713	10418.953943
2864	2023-11-04	34736.324219	24505.558053	10230.766166
2865	2023-11-05	35090.011719	24479.883658	10610.128061
2866	2023-11-06	35044.789062	24451.064407	10593.724656
2867	2023-11-07	35047.792969	24389.500061	10658.292908

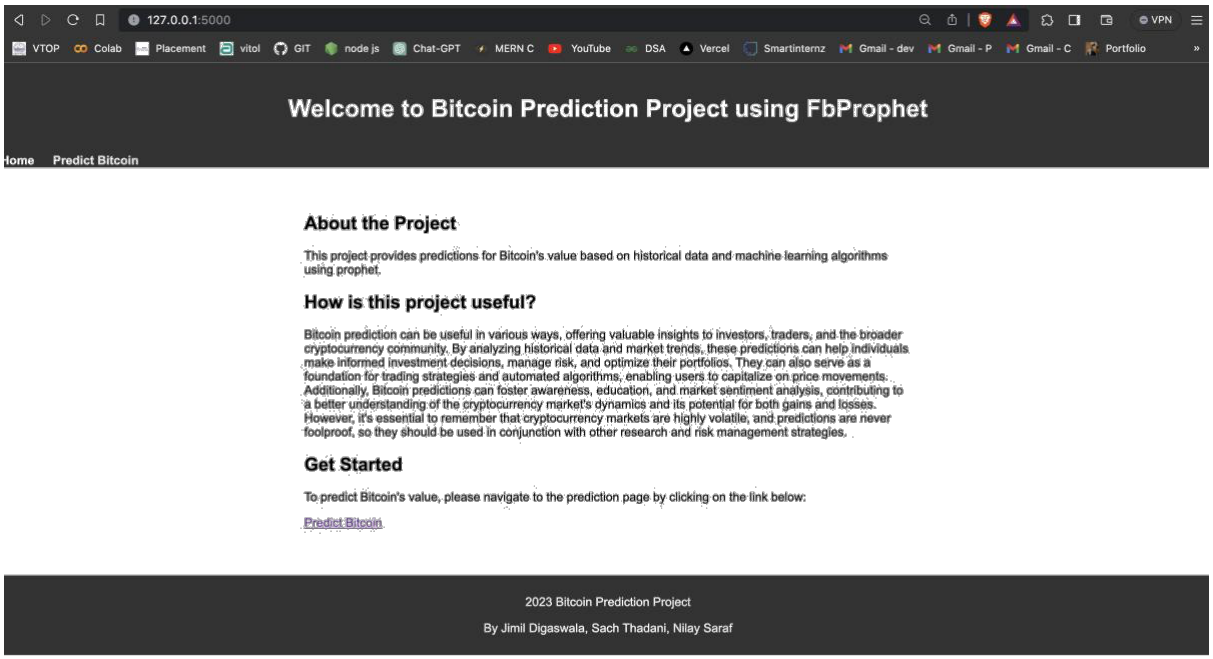
311 rows × 4 columns

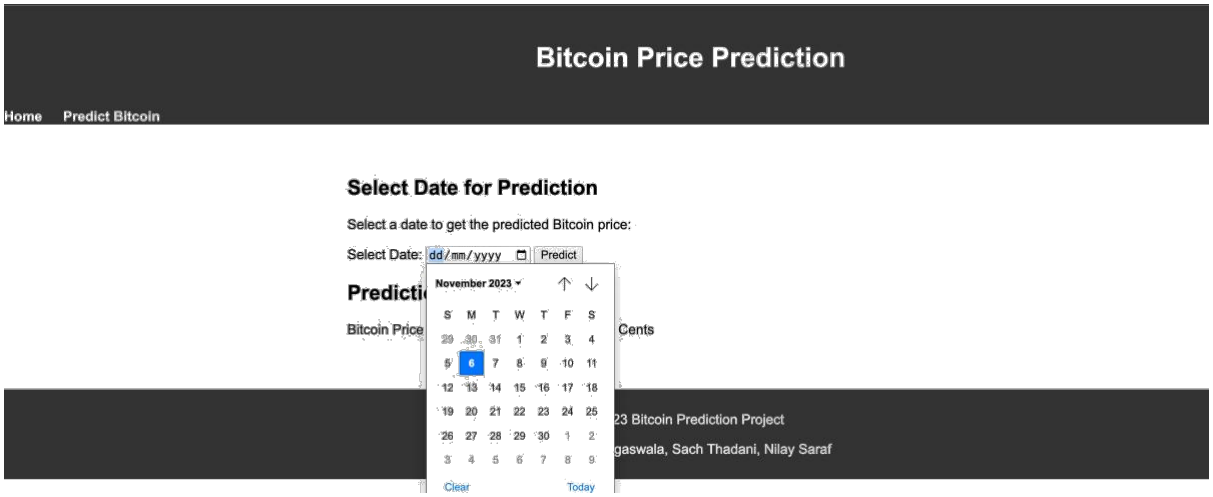
```
[41]  1  mae = mean_absolute_error(error["Actual"], error["Predicted"])
      2  mse = mean_squared_error(error["Actual"], error["Predicted"])
      3  rmse = np.sqrt(mse)
      4  r2 = r2_score(error["Actual"], error["Predicted"])
```

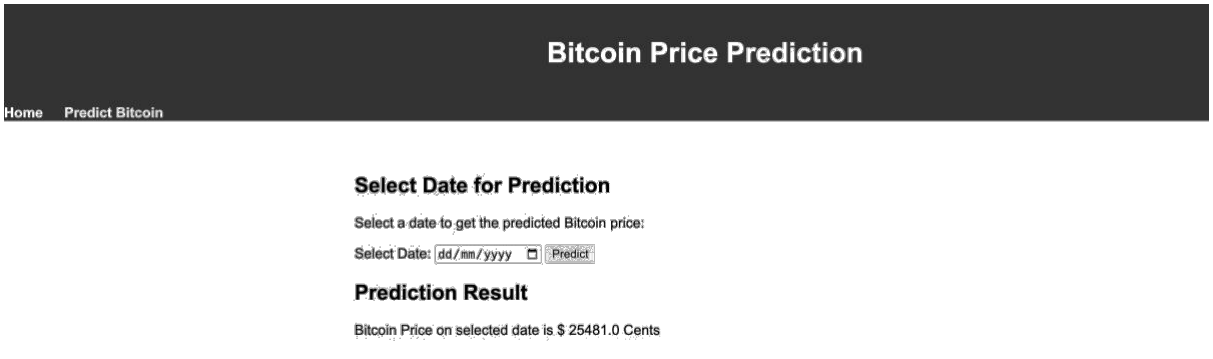
```
[42]  1  print("Mean Absolute Error: ", mae)
      2  print("Mean Squared Error: ", mse)
      3  print("Root Mean Squared Error: ", rmse)
      4  print("R2 Score: ", r2)
```

```
Mean Absolute Error:  6366.772234151625
Mean Squared Error:  49493173.942134805
Root Mean Squared Error:  7035.138516201
R2 Score:  -2.777329452234827
```

9. Output Screenshots :







10. Advantages and Disadvantages:

Advantages:

Ease of Use: FbProphet is designed to be user-friendly, making it easy for developers and analysts to implement time series forecasting without extensive expertise.

Handling Missing Data: FbProphet can handle missing data and outliers effectively, which is crucial when working with financial time series data that may have irregularities.

Automatic Seasonality Detection: The model can automatically detect and incorporate seasonality patterns in the data, simplifying the process for users who may not be familiar with the underlying seasonality components.

Interpretability: FbProphet provides insights into the forecast components, making it easier for users to interpret and understand the factors influencing the predicted prices.

Flexibility in Input Data: The model can handle both daily and seasonal data, providing flexibility in capturing different patterns and trends in the cryptocurrency market.

Open Source and Community Support: Being an open-source library, FbProphet benefits from community contributions and continuous improvement, ensuring updates and enhancements.

Scalability: FbProphet is scalable and can handle large datasets, making it suitable for applications with substantial historical financial data.

Disadvantages:

Sensitivity to Hyperparameters: The model's performance can be sensitive to the choice of hyperparameters, and tuning them for optimal results may require domain-specific knowledge.

Limited Forecast Horizon: FbProphet may not perform as well when forecasting for a very long-term horizon. Its strength lies in short to medium-term predictions.

Assumption of Additive Components: The model assumes that different components (trend, seasonality, holidays) contribute additively to the overall forecast. In cases where this assumption doesn't hold, the model's accuracy may be compromised.

Lack of Advanced Features: FbProphet may lack some advanced features available in other time series forecasting models, limiting its applicability in certain scenarios.

Handling Rapid Market Changes: Cryptocurrency markets can experience rapid changes and unexpected events. FbProphet may struggle to adapt quickly to sudden shifts in market dynamics.

Limited Uncertainty Estimation: While FbProphet provides uncertainty intervals, the estimation might be oversimplified, and more advanced models could offer more nuanced uncertainty estimates.

Dependency on Historical Data Quality: The accuracy of the predictions heavily relies on the quality and representativeness of the historical data. Inaccuracies or biases in the data can affect the model's performance.

11. Conclusion:

In conclusion, the crypto price prediction project employing FbProphet has successfully navigated through crucial phases, showcasing a robust framework for forecasting Bitcoin prices. The establishment of a dedicated Anaconda environment and the collection of comprehensive historical data from Yahoo Finance underscore the project's technical foundation. Through meticulous data pre-processing and visualization, the team ensured data integrity and visual clarity, laying the groundwork for effective model training. The implementation of FbProphet as the forecasting tool, combined with Flask for web deployment, reflects a holistic approach to time series prediction, demonstrating the team's technical proficiency. While achievements include user-friendly interfaces and insightful predictions, challenges such as hyperparameter sensitivity and limitations in forecasting horizons have been acknowledged. The project's learning outcomes span technical skills in data analysis, time series forecasting, and model deployment. Looking ahead, considerations for model refinement, exploration of advanced features, and continuous evaluation underscore the project's commitment to adaptability in the dynamic realm of cryptocurrency markets. In essence, this project not only contributes to the understanding of crypto market dynamics but also serves as a practical demonstration of machine learning applications in the financial domain, with implications for future advancements in predictive modelling.

12. Future Scope:

The future scope of this cryptocurrency price prediction project is promising and multifaceted. Firstly, continuous refinement and optimization of the forecasting model can enhance its accuracy and reliability, offering more precise predictions in an ever-evolving crypto market. Exploring advanced machine learning techniques, incorporating additional data sources, and leveraging ensemble models could further elevate the project's predictive capabilities. Additionally, integrating sentiment analysis from social media and news platforms may provide valuable insights into external factors influencing cryptocurrency prices. The project's scalability allows for the incorporation of multiple cryptocurrencies, expanding its utility across a broader spectrum of digital assets. Furthermore, collaboration with financial institutions, traders, and policymakers could yield valuable feedback and foster the project's integration into real-world financial decision-making processes. As the crypto space continues to grow and mature, the project holds the potential to contribute to a more informed and predictable market, promoting wider acceptance and understanding of digital assets among the general public.

13. Appendix:

GitHub link:

https://github.com/smartinternz02/SI-GuidedProject-589580-1697519158.git

Demonstration video link:

https://drive.google.com/file/d/1P0kfZ8UDmD4X3-pgpn52Eh3StG0sH8Rh/view?usp=sharing