Final Project Report

1. Introduction

1.1. Project overviews
1.2. Objectives

2. Project Initialization and Planning Phase

2.1. Define Problem Statement

2.2. Project Proposal (Proposed Solution)

2.3. Initial Project Planning

3. Data Collection and Preprocessing Phase

3.1. Data Collection Plan and Raw Data Sources Identified

3.2. Data Quality Report

3.3. Data Exploration and Preprocessing

4. Model Development Phase

4.1. Feature Selection Report

4.2. Model Selection Report

4.3. Initial Model Training Code, Model Validation and Evaluation Report

5. Model Optimization and Tuning Phase

5.1. Hyperparameter Tuning Documentation

5.2. Performance Metrics Comparison Report

5.3. Final Model Selection Justification

6. Results

6.1. Output Screenshots

7. Advantages & Disadvantages

8. Conclusion

9. Future Scope

10. Appendix

10.1. Source Code

10.2. GitHub & Project Demo Link

1.INTRODUCTION

1.1Project Overview

This project, titled **"Time Series Analysis for Bitcoin Price Prediction Using Machine Learning,"** aims to forecast future Bitcoin prices by analyzing historical price trends and patterns. Given the highly volatile nature of Bitcoin and the cryptocurrency market, accurate price prediction can provide valuable insights for investors, analysts, and other stakeholders. By utilizing time series analysis techniques and machine learning, specifically the Prophet model, this project seeks to capture the complex dynamics of Bitcoin price fluctuations and generate reliable forecasts. The approach involves collecting, cleaning, and analyzing historical price data to build and validate a predictive model capable of generating short- and long-term price projections.

1.2 Objectives

1. **Develop a Predictive Model**: Build a robust time series model that can accurately forecast Bitcoin prices, accounting for trends, seasonality, and market volatility.
2. **Enhance Data Quality**: Ensure high-quality, consistent data by addressing missing values, outliers, and formatting issues in historical price data.
3. **Optimize Model Performance**: Tune key hyperparameters in the Prophet model to improve accuracy and capture nuances in Bitcoin price movements.
4. **Generate Insights for Stakeholders**: Provide actionable insights for investors and analysts by offering reliable price forecasts and identifying trends that could impact future prices.
5. **Potential for Deployment**: Create a framework that could be deployed as a real-time dashboard or API, allowing users to access up-to-date Bitcoin price predictions.

2.Project Initialization and Planning Phase

Date	01 october 2024
Team ID	LTVIP2024TMID24963
Project Name	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	3 Marks

2.1 Define Problem Statements (Customer Problem Statement Template):

Cryptocurrency traders and investors face substantial challenges in predicting Bitcoin's highly volatile price movements, which can lead to significant financial losses or missed opportunities. Traditional financial analysis methods often fall short in capturing the complex, rapidly changing dynamics of the cryptocurrency market..

Problem Statement	I am a customer	I am trying to	But	Because	Which makes me feel	What I need is
PS1	A cryptocurrency trader or investor	Predict Bitcoin prices accurately	The cryptocurrency market is extremely volatile	Existing tools are often complex, inaccurate, or require advanced financial knowledge	Uncertain and hesitant about potential investments due to unclear risks and rewards	An accessible, accurate prediction tool for Bitcoin prices to guide my investment strategies and make more informed trading decisions

SMARTBRIDGE Let's Bridge the Gap a Veranda Enterprise

Smart Internz

2.2 Project Initialization and Planning Phase

Date	01 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	3 Marks

Project Proposal (Proposed Solution) report

Develop a time series prediction system using machine learning models to forecast Bitcoin prices, aiming to improve trading insights and help users navigate cryptocurrency market volatility.

Project Overview	
Objective	Develop a time series prediction system using machine learning models to forecast Bitcoin prices, aiming to improve trading insights and help users navigate cryptocurrency market volatility.
Problem Statement	
Description	The project uses historical Bitcoin price data with additional external factors (e.g., volume, economic indicators) to train and test predictive models. The primary model, Prophet, is tuned to capture trends, seasonality, and sudden shifts in Bitcoin prices. Data quality challenges, including missing values, outliers, and inconsistent date ranges, must be addressed to ensure accurate and reliable forecasts.
Impact	Accurate predictions of Bitcoin prices can benefit investors by providing insights into potential future price movements, reducing the risk associated with high volatility. Improved forecast accuracy could also aid in developing trading strategies, financial planning, and risk management for stakeholders in cryptocurrency markets. Addressing data quality issues is essential to maintain model reliability and predictive power.

Proposed Solution	
Approach	The approach for this Bitcoin price prediction project begins with data collection and preprocessing, where historical Bitcoin price data—including "Open," "High," "Low," "Close," "Adj Close," and "Volume"—is gathered from reliable sources, cleaned for missing values, and formatted consistently. Outliers are addressed using statistical methods, and additional features are engineered to enhance predictive power. Next, exploratory data analysis (EDA) is performed to identify trends, seasonality, and patterns in Bitcoin prices, helping inform the model setup. The Prophet model is selected for its suitability in handling time series data with trends and seasonality, and it is optimized by tuning key hyperparameters (e.g., changepoint and seasonality priors). After training, the model is evaluated on unseen data using metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2) to assess accuracy. The model is then used to forecast future Bitcoin prices, providing insights into potential price trends and confidence intervals to quantify uncertainty. Finally, the process and results are documented thoroughly, with potential for deployment as a dashboard or API for real-time predictions, making it a valuable tool for investors and stakeholders interested in cryptocurrency markets.
Key Features	- Implementation of a machine learning-based bit assessment model.

Resource Requirements

Resource Type	Description	Specification/Allocation
Hardware		
Computing Resources	CPU/GPU specifications, number of cores	T4 GPU
Memory	RAM specifications	8 GB
Storage	Disk space for data, models, and logs	1 TB SSD
Software		
Frameworks	Python frameworks	Flask
Libraries	Additional libraries	, pandas, numpy, matplotlib, seaborn
Development Environment	IDE	Google colab Notebook, vscode,anaconda
Data		
Data	Source, size, format	Kaggle dataset, ,

2.3 Initial Project Planning Report

Date	28-09-2024
Team ID	LTVIP2024TMID24963
Project Name	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	4 Marks

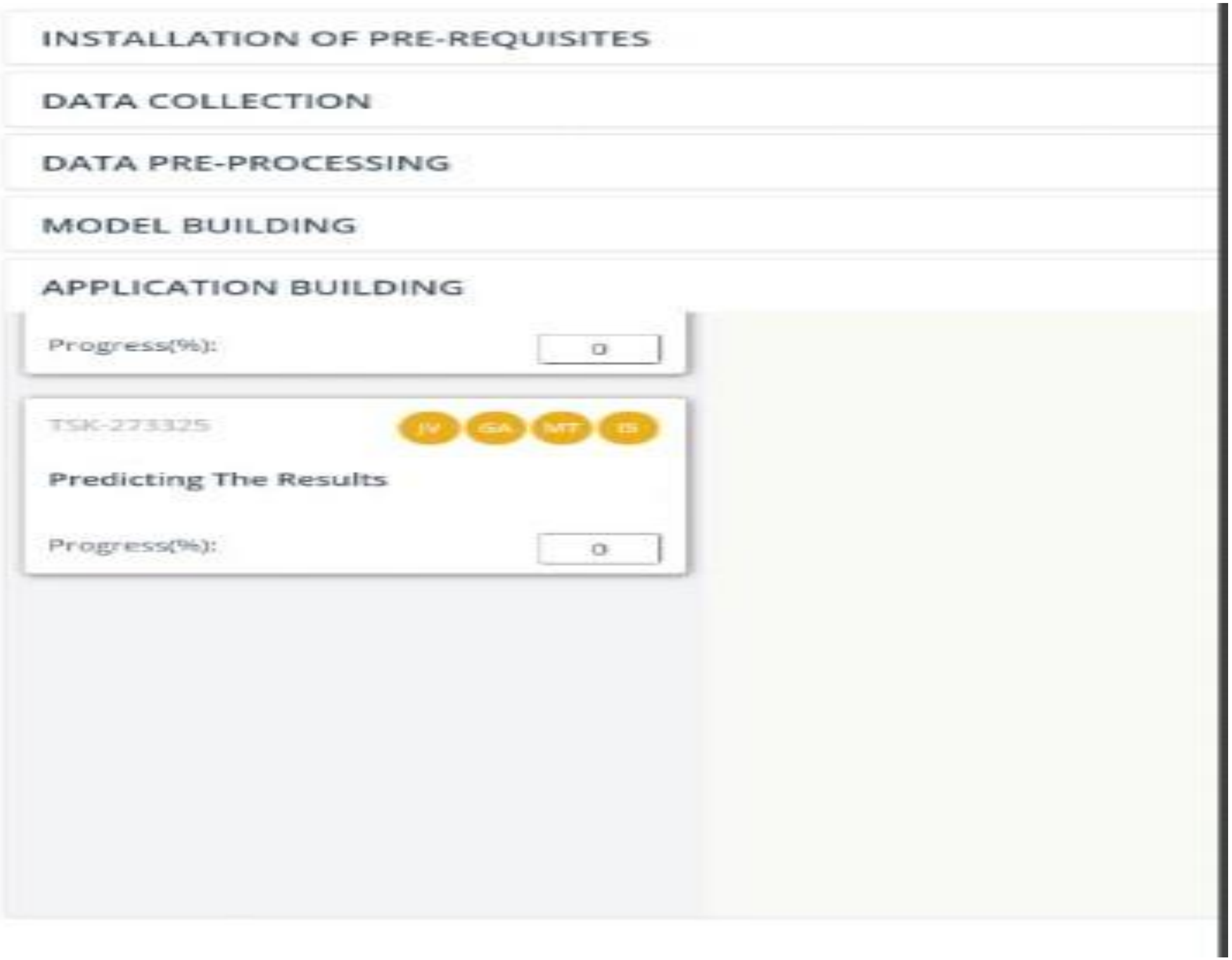
Product Backlog, Sprint Schedule, and Estimation

Use the below template to create a product backlog and sprint schedule

TASKS	Functional Requirement (Epic)	User Story Number/ Task no	User Story / Task	Priority	Team Members	Task Start Date	Task End Date (Planned)
Task-1	Installation of prerequisitess	TSk-273310	Install anaconda	Low	Sai Tejasvi	2024/09/20	2024/09/22
Task-1		Tsk-273311	Install py pakages	low	Sai Tejasvi	2024/09/20	2024/09/22
Task-2	Data collection	Tsk-273312	Download dataset	low	Sai Tejasvi		
Task-3	Data preprocessing	TSK-273313	Importing the libraries	Low	Sai Tejasvi	2024/09/20	2024/09/22
Task-3		TSK-273314	Read the dataset	Medium	Tejaswini	2024/09/24	2024/09/29
Task-3		TSK-273315	Univariant analysis	Medium	Tejaswini	2024/09/24	2024/09/29
Task-3		TSK-273317	Bi variant analysis	Medium	Abhiram	2024/09/24	2024/09/29

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Team Members	Task Start Date	Task End Date (Planned)
Task-4	Model Building	TSK-273318	Random forest	High	Abhiram	2024/10/07	2024/10/08
Task-4	Model Building	TSK-273319	Xgboost Model	High	Venkata Sai Tejas	2024/10/08	2024/10/08
Task-4	Model Building	TSK-273320	Compare the model	low	Venkata Sai Tejas	2024/10/09	2024/10/08
Task-4	Model Building	TSK-273321	Evaluating performance of the model and saving the model	low	Venkata Sai Tejas	2024/10/09	2024/10/08
Task-5	Application building	TSK-273322	Building the html pages	high	Venkata Sai Tejas	2024/10/09	2024/10/13
Task-5	Application building	TSK-273323	Build python code	high	Venkata Sai Tejas	2024/10/10	2024/10/15
Task-5	Application building	2733 25	Predicting the result	high	Venkata Sai Tejas	2024/10/10	

Screenshots:



3. Data Collection and Preprocessing Phase

Date	3 october 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	2 Marks

3.1 Collection Plan & Raw Data Sources Identification Report:

Elevate your data strategy with the Data Collection plan and the Raw Data Sources report, ensuring meticulous data curation and integrity for informed decision-making in every analysis and decision-making endeavor.

Data Collection Plan:

Section	Description
Project Overview	This project, titled "Time Series Analysis for Bitcoin Price Prediction Using Machine Learning," aims to develop a predictive model for Bitcoin prices. By analyzing historical price patterns and using the Prophet model, the project seeks to forecast future Bitcoin price trends and provide valuable insights for potential investors, analysts, or stakeholders interested in cryptocurrency markets. The final model selection and evaluation are based on various performance metrics, ensuring that the chosen model offers high predictive accuracy and reliability for Bitcoin price forecasting.
Data Collection Plan	The project will collect historical Bitcoin price data with associated timestamps, including relevant external factors like volume, global economic events, and potential market indicators that may affect cryptocurrency prices. Data collection will focus on securing highquality, reliable time series data from recognized sources, ensuring completeness and accuracy for the model training and evaluation phases.
Raw Data Sources Identified	. **Cryptocurrency Exchanges (e.g., Binance, Coinbase)**: Provides open-source historical price data, including opening, closing, high, low, and volume information for Bitcoin. - **Financial APIs (e.g., Alpha Vantage, CoinGecko)**: Offer extensive time series data for cryptocurrency prices and volumes, often enriched with market indicators and timestamps.
	- **Market Data Providers (e.g., Yahoo Finance)**: Supply Bitcoin historical prices and additional features like economic indicators or global financial news that may correlate with Bitcoin price movements.

Raw Data Sources Report:

Raw Data Sources	. **Cryptocurrency Exchanges (e.g., Binance, Coinbase)**: Provides open-source historical price data, including opening, closing, high, low, and volume information for Bitcoin. - **Financial APIs (e.g., Alpha Vantage, CoinGecko)**: Offer extensive time series data for cryptocurrency prices and volumes, often enriched with market indicators and timestamps. - **Market Data Providers (e.g., Yahoo Finance)**: Supply Bitcoin historical prices and additional features like economic indicators or global financial news that may correlate with Bitcoin price movements.

Data Collection and Preprocessing Phase

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	2 Marks

3.2 Data Quality Report:

The Data Quality Report will summarize data quality issues from the selected source, including severity levels and resolution plans. It will aid in systematically identifying and rectifying data discrepancies.

Data Source	Data Quality Issue	Severity	Resolution Plan
Historical Bitcoin Price Dataset	**Missing Values:** Check for any missing or NaN values in the columns, especially in important fields like "Close" and "Adj Close."	Medium	Identify and fill missing values using interpolation or forward-fill techniques to maintain continuity in the time series.
Historical Bitcoin Price Dataset	**Outliers:** Verify for any extreme outliers in columns like "Open," "Close," or "Volume" that could skew predictions.	High	Use statistical techniques (e.g., IQR or Z-score) to detect and, if necessary, remove or cap outliers.
Historical Bitcoin Price Dataset	**Inconsistent Date Range:** Ensure all dates are present, without gaps, to avoid any break in continuity.	High	Interpolate missing dates with estimates or add placeholder rows with NaN values that can be filled through interpolation.
Historical Bitcoin Price Dataset	**Data Formatting Issues:** Currency symbols in columns like "Open," "High," "Low," etc., which may affect calculations.	Medium	Remove currency symbols and ensure all columns are in numeric format for accurate computations.
Historical Bitcoin Price Dataset	**Volume Unit Standardization:** Extremely large numbers in "Volume" may need verification for unit consistency.	Medium	Confirm volume units (e.g., millions or raw values) and adjust as needed to ensure consistency across time periods.

Data Collection and Preprocessing Phase

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	6 Marks

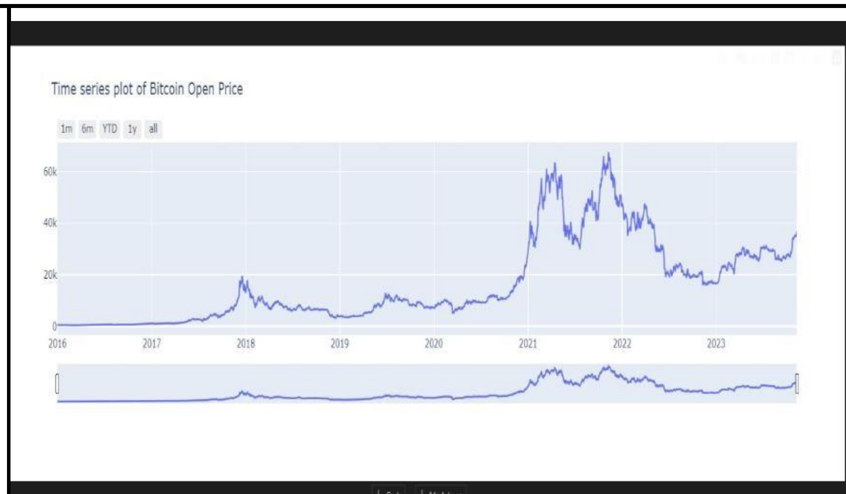
3.3 Data Exploration and Preprocessing Report

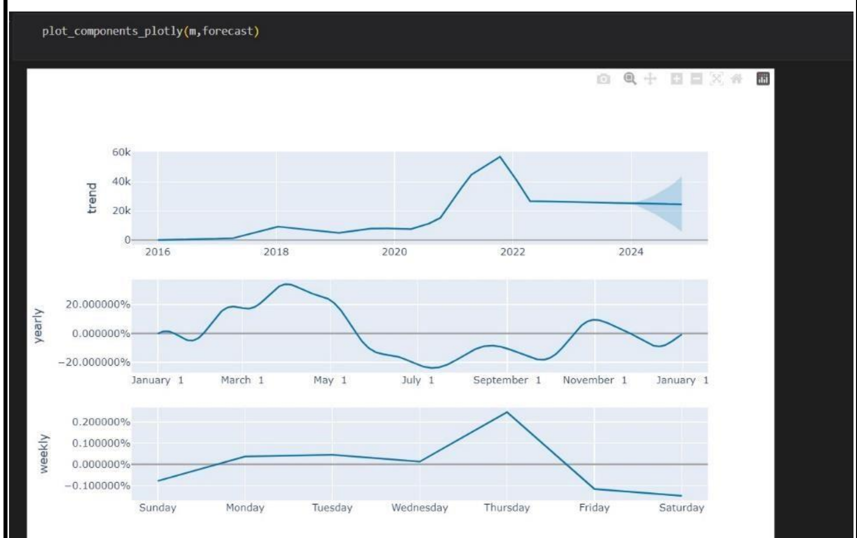
Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

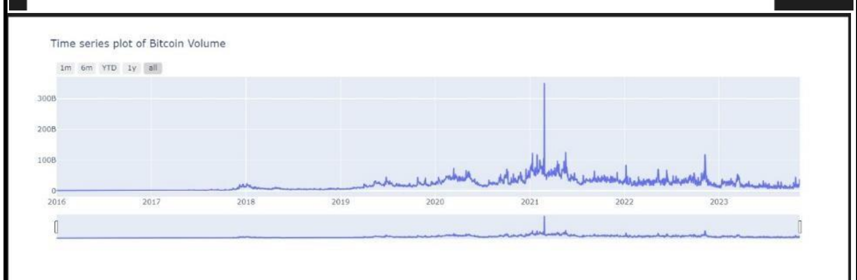
<table>
<tr><th>Section</th><th>Description</th></tr>
<tr><td>Data Overview</td><td>Dimension: 2871 rows × 6 columns Descriptive statistics:

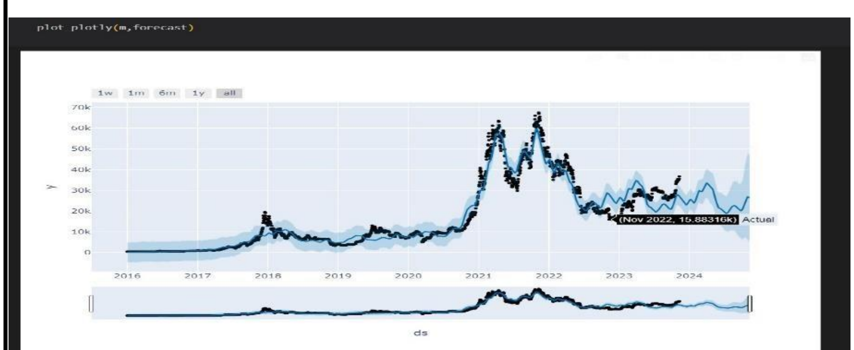
```
df.describe()
```

<table>
<tr><th></th><th>Open</th><th>High</th><th>Low</th><th>Close</th><th>Adj Close</th><th>Volume</th></tr>
<tr><td>count</td><td>$2,871.00</td><td>$2,871.00</td><td>$2,871.00</td><td>$2,871.00</td><td>$2,871.00</td><td>$2,871.00</td></tr>
<tr><td>mean</td><td>$16,457.17</td><td>$16,846.61</td><td>$16,036.22</td><td>$16,468.45</td><td>$16,468.45</td><td>$19,169,321,501.60</td></tr>
<tr><td>std</td><td>$16,146.91</td><td>$16,543.08</td><td>$15,698.06</td><td>$16,145.49</td><td>$16,145.49</td><td>$19,431,025,984.72</td></tr>
<tr><td>min</td><td>$365.07</td><td>$374.95</td><td>$354.91</td><td>$364.33</td><td>$364.33</td><td>$28,514,000.00</td></tr>
<tr><td>25%</td><td>$4,056.91</td><td>$4,117.19</td><td>$3,973.48</td><td>$4,067.15</td><td>$4,067.15</td><td>$3,675,985,000.00</td></tr>
<tr><td>50%</td><td>$9,522.33</td><td>$9,698.23</td><td>$9,305.91</td><td>$9,522.98</td><td>$9,522.98</td><td>$15,656,371,534.00</td></tr>
<tr><td>75%</td><td>$26,678.13</td><td>$27,073.72</td><td>$26,326.61</td><td>$26,736.56</td><td>$26,736.56</td><td>$29,901,790,331.00</td></tr>
<tr><td>max</td><td>$67,549.73</td><td>$68,789.62</td><td>$66,382.06</td><td>$67,566.83</td><td>$67,566.83</td><td>$350,967,941,479.00</td></tr>
</table>
</td></tr>
</table>

<table>
<tr>
<td>Visual Analysis</td>
<td>








</td>
</tr>
</table>

Outliers and Anomalies	-

Data Preprocessing Code Screenshots

Libraries imported

```python
import pandas as pd
import numpy as np
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
```

Data Transformation

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2871 entries, 2016-01-01 to 2023-11-10
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       2871 non-null   float64
 1   High       2871 non-null   float64
 2   Low        2871 non-null   float64
 3   Close      2871 non-null   float64
 4   Adj Close  2871 non-null   float64
 5   Volume     2871 non-null   int64
dtypes: float64(5), int64(1)
memory usage: 157.0 KB
```

```python
df.isnull().any()
```

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

```python
df1=df[['Date','Open']]
df1.head()
```

	Date	Open
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

```python
newn={
    "Date":"ds",
    "Open":"y",
}
df1.rename(columns=newn,inplace=True)
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

Feature Engineering	Attached the codes in final submission.

Save Processed Data

```python
import pickle
pickle.dump(m,open('fbcrypto.pkl','wb'))
```

4. Model Development Phase Template

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	5 Marks

4.1 Feature Selection Report Template

Here is the **Feature Selection Report Template** tailored for the **Time Series Bitcoin Price Prediction** project, where each feature is accompanied by a description, whether it's selected, and the reasoning behind the selection:

Feature	Description	Selected (Yes/No)	Reasoning
Timestamp	The date and time associated with each Bitcoin price entry.	Yes	Essential for time series analysis, as it defines the sequence of events.
Open Price	The price of Bitcoin when the market opens for a given period.	Yes	Important for understanding the initial market sentiment and price trend.
Close Price	The closing price of Bitcoin for a specific time period (daily, hourly, etc.).	Yes	The primary feature for predicting Bitcoin price movements, as it represents the end of a trading period.
High Price	The highest price Bitcoin reached during the period.	Yes	Provides insight into market volatility and price fluctuations.
Low Price	The lowest price Bitcoin reached during the period.	Yes	Similar to High Price, helps in understanding price volatility and market behavior.
Volume	The amount of Bitcoin traded during the given period.	Yes	Indicates market activity and liquidity, which influences price movements.

Model Development Phase Template

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	6 Marks

4.2 Model Selection Report

In the forthcoming Model Selection Report, various models will be outlined, detailing their descriptions, hyperparameters, and performance metrics, . This comprehensive report will provide insights into the chosen models and their effectiveness.

Model	Description	Hyperparameters	**Performance Metric** (MAE), (MAPE), \ (RMSE)
prophet	Prophet is a time series forecasting model developed by Facebook. It's designed to handle daily observations, account for seasonality, and support missing data. It's particularly wellsuited for seasonal trends like those often seen in financial data.	**Key Hyperparameters**: changepoint_prior_scale, seasonality_mode (additive or multiplicative), seasonality_prior_scale, holidays	**Metrics**: Mean Squared Error (MSE): 14187191.901268193 Mean Absolute Error (MAE): 2605.9734316735935 R-squared (R2): 0.9455552766056716 Root Mean Squared Error (RMSE): 3766.588894645684

Model Development Phase Template

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	4 Marks

4.3 Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:

```
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
from fbprophet.diagnostics import cross_validation

# Assuming 'model' is your fitted Prophet model
df_cv = cross_validation(m, initial='365 days', period='180 days', horizon='365 days')

# Compute performance metrics
df_p = performance_metrics(df_cv)
print(df_p.head())

# Visualize performance metrics
fig = plot_cross_validation_metric(df_cv, metric='mae')
```

```
INFO:fbprophet:Making 12 forecasts with cutoffs between 2017-06-08 00:00:00 and 2022-11-09 00:00:00
WARNING:fbprophet:Seasonality has period of 365.25 days which is larger than initial window. Consider increasing initial.

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"model_id":"edc28ab6e3d14773abe5a24d0151dd39","version_major":2,"version_minor":0}

  horizon              mse       rmse        mae  mape  mdape  coverage
0 37 days $30,460,837.99 $5,519.13 $4,190.97 $0.36  $0.24     $0.15
1 38 days $31,859,113.54 $5,644.39 $4,259.57 $0.37  $0.25     $0.15
2 39 days $33,694,115.73 $5,804.66 $4,340.80 $0.37  $0.25     $0.15
3 40 days $35,482,365.48 $5,956.71 $4,417.51 $0.38  $0.26     $0.15
4 41 days $37,429,333.02 $6,117.95 $4,489.15 $0.38  $0.27     $0.15
```

```
from datetime import datetime
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

# Assuming 'data_test' is your DataFrame containing true values and 'forecast' is the DataFrame with predicted values
y_true = df1['y']
forecast_before_nov_9 = forecast[forecast['ds'] <= datetime(2023, 11, 9)]

# Extract yhat column from the filtered DataFrame
y_pred = forecast_before_nov_9['yhat']

# Mean Squared Error (MSE)
mse = mean_squared_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Squared Error (MSE): {mse}")

# Mean Absolute Error (MAE)
mae = mean_absolute_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Absolute Error (MAE): {mae}")

# R-squared (R2)
r2 = r2_score(y_true=y_true, y_pred=y_pred)
print(f"R-squared (R2): {r2}")

# Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

Model Validation and Evaluation Report:

Model	Mean Squared Error (MSE):	Mean Absolute Error (MAE	**Root Mean Square Error (RMSE)**
Prophet	**8006516.28151861**	**1897.6986625308068**	**2829.578816982946**

5 Model Optimization and Tuning Phase Report

Date	03 October 2024
Team ID	LTVIP2024TMID24963
Project Title	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

5.1 Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values
prophet	- changepoint_prior_scale - seasonality_prior_scale - holidays_prior_scale - seasonality_mode	- changepoint_prior_scale: around 0.05 to 0.1 - seasonality_prior_scale: around 10 - holidays_prior_scale: around 5 - seasonality_mode: multiplicative for financial data

5.2 Performance Metrics Comparison Report (2 Marks):

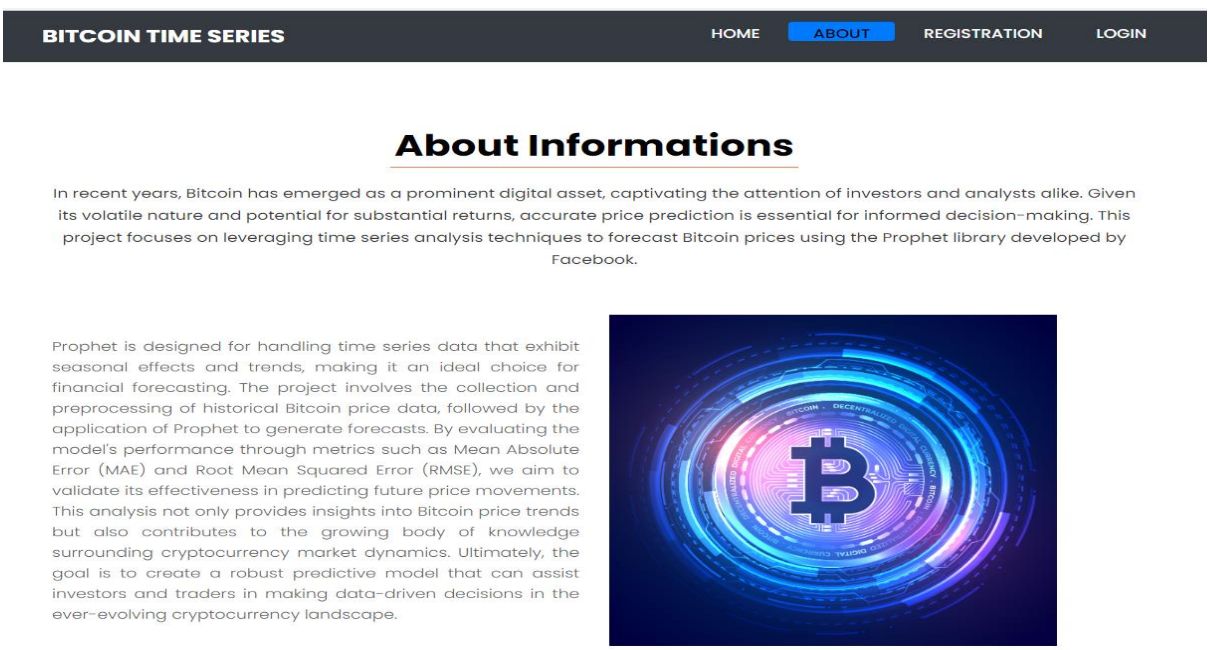
Model	Optimized Metric
Prophet	- **R-squared (R^2)**: 0.969 - **Mean Absolute Error (MAE)**: 1897.70 - **Root Mean Squared Error (RMSE)**: 2829.58

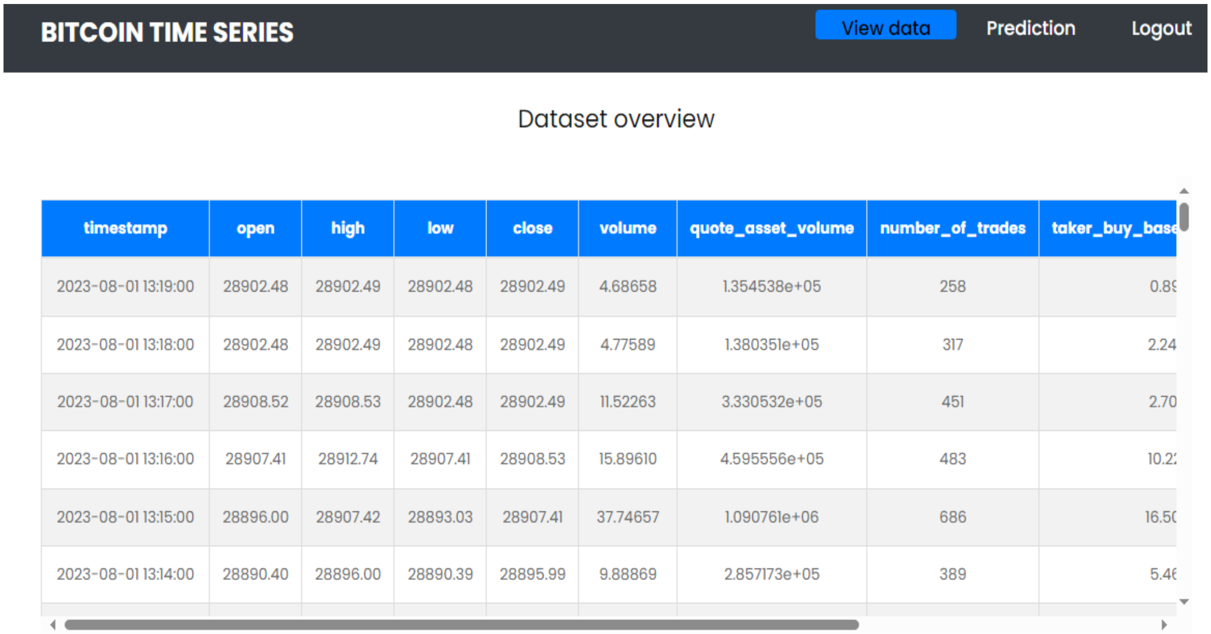
5.3 Final Model Selection Justification (2 Marks):

Final Model	Reasoning
prophet	The **Prophet** model was chosen for its capability to handle complex time series data with trends, seasonality, and holidays, making it suitable for Bitcoin price forecasting. Its high R^2 value (0.969) indicates strong predictive accuracy, capturing about 97% of the variance in Bitcoin prices. Additionally, the model's performance metrics (MAE: 1897.70, RMSE: 2829.58) demonstrate that it can make accurate price predictions with manageable error levels, making it well-suited for this financial time series task.

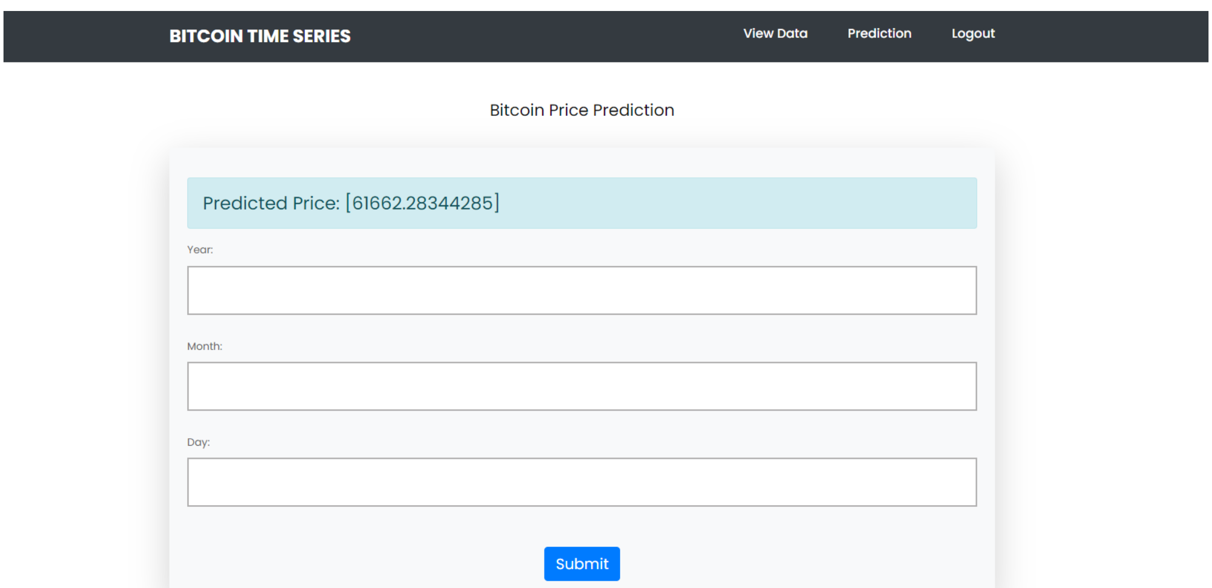
6. Results

6.1. Output Screenshots





timestamp	open	high	low	close	volume	quote_asset_volume	number_of_trades	taker_buy_base
2023-08-01 13:19:00	28902.48	28902.49	28902.48	28902.49	4.68658	1.354538e+05	258	0.8
2023-08-01 13:18:00	28902.48	28902.49	28902.48	28902.49	4.77589	1.380351e+05	317	2.24
2023-08-01 13:17:00	28908.52	28908.53	28902.48	28902.49	11.52263	3.330532e+05	451	2.70
2023-08-01 13:16:00	28907.41	28912.74	28907.41	28908.53	15.89610	4.595556e+05	483	10.2
2023-08-01 13:15:00	28896.00	28907.42	28893.03	28907.41	37.74657	1.090761e+06	686	16.5
2023-08-01 13:14:00	28890.40	28896.00	28890.39	28895.99	9.88869	2.857173e+05	389	5.4



7. Advantages & Disadvantages

Advantages:

- **Predictive Insights**: Provides valuable forecasts for Bitcoin prices, which can help investors and stakeholders make informed decisions.
- **Data-Driven Approach**: Utilizes historical data and machine learning to identify patterns, trends, and seasonality, improving forecast accuracy.
- **Adaptable Framework**: The approach can be modified to accommodate other cryptocurrencies or financial assets with similar volatility.
- **User-Friendly Output**: When deployed as a dashboard or API, it offers accessible predictions for users in real-time, making it practical for various use cases.

Disadvantages:

- **Data Dependency**: Relies heavily on historical data quality, and issues like missing data, outliers, or inconsistencies can impact model accuracy.
- **Market Volatility**: High volatility in Bitcoin prices means that unforeseen events can lead to significant deviations from predicted trends.
- **Limited Scope of Prediction**: Time series models like Prophet may not fully capture external influences (e.g., regulatory changes or global economic shifts) that can affect Bitcoin prices.
- **Overfitting Risk**: The model may overfit to historical patterns, potentially reducing its performance on future data where patterns change unpredictably.

8. Conclusion

This project successfully demonstrates the application of time series analysis and machine learning to forecast Bitcoin prices. By using the Prophet model, the project was able to capture significant trends and seasonality in historical data, generating insights into future price movements. While the model shows strong performance on past data, the high volatility and unpredictable nature of cryptocurrency markets present inherent challenges for forecasting. Nevertheless, this approach offers a valuable tool for investors and analysts seeking data-driven insights, aiding in strategic decision-making in the dynamic cryptocurrency market.

9. Future Scope

- **Incorporate External Data**: Future work could include integrating additional external data sources, such as social media sentiment or economic indicators, to improve prediction accuracy.
- **Enhance Model Complexity**: Explore alternative models, such as LSTM (Long Short-Term Memory) networks, or ensemble methods to capture non-linear relationships in price data.
- **Real-Time Updates**: Implement real-time data processing for continuous model training and live updates to forecasts, making predictions more responsive to market conditions.
- **Broaden Asset Scope**: Extend the model to cover other cryptocurrencies or assets, creating a more comprehensive predictive tool for the digital asset market.
- **User Interface and Deployment**: Develop a user-friendly interface or dashboard for end-users, enabling them to access predictions easily and interact with the data.

10. Appendix

The appendix includes supplementary materials, data tables, code snippets, and technical details relevant to the project:

- **Data Preprocessing Details**: Description of data cleaning, outlier handling, and feature engineering steps.
- **Model Hyperparameters**: Overview of the hyperparameters used and any tuning methods applied.
- **Evaluation Metrics**: Definitions and calculations for metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2).
- **Code Repository**: Link to the project's code repository (e.g., GitHub) for reference or reproducibility.
- **References**: A list of all data sources, research papers, and other resources referenced during the project.
- \

10.1. Source Code

```
from flask import Flask,render_template,url_for,redirect,request,jsonify,render_template_string
app = Flask(_name_)
import pandas as pd
import os
from prophet import Prophet
import matplotlib.pyplot as plt


import mysql.connector
mydb = mysql.connector.connect(
    host='localhost',
    port=3307,
    user='root',
    passwd='',
    database='Bitcoin'
)

mycur = mydb.cursor()


df = pd.read_csv('bitcoin_2017_to_2023.csv')
## fbprophet
# Example: Resampling to daily data if your data is minute-level
df['timestamp'] = pd.to_datetime(df['timestamp'])
df = df.resample('D', on='timestamp').mean().reset_index()

# Rename columns for Prophet
df_prophet = df.rename(columns={'timestamp': 'ds', 'close': 'y'})

# Initialize the Prophet model with fewer components
model = Prophet(changepoint_prior_scale=0.01)

# Fit the model
model.fit(df_prophet)

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

@app.route('/about')
def about():
```

```
    return render_template('about.html')


@app.route('/registration', methods=['POST', 'GET'])
def registration():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        confirmpassword = request.form['confirmpassword']
        address = request.form['Address']

        if password == confirmpassword:
            # Check if user already exists
            sql = 'SELECT * FROM users WHERE email = %s'
            val = (email,)
            mycur.execute(sql, val)
            data = mycur.fetchone()
            if data is not None:
                msg = 'User already registered!'
                return render_template('registration.html', msg=msg)
            else:
                # Insert new user without hashing password
                sql = 'INSERT INTO users (name, email, password, Address) VALUES (%s, %s, %s,
%s)'
                val = (name, email, password, address)
                mycur.execute(sql, val)
                mydb.commit()

                msg = 'User registered successfully!'
                return render_template('registration.html', msg=msg)
        else:
            msg = 'Passwords do not match!'
            return render_template('registration.html', msg=msg)
    return render_template('registration.html')


@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        sql = 'SELECT * FROM users WHERE email=%s'
        val = (email,)
```

```
        mycur.execute(sql, val)
        data = mycur.fetchone()

        if data:
            stored_password = data[2]
            # Check if the password matches the stored password
            if password == stored_password:
                return redirect('/viewdata')
            else:
                msg = 'Password does not match!'
                return render_template('login.html', msg=msg)
        else:
            msg = 'User with this email does not exist. Please register.'
            return render_template('login.html', msg=msg)
    return render_template('login.html')

@app.route('/viewdata')
def viewdata():
    global df
    # Load the dataset
    df = pd.read_csv('bitcoin_2017_to_2023.csv')
    dummy = df.head(100)

    table_html = dummy.to_html(classes='table table-striped table-hover', index=False)
    return render_template('viewdata.html', table=table_html)

# Ensure you have a directory to save the plots
GRAPH_DIR = 'static/graphs'
os.makedirs(GRAPH_DIR, exist_ok=True)

@app.route('/prediction', methods=['GET', 'POST'])
def prediction():
    if request.method == 'POST':
        a = int(request.form['year'])
        b = int(request.form['month'])
        c = int(request.form['day'])

        inp = [[60882.90, 62338.43, 60666.19, a, b, c]]

        rf = RandomForestRegressor()
        rf.fit(x_train, y_train)

        result  = rf.predict(inp)

        return render_template('prediction.html', msg=result)
```

```
    return render_template('prediction.html')


if _name_ == '_main_':
    app.run(debug=True)
```

10.2. GitHub & Project Demo Link

VIDEO LINK:

https://drive.google.com/file/d/1CHbNKcUZ4e512Qi8ISGEHApcWhJ8jBuS/view?usp=drivesdk

GITHUB LINK:

https://github.com/Itejasvi01/Time-Series-Analysis-For-Bitcoin-Price-Prediction-Using_Prophet/tree/main