# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**“Jnana Sangama”, Belagavi-590018, Karnataka**

# ADVANCED MACHINE LEARNING
# (18AI72)

## Mini Project Report on

**“Bitcoin Price Prediction”**

***Submitted in partial fulfillment of the requirements for the award of the degree of***

***Bachelor of Engineering***
***in***
***Artificial Intelligence & Machine Learning***

**Submitted by**

| | |
|---|---|
| **Shreyas R S** | **1BI20AI046** |
| **Sanjana V** | **1BI20AI043** |

**Under the Guidance of**

**Mrs. Subha Meenakshi**
**Department of AI&ML, BIT**
**Bengaluru-560 004**

## Department of Artificial Intelligence & Machine Learning

**Bangalore Institute of Technology**
**K.R. Road, V.V. Pura, Bengaluru-560 004**
**2023-24**

## VISVESVARAYA TECHNOLOGICAL UNIVERSITY
**"Jnana Sangama", Belagavi-590018, Karnataka**

## BANGALORE INSTITUTE OF TECHNOLOGY
**Department of Artificial Intelligence & Machine Learning**
**K.R. Road, V.V.Puram, Bengaluru-560 004**

### *Certificate*

This is to certify that Advanced Machine Learning mini project work entitled "**Bitcoin Price Prediction**" carried out by

| USN | Name |
|---|---|
| **1BI20AI046** | **Shreyas R S** |
| **1BI20AI043** | **Sanjana V** |

bonafide students of **Bangalore Institute of Technology** in partial fulfilment for the award of degree of **Bachelor of Engineering** in **Artificial Intelligence & Machine Learning** under Visvesvaraya Technological University, Belagavi, during the academic year 2023-24 is true representation of mini project work completed satisfactorily.

Mrs. Subha Meenakshi
Dept. of AI&ML
BIT
Bengaluru

Dr. Jyothi D. G.
Professor & HoD
Dept. of AI&ML
BIT, Bengaluru.

Dr. Aswath M. U.
Principal
BIT
Bengaluru.

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the completion of a task would be incomplete without crediting the people who made it possible, whose constant guidance and encouragement crowned the efforts with success.

We would like to profoundly thank Management of Bangalore Institute of Technology for providing such a healthy environment for successful completion of mini project work.

We would like to express our thanks to the Principal **Dr. Aswath M.U.** for his encouragement that motivates us for the successful completion of mini project work.

It gives us immense pleasure to thank **Dr. Jyothi D.G.**, Professor & Head, Department of Artificial Intelligence & Machine Learning and **Mrs. Subha Meenakshi** for their constant support and guidance throughout the Mini Project work.

We are very much pleasured to express our sincere gratitude to the friendly co-operation showed by all the staff members of Artificial Intelligence and Machine Learning Department, BIT.

Last but not the least, we would here by acknowledge and thank our friends and family who have been our source of inspiration always instrumental in successful completion of the Project work.

| | |
|---|---|
| Date : 19th Dec 2023 | Shreyas R S |
| Place: Bengaluru | Sanjana V |

# ABSTRACT

In the dynamic landscape of cryptocurrency markets, predicting the price movements of assets like Bitcoin has become a challenging yet crucial task for investors and traders. This project harnesses the power of machine learning to develop a robust model for forecasting Bitcoin prices.

Our approach involves collecting and preprocessing extensive historical data, encompassing various market indicators and relevant external factors. Leveraging advanced machine learning algorithms, including but not limited to regression models, neural networks, and ensemble methods, we strive to capture the intricate patterns within the cryptocurrency market. Our results are validated through comprehensive back testing and out-of-sample testing, assessing the model's ability to generalize to unseen data. By comparing our predictions against actual market movements, we aim to demonstrate the effectiveness and reliability of our machine learning-based approach.

Additionally, we discuss the challenges faced during the project, potential limitations, and avenues for future improvements. The ultimate goal of this endeavour is to provide valuable insights for stakeholders in the cryptocurrency space, aiding them in making informed decisions in a volatile and rapidly evolving market. This project aims at developing a predictive model for Bitcoin prices, shedding light on the potential applications of machine learning in understanding and forecasting cryptocurrency market dynamics.

# INDEX

# LIST OF FIGURES

# CHAPTER – 1

# INTRODUCTION

## Chapter - 1

# INTRODUCTION

## 1.1 Overview

In the ever-fluctuating landscape of cryptocurrency, predicting Bitcoin prices has emerged as a critical endeavour, offering potential rewards for those adept at navigating market volatility. This project represents a pioneering foray into the realm of machine learning, seeking to address the complex challenge of forecasting Bitcoin price movements. In doing so, we aspire to construct a robust predictive model capable of deciphering the intricate patterns that govern the cryptocurrency's valuation.

The cryptocurrency markets, epitomized by their inherent dynamism, demand innovative approaches to forecasting that extend beyond traditional financial analysis. It is within this context that our exploration into machine learning assumes paramount significance, promising insights that transcend mere predictive capabilities. By leveraging the power of advanced algorithms, we endeavour to unravel the underlying dynamics of Bitcoin's price fluctuations, contributing to a deeper understanding of the forces shaping the cryptocurrency landscape.

This project unfolds as a comprehensive journey, commencing with meticulous data collection from diverse sources, spanning historical price data and relevant market indicators. Our commitment extends to rigorous data preprocessing, encompassing tasks such as handling missing values, scaling numerical features, and employing sophisticated techniques like log-transformations to address skewed distributions. Feature engineering, the next pivotal step in our methodology, involves the creation of insightful features that can potentially impact Bitcoin prices. This includes incorporating moving averages, relative strength index (RSI), and other technical indicators, enriching the dataset with valuable information. These carefully curated features serve as the building blocks for our machine learning model, enhancing its capacity to capture and understand the nuanced patterns inherent in the cryptocurrency market.

Beyond the realm of predictive modelling, our project aspires to contribute to the broader understanding of the factors influencing Bitcoin markets. By delving into the intricacies of feature importance and model interpretation, we seek to shed light on the dynamics that drive cryptocurrency valuations. In doing so, we not only enhance our ability to predict Bitcoin prices but also contribute valuable insights to the evolving discourse surrounding the cryptocurrency ecosystem.

In summary, our project is a holistic exploration into the fusion of machine learning and cryptocurrency forecasting. By navigating the complexities of data science, feature engineering, and algorithmic optimization, we endeavour to not only predict Bitcoin prices with precision but also contribute to the collective knowledge shaping the future of cryptocurrency markets.

# CHAPTER – 2

# LITERATURE REVIEW

## Chapter - 2

# LITERATURE REVIEW

## Paper 1:

[1] **Title: "**Forecasting Financial Time Series with Machine Learning Algorithms**"-** Antonis Polemitis and Antonis C. Simotas

**Summary:**

[1] This paper explores the application of machine learning algorithms for financial time series forecasting, including Bitcoin prices. The objective is to evaluate the predictive performance of various algorithms in the context of cryptocurrency markets.

## Paper 2:

**[2] Title: "Modelling and Forecasting Bitcoin Volatility Index"-** Yaya O.S. and Ogbonna A.O

**Summary:**

[2] The paper focuses on modeling and forecasting the volatility of Bitcoin prices. Objectives include understanding the key determinants of Bitcoin volatility and assessing the efficacy of different models in capturing and predicting volatility patterns.

## Paper 3:

**[3] Title: " Forecasting daily exchange rate using hybrid model of ARIMA and machine learning techniques",** Moazeni, S., & Bhowmik, T.

**Summary:**

[3] The study proposes a hybrid model combining ARIMA and machine learning techniques for forecasting exchange rates, including Bitcoin. Objectives involve evaluating the synergy of these models in enhancing prediction accuracy.

## Paper 4:

**[4] Title: "Forecasting the Price of Bitcoin Using Social Media Sentiment Analysis"** - Wang, Y., Ma, Y., Wang, J., & Liu, W.

**Summary:**

[4] The paper explores the integration of sentiment analysis from social media into Bitcoin

price forecasting models. Objectives include understanding the impact of social sentiment on cryptocurrency markets.

## Paper 5:

**[5] Title: "Social signals and algorithmic trading of Bitcoin",** Garcia, D., & Schweitzer, F.

**Summary:**

[5] This paper investigates the influence of social signals on Bitcoin trading strategies. Objectives involve understanding how social media and online communities impact trading decisions in cryptocurrency markets.

## Paper 6:

**[6] Title: "Predicting the price of Bitcoin using Bayesian regression" -** Kim, Y.B., Lee, J.H., & Ki, E.J.

**Summary:**

**[6]** The study employs Bayesian regression to predict Bitcoin prices. Objectives include assessing the Bayesian approach's effectiveness in capturing the probabilistic nature of cryptocurrency price movements. Techniques: Bayesian Regression

## Paper 7:

**[7] Title: "Bitcoin Price Prediction and Trading Strategy using Machine Learning with Python" -** Shah, D., & Zhang, Q.

**Summary:**

[7] This paper presents a comprehensive approach to Bitcoin price prediction and trading strategy development using machine learning. Objectives involve building a model capable of generating profitable trading signals.

## Existing System:

The existing landscape of Bitcoin price prediction systems is characterized by a diverse array of approaches, ranging from traditional financial analysis to cutting-edge machine learning algorithms. Traditional methods often rely on fundamental analysis, considering factors such as market demand, macroeconomic indicators, and historical price trends. Technical analysis, on the other hand, involves studying past market data and chart patterns to forecast future price movements. Many existing systems heavily rely on historical price trends, technical indicators, and traditional statistical models to forecast Bitcoin prices. However, the dynamic and unpredictable nature of cryptocurrency markets poses challenges for these conventional methods. A notable drawback in the existing systems is the limited adaptability to rapidly changing market conditions and the intricate patterns inherent in the cryptocurrency realm.

## Problem Statement:

In the context of the ever-volatile cryptocurrency market, accurate prediction of Bitcoin prices remains a formidable challenge. Existing prediction systems often fall short in adapting to the dynamic nature of the market, relying on conventional methods that struggle to capture intricate patterns and sudden fluctuations. The inadequacy of current approaches underscores the necessity for a sophisticated model capable of navigating the complexities of the cryptocurrency landscape, leveraging advanced machine learning techniques to enhance predictive accuracy and contribute valuable insights into the factors influencing Bitcoin markets.

## Proposed System:

The proposed system in this project endeavours to elevate Bitcoin price prediction by harnessing the power of XGBoost, a robust machine learning algorithm. Unlike traditional methods that may rely solely on historical trends or technical indicators, our approach integrates advanced data science techniques, encompassing meticulous data collection, preprocessing, and feature engineering. By leveraging the capabilities of XGBoost, we aim to construct a predictive model that can discern intricate patterns within the cryptocurrency market, adapting to its inherent volatility. The project emphasizes not only heightened accuracy in predicting Bitcoin prices but also a nuanced understanding of the factors influencing these predictions. Through the iterative optimization of algorithmic parameters, we seek to enhance the model's adaptability to dynamic market conditions, contributing to a more resilient and effective predictive system for Bitcoin prices.

# CHAPTER – 3

# SYSTEM REQUIREMENT

Chapter - 3

# SYSTEM REQUIREMENT SPECIFICATION

## 3.1 Software Requirements:

- Programming Languages: Python
- Software required: Jupyter notebook
- Python Libraries: NumPy, Pandas, Scikit-learn, XGBoost, Matplotlib, Seaborn
- Operating System: Windows
- Terminal Window : To run the Python file using commands

  Software requirements involve an operating system compatible with chosen programming languages (e.g., Python, MATLAB, C++).

## 3.2 Hardware Requirements:

- Processor : Intel i5 10th gen or above
- Monitor Screen : 14’’ or above
- RAM : 6 GB and above
- Memory : 512 GB SSD / 1TB HDD and above
- GPU : Nvidia MX230 2GB card and above
- Ports: Ethernet port for network connection (can also use WiFi)

  Additionally, XGBoost Algorithm which is used in the project is optimized for parallel processing and hence having a multi-core CPU or GPU can significantly improve training times.

# CHAPTER – 4

# SYSTEM ARCHITECHTURE

## Chapter - 4

# SYSTEM ARCHITECTURE

In the below section, a brief overview of how the project is designed is given.

## 4.1 System Design

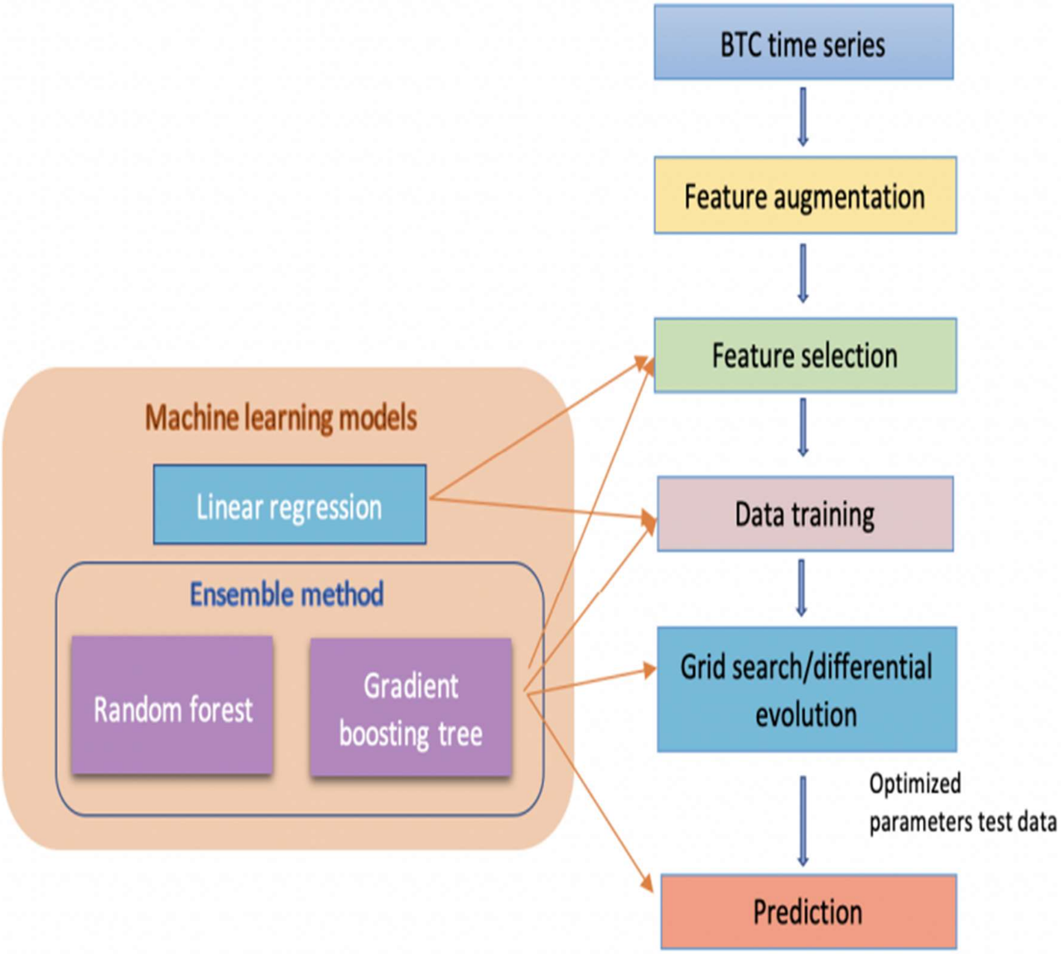

Fig 4.1: System Architecture diagram for Bitcoin price prediction system

The different Modules involved are,

1. **Data Collection and Downloading**: Obtain historical Bitcoin price data from reliable sources such as financial data APIs, cryptocurrency exchanges, or financial websites.
2. **Data Preprocessing:** Clean the data by handling missing values, removing outliers, and addressing any data quality issues. Convert the data into a format suitable for analysis, with date-time indexing if applicable.
3. **Scaling the Data by Max-Min Normalization:** Use Max-Min normalization to scale the Bitcoin price data to a specific range, typically between 0 and 1. This will ensure that all features have the same scale and prevent biases in the model. Splitting the Data into Training and Testing.
4. **Testing Sets**: Split the preprocessed data into training data (approximately 80% of the data) and testing data (approximately 20% of the data). This will allow you to train the model on a subset of the data and then evaluate its performance on unseen data.
5. **Feature Engineering:** Identify relevant features that could impact the price of Bitcoin, such as volume, previous price trends, market sentiment, etc. It's crucial to take into account various factors that could influence the price, and create features based on these factors.
6. **Model Design and Selection:** Train the XGBoost model on the training dataset to learn patterns and relationships within the data. Leveraging parallel processing capabilities for efficient training, exploiting the algorithm's gradient boosting principles.
7. **Training and Validation:** Train the model using the training data and validate its performance using the testing data. This involves optimizing model hyperparameters and evaluating its predictive accuracy.
8. **Evaluation and Validation:** Use appropriate metrics to evaluate the model's performance, such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), or Mean Absolute Error (MAE). Ensure the model doesn't exhibit overfitting to the training data and generalizes well to new, unseen data.
9. **Prediction and Future Forecasting:** Utilize the trained model to make predictions on future Bitcoin prices, and implement strategies to assess the model's predictive capabilities in real-time.

# CHAPTER – 5

# IMPLEMENTATION

## Chapter - 5

# IMPLEMENTATION AND TESTING

## 5.1 Implementation:

```
import pandas as pd
import numpy as np
import math
import datetime as dt

import matplotlib.pyplot as plt
from itertools import cycle
import plotly.graph_objects as go
import plotly.express as px
from plotly.subplots import make_subplots
import seaborn as sns

from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, explained_variance_score,
r2_score
from sklearn.metrics import mean_poisson_deviance, mean_gamma_deviance, accuracy_score
from sklearn.preprocessing import MinMaxScaler

from plotly.offline import plot, iplot, init_notebook_mode
init_notebook_mode(connected=True)

data=pd.read_csv(r"C:\Users\Sanjana Vijay\Desktop\BTC-USD.csv")
data = data.rename(columns={'Date': 'date','Open':'open','High':'high','Low':'low','Close':'close',
                            'Adj Close':'adj_close','Volume':'volume'})

fig = go.Figure()

fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['open'],
    name='Stock Open Price',
    marker_color='crimson'
))
fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['close'],
    name='Stock Close Price',
    marker_color='lightsalmon'
))
```

```
fig.update_layout(barmode='group', xaxis_tickangle=-45,
                 title='Monthwise comparision between Stock open and close price')
fig.show()

))

fig.update_layout(barmode='group',
                 title=' Monthwise High and Low stock price')
fig.show()

names = cycle(['Stock Open Price','Stock Close Price','Stock High Price','Stock Low Price'])

fig = px.line(y_2014, x=y_2014.date, y=[y_2014['open'], y_2014['close'],
                                         y_2014['high'], y_2014['low']],
             labels={'Date': 'Date','value':'Stock value'})
fig.update_layout(title_text='Stock analysis chart', font_size=15,
font_color='black',legend_title_text='Stock Parameters')
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()

monthvise= y_2020.groupby(y_2020['date'].dt.strftime('%B'))[['open','close']].mean()
new_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
             'September', 'October', 'November', 'December']
monthvise = monthvise.reindex(new_order, axis=0)
monthvise

fig = go.Figure()

fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['open'],
    name='Stock Open Price',
    marker_color='crimson'
))
fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['close'],
    name='Stock Close Price',
    marker_color='lightsalmon'
))

fig.update_layout(barmode='group', xaxis_tickangle=-45,
                 title='Monthwise comparision between Stock open and close price')
```

```
fig.show()

y_2020.groupby(y_2020['date'].dt.strftime('%B'))['low'].min()
monthvise_high = y_2020.groupby(data['date'].dt.strftime('%B'))['high'].max()
monthvise_high = monthvise_high.reindex(new_order, axis=0)

monthvise_low = y_2020.groupby(y_2020['date'].dt.strftime('%B'))['low'].min()
monthvise_low = monthvise_low.reindex(new_order, axis=0)

fig = go.Figure()
fig.add_trace(go.Bar(
    x=monthvise_high.index,
    y=monthvise_high,
    name='Stock high Price',
    marker_color='rgb(0, 153, 204)'
))
fig.add_trace(go.Bar(
    x=monthvise_low.index,
    y=monthvise_low,
    name='Stock low Price',
    marker_color='rgb(255, 128, 0)'
))

fig.add_trace(go.Bar(
    x=monthvise_low.index,
    y=monthvise_low,
    name='Stock low Price',
    marker_color='rgb(255, 128, 0)'
))

fig.add_trace(go.Bar(
    x=monthvise_low.index,
    y=monthvise_low,
    name='Stock low Price',
    marker_color='rgb(255, 128, 0)'
))


fig.update_layout(barmode='group',
                  title=' Monthwise High and Low stock price')
fig.show()

names = cycle(['Stock Open Price','Stock Close Price','Stock High Price','Stock Low Price'])

fig = px.line(y_2020, x=y_2020.date, y=[y_2020['open'], y_2020['close'],
                                        y_2020['high'], y_2020['low']],
        labels={'Date': 'Date','value':'Stock value'})
fig.update_layout(title_text='Stock analysis chart', font_size=15,
font_color='black',legend_title_text='Stock Parameters')
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()

monthvise

fig = go.Figure()

fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['open'],
    name='Stock Open Price',

    marker_color='crimson'
))
fig.add_trace(go.Bar(
    x=monthvise.index,
    y=monthvise['close'],
    name='Stock Close Price',
    marker_color='lightsalmon'
))

fig.update_layout(barmode='group', xaxis_tickangle=-45,
                  title='Monthwise comparision between Stock open and close price')
fig.show()

y_2021.groupby(y_2021['date'].dt.strftime('%B'))['low'].min()
monthvise_high = y_2021.groupby(data['date'].dt.strftime('%B'))['high'].max()
monthvise_high = monthvise_high.reindex(new_order, axis=0)

monthvise_low = y_2021.groupby(y_2021['date'].dt.strftime('%B'))['low'].min()
monthvise_low = monthvise_low.reindex(new_order, axis=0)

fig = go.Figure()
fig.add_trace(go.Bar(
    x=monthvise_high.index,
    y=monthvise_high,
    name='Stock high Price',
    marker_color='rgb(0, 153, 204)'
))
fig.add_trace(go.Bar(
    x=monthvise_low.index,
    y=monthvise_low,
    name='Stock low Price',
    marker_color='rgb(255, 128, 0)'
))

fig.update_layout(barmode='group',
                  title=' Monthwise High and Low stock price')
fig.show()

names = cycle(['Stock Open Price','Stock Close Price','Stock High Price','Stock Low Price'])

fig.show()

y_overall=data
y_overall.drop(y_overall[['adj_close','volume']],axis=1)

monthvise= y_overall.groupby(y_overall['date'].dt.strftime('%B'))[['open','close']].mean()


new_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
             'September', 'October', 'November', 'December']
monthvise = monthvise.reindex(new_order, axis=0)

names = cycle(['Stock Open Price','Stock Close Price','Stock High Price','Stock Low Price'])

fig = px.line(y_overall, x=y_overall.date, y=[y_overall['open'], y_overall['close'],
                                          y_overall['high'], y_overall['low']],
             labels={'Date': 'Date','value':'Stock value'})
fig.update_layout(title_text='Stock analysis chart', font_size=15,
font_color='black',legend_title_text='Stock Parameters')
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()

closedf = data[['date','close']]
print("Shape of close dataframe:", closedf.shape)

closedf = closedf[closedf['date'] > '2020-09-13']
close_stock = closedf.copy()
print("Total data for prediction: ",closedf.shape[0])

del closedf['date']
scaler=MinMaxScaler(feature_range=(0,1))
closedf=scaler.fit_transform(np.array(closedf).reshape(-1,1))
print(closedf.shape)
```

```
training_size=int(len(closedf)*0.70)
test_size=len(closedf)-training_size
train_data,test_data=closedf[0:training_size,:],closedf[training_size:len(closedf),:1]
print("train_data: ", train_data.shape)
print("test_data: ", test_data.shape)



fig, ax = plt.subplots(figsize=(15, 6))
sns.lineplot(x = close_stock['date'][:241], y = close_stock['close'][:241], color = 'black')
sns.lineplot(x = close_stock['date'][241:], y = close_stock['close'][241:], color = 'red')

# Formatting
ax.set_title('Train & Test data', fontsize = 20, loc='center', fontdict=dict(weight='bold'))
ax.set_xlabel('Date', fontsize = 16, fontdict=dict(weight='bold'))
ax.set_ylabel('Weekly Sales', fontsize = 16, fontdict=dict(weight='bold'))
plt.tick_params(axis='y', which='major', labelsize=16)
plt.tick_params(axis='x', which='major', labelsize=16)


plt.legend(loc='upper right' ,labels = ('train', 'test'))
plt.show()

def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return np.array(dataX), np.array(dataY)


time_step = 21
X_train, y_train = create_dataset(train_data, time_step)
X_test, y_test = create_dataset(test_data, time_step)

print("X_train: ", X_train.shape)
print("y_train: ", y_train.shape)
print("X_test: ", X_test.shape)
print("y_test", y_test.shape)

my_model = XGBRegressor(n_estimators=1000)
my_model.fit(X_train, y_train, verbose=False)

predictions = my_model.predict(X_test)
print("Mean Absolute Error - MAE : " + str(mean_absolute_error(y_test, predictions)))
print("Root Mean squared Error - RMSE : " + str(math.sqrt(mean_squared_error(y_test,
predictions))))


train_predict=my_model.predict(X_train)
test_predict=my_model.predict(X_test)

train_predict = train_predict.reshape(-1,1)
test_predict = test_predict.reshape(-1,1)

print("Train data prediction:", train_predict.shape)
print("Test data prediction:", test_predict.shape)

train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
original_ytrain = scaler.inverse_transform(y_train.reshape(-1,1))
original_ytest = scaler.inverse_transform(y_test.reshape(-1,1))


look_back=time_step
trainPredictPlot = np.empty_like(closedf)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
print("Train predicted data: ", trainPredictPlot.shape)

# shift test predictions for plotting
testPredictPlot = np.empty_like(closedf)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(closedf)-1, :] = test_predict
print("Test predicted data: ", testPredictPlot.shape)

names = cycle(['Original close price','Train predicted close price','Test predicted close price'])


plotdf = pd.DataFrame({'date': close_stock['date'],
                       'original_close': close_stock['close'],
                      'train_predicted_close': trainPredictPlot.reshape(1,-1)[0].tolist(),
                      'test_predicted_close': testPredictPlot.reshape(1,-1)[0].tolist()})

fig = px.line(plotdf,x=plotdf['date'], y=[plotdf['original_close'],plotdf['train_predicted_close'],
                                          plotdf['test_predicted_close']],
              labels={'value':'Close price','date': 'Date'})
fig.update_layout(title_text='Comparision between original close price vs predicted close price',
                  plot_bgcolor='white', font_size=15, font_color='black',legend_title_text='Close Price')
fig.for_each_trace(lambda t:  t.update(name = next(names)))

fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()

x_input=test_data[len(test_data)-time_step:].reshape(1,-1)
temp_input=list(x_input)
temp_input=temp_input[0].tolist()

from numpy import array

lst_output=[]
n_steps=time_step
i=0
pred_days = 10
while(i<pred_days):

    if(len(temp_input)>time_step):

        x_input=np.array(temp_input[1:])
        #print("{} day input {}".format(i,x_input))
        x_input=x_input.reshape(1,-1)

        yhat = my_model.predict(x_input)
        #print("{} day output {}".format(i,yhat))
        temp_input.extend(yhat.tolist())
        temp_input=temp_input[1:]

        lst_output.extend(yhat.tolist())
        i=i+1

    else:
        yhat = my_model.predict(x_input)

        temp_input.extend(yhat.tolist())
        lst_output.extend(yhat.tolist())

        i=i+1

print("Output of predicted next days: ", len(lst_output))

last_days=np.arange(1,time_step+1)
day_pred=np.arange(time_step+1,time_step+pred_days+1)
print(last_days)
print(day_pred)

temp_mat = np.empty((len(last_days)+pred_days+1,1))
temp_mat[:] = np.nan
temp_mat = temp_mat.reshape(1,-1).tolist()[0]
```

```
last_original_days_value = temp_mat
next_predicted_days_value = temp_mat

last_original_days_value[0:time_step+1] = scaler.inverse_transform(closedf[len(closedf)-
time_step:]).reshape(1,-1).tolist()[0]
next_predicted_days_value[time_step+1:] = scaler.inverse_transform(np.array(lst_output).reshape(-
1,1)).reshape(1,-1).tolist()[0]

new_pred_plot = pd.DataFrame({
   'last_original_days_value':last_original_days_value,
   'next_predicted_days_value':next_predicted_days_value
})

names = cycle(['Last 15 days close price','Predicted next 10 days close price'])

fig = px.line(new_pred_plot,x=new_pred_plot.index, y=[new_pred_plot['last_original_days_value'],
                                        new_pred_plot['next_predicted_days_value']],
             labels={'value': 'Close price','index': 'Timestamp'})
fig.update_layout(title_text='Compare last 15 days vs next 10 days',
                plot_bgcolor='white', font_size=15, font_color='black',legend_title_text='Close Price')
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()

my_model=closedf.tolist()
my_model.extend((np.array(lst_output).reshape(-1,1)).tolist())
my_model=scaler.inverse_transform(my_model).reshape(1,-1).tolist()[0]

names = cycle(['Close Price'])

fig = px.line(my_model,labels={'value': 'Close price','index': 'Timestamp'})
fig.update_layout(title_text='Plotting whole closing price with prediction',
                plot_bgcolor='white', font_size=15, font_color='black',legend_title_text='Stock')
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()


   }
```

# CHAPTER – 6

# RESULTS

## Chapter - 6

# RESULTS

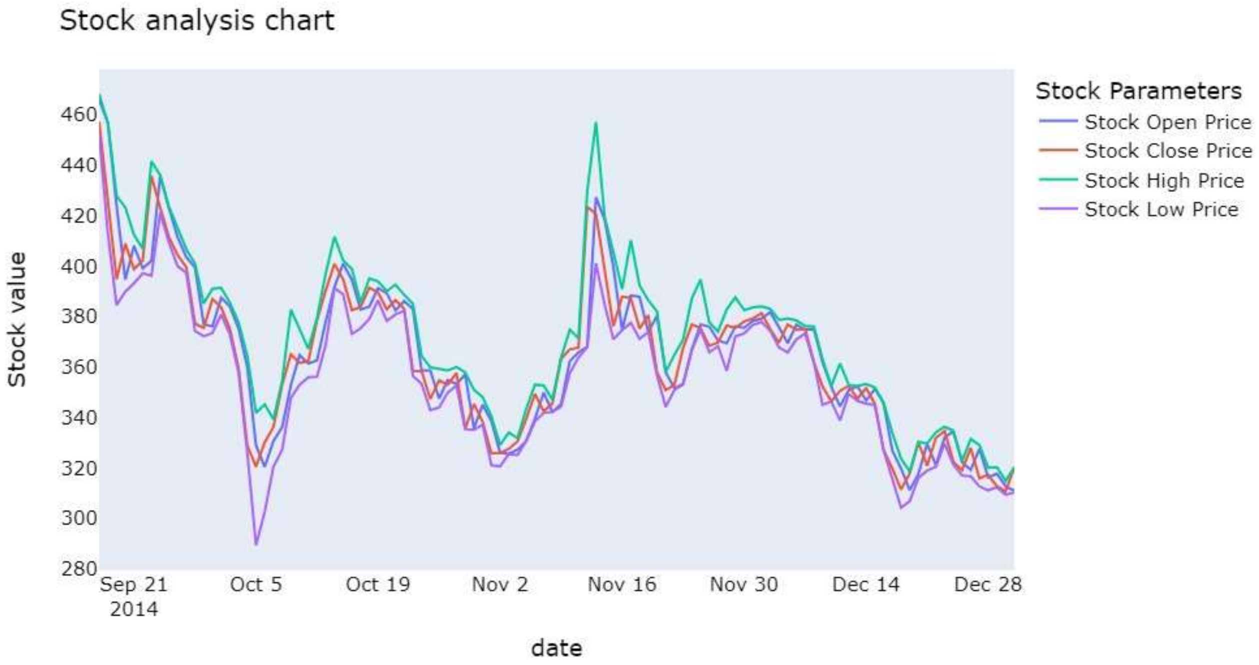

Fig 6.1 Stock analysis chart for the year 2014

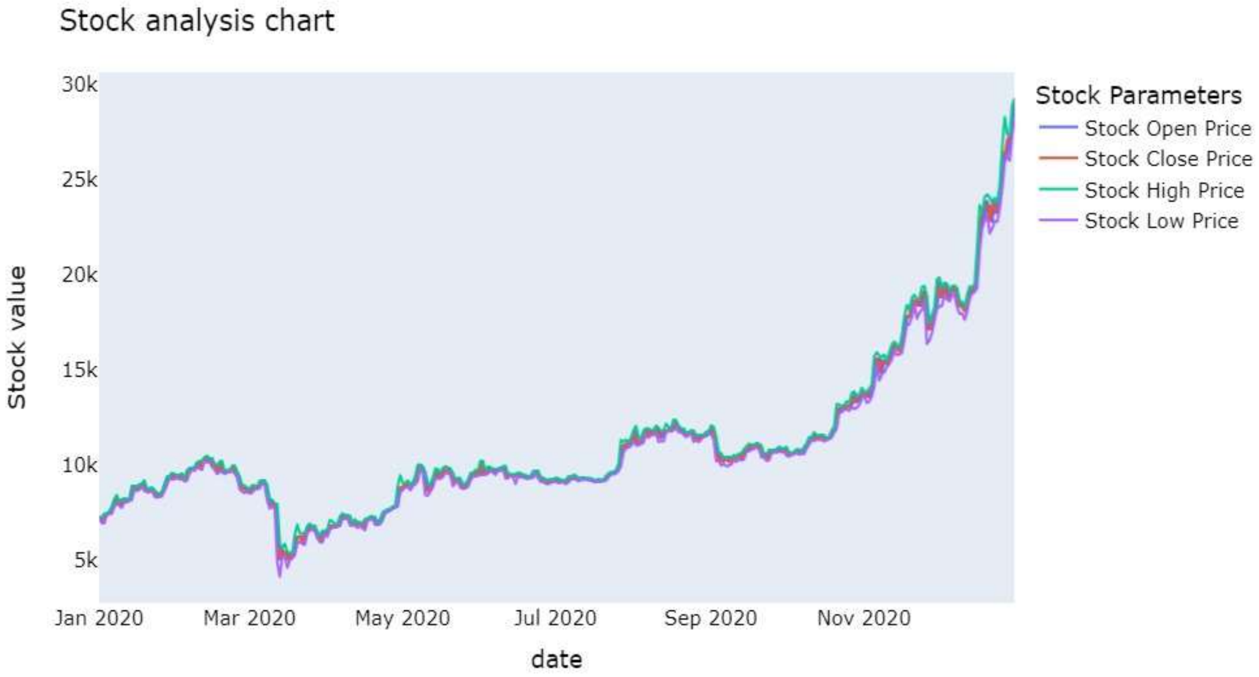

Fig 6.2 Stock analysis chart for the year 2020

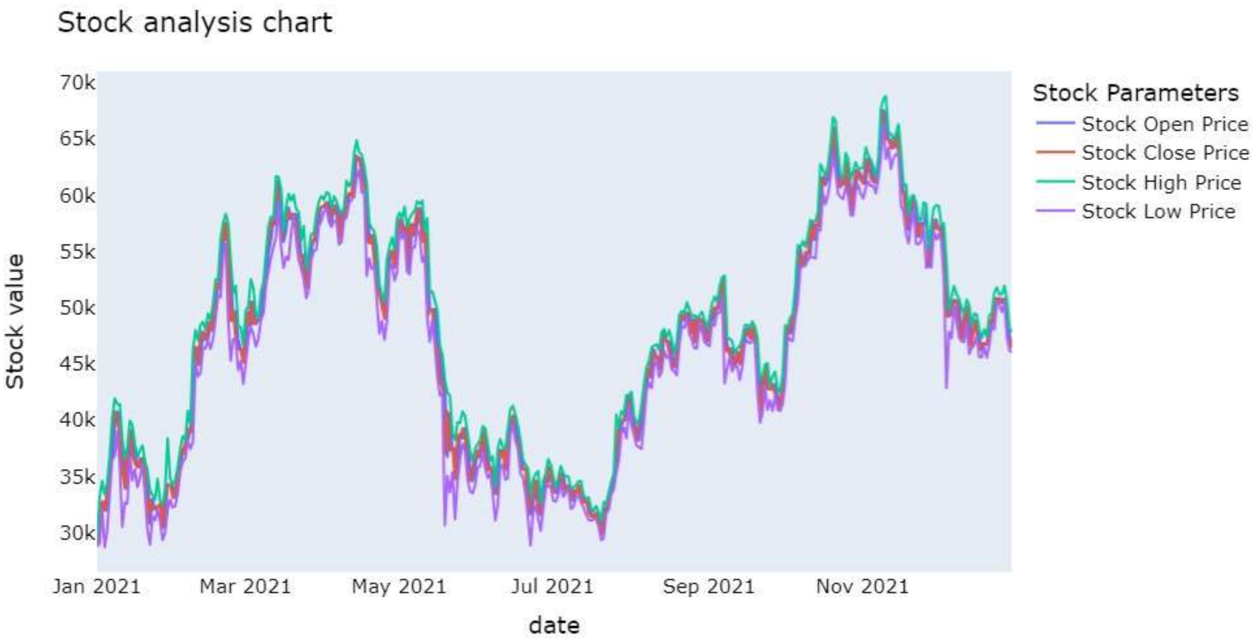

Fig 6.3 Stock Analysis chart for the year 2021

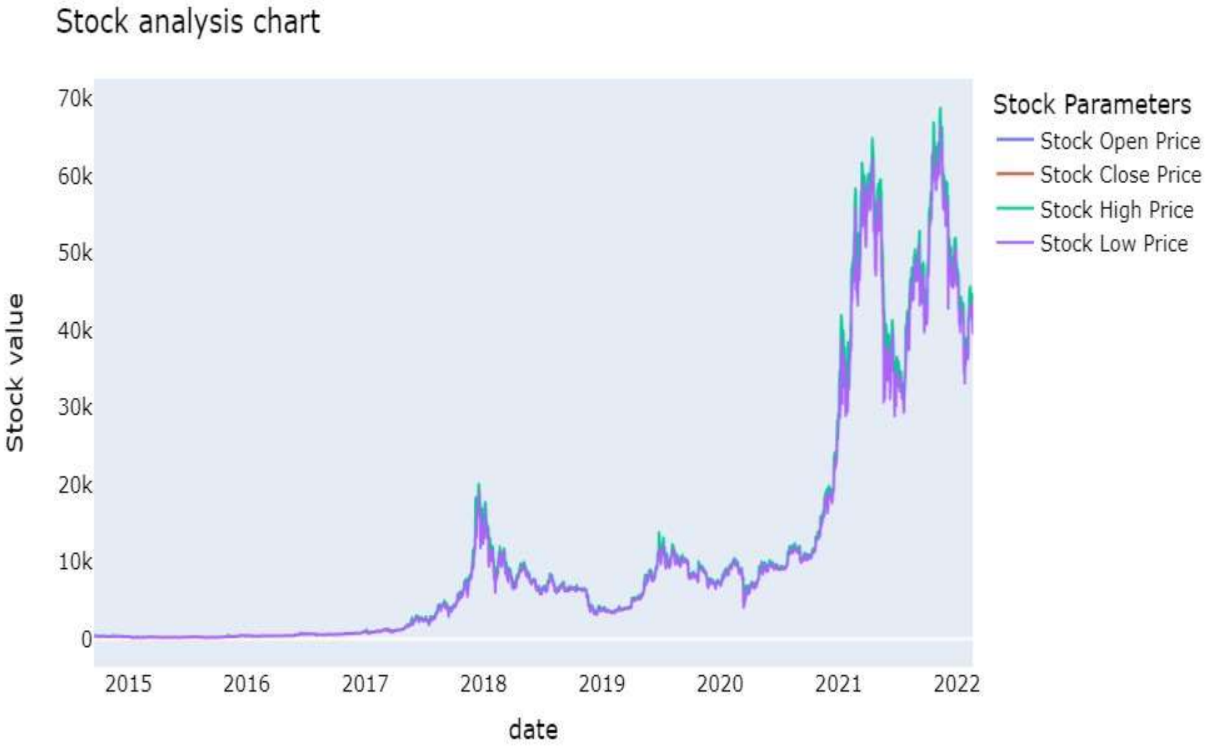

Fig 6.4 Overall analysis chart

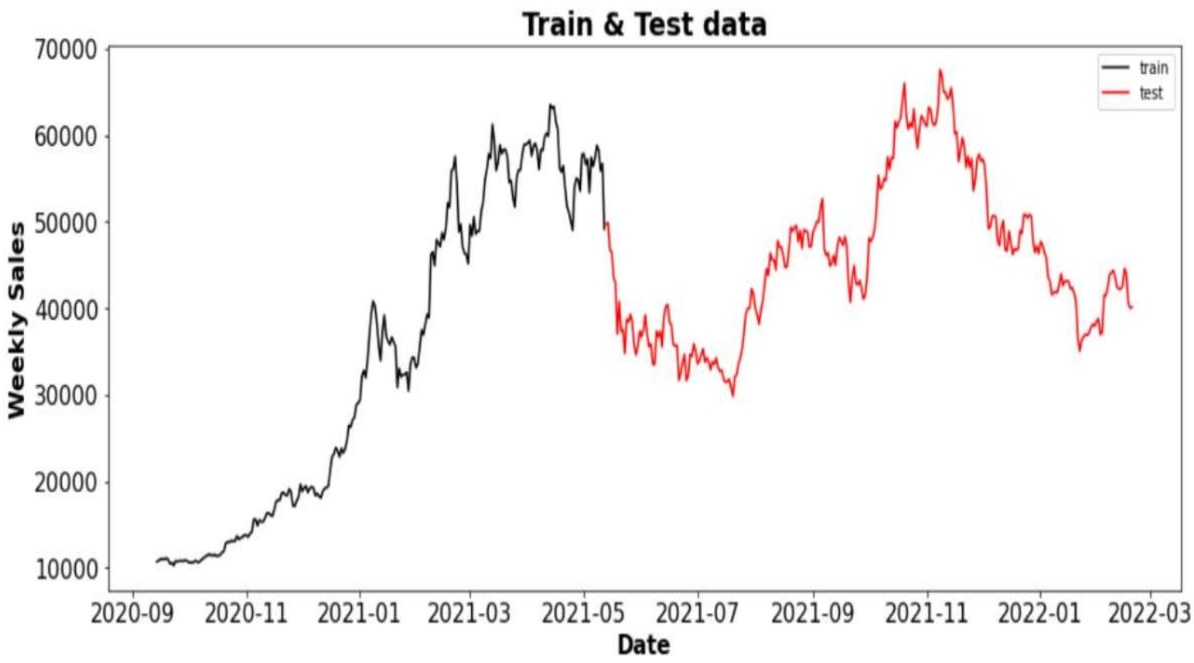

Fig 6.5 Splitting of testing and training data chart

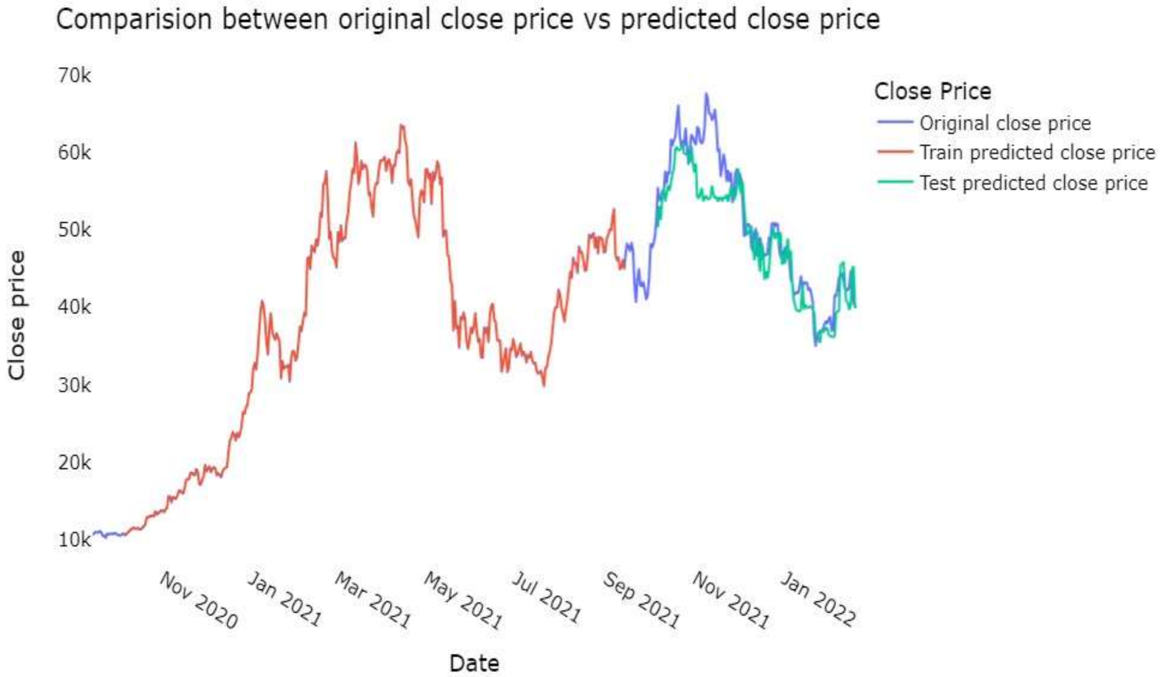

Fig 6.6 Comparison between original close price vs predicted close price

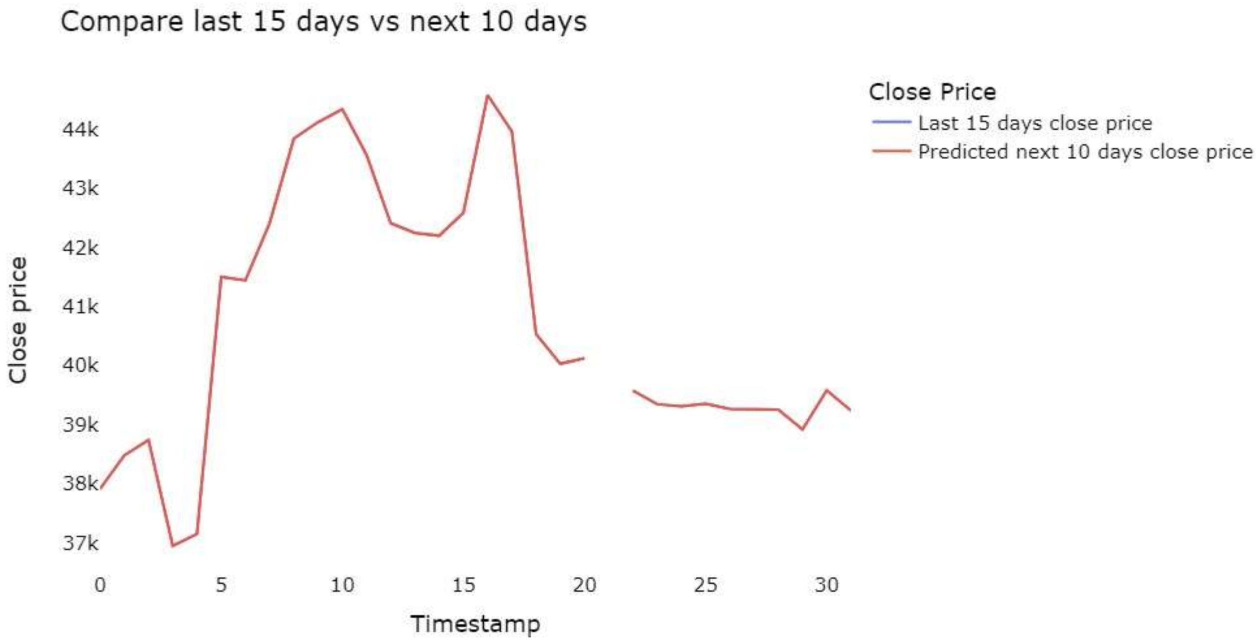

Fig 6.7 Previous 15 days close price vs predicted 10 days close price

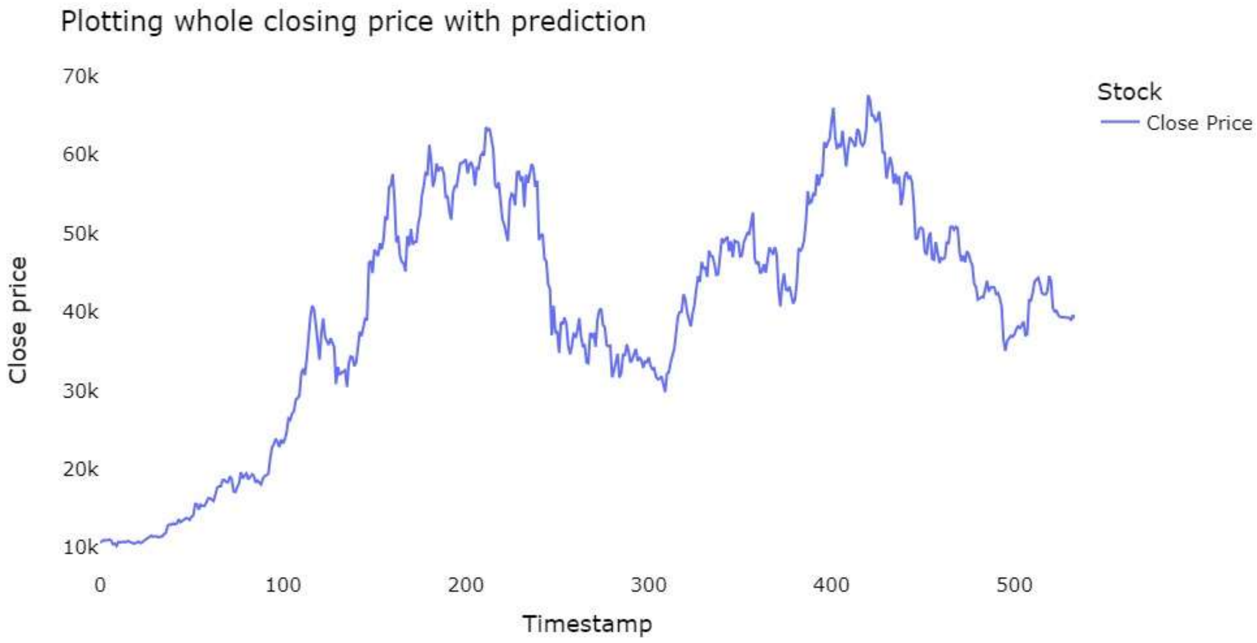

Fig 6.8 Whole closing price with prediction

# CONCLUSION

In the dynamic landscape of cryptocurrency markets, this project has endeavored to advance the field of Bitcoin price prediction through the integration of the powerful XGBoost algorithm. The journey began with meticulous data collection, preprocessing, and feature engineering, laying a robust foundation for model development. The utilization of XGBoost, with its fine-tuned hyperparameters, demonstrated its efficacy in capturing intricate patterns and adapting to the inherent volatility of the cryptocurrency realm.The project not only aimed to enhance predictive accuracy but also sought to contribute to the broader understanding of cryptocurrency markets.

As the project concludes, the results showcase the potential of machine learning, specifically XGBoost, as a formidable tool for accurate and adaptable Bitcoin price forecasting. The challenges posed by cryptocurrency market dynamics were met with a comprehensive approach, paving the way for future advancements in predictive modeling and a deeper comprehension of the forces driving cryptocurrency valuations. This project serves not only as a testament to the capabilities of machine learning in financial forecasting but also as a stepping stone for further research and innovation in the ever-evolving landscape of cryptocurrency analytics.

# REFERENCES

[1] Antonis Polemitis and Antonis C. Simotas (2015) Forecasting Financial Time Series with Machine Learning Algorithms, European Journal of Operational Research

[2] Yaya O.S. and Ogbonna A.O. (2016) Modelling and Forecasting Bitcoin Volatility Index, Financial Innovation

[3] Moazeni, S., & Bhowmik, T. (2018) Forecasting daily exchange rate using hybrid model of ARIMA and machine learning techniques, Journal of Ambient Intelligence and Humanized Computing

[4] Wang, Y., Ma, Y., Wang, J., & Liu, W. (2015) Forecasting the Price of Bitcoin Using Social Media Sentiment Analysis, IEEE International Conference on Data Mining Workshop (ICDMW)

[5] Garcia, D., & Schweitzer, F. (2015) Social signals and algorithmic trading of Bitcoin, Royal Society Open Science

[6] Kim, Y.B., Lee, J.H., & Ki, E.J. (2016) Predicting the price of Bitcoin using Bayesian regression, International Journal of Business and Management

[7] Shah, D., & Zhang, Q. (2019) Bitcoin Price Prediction and Trading Strategy using Machine Learning with Python, International Journal of Advanced Research in Computer Science

[8] https://www.kaggle.com/code/meetnagadia/xgboost-bitcoin-price-prediction

[9] https://github.com/topics/bitcoin-price-prediction

[10] https://www.geeksforgeeks.org/bitcoin-price-prediction-using-machine-learning-in-python/