BTC/INR DAILY PRICE PREDICTION

Gourab Pal

OBJECTIVE

Exploratory data analysis

Prediction using ML

Trading back testing

Objective	Package
Data fetching	requests
Data handling and manipulation	pandas, numpy
Visualization	mplfinance, Matplotlib
Split and tuning	GridSearchCV, TimeSeriesSplit from sklearn
ML algorithm	DecisionTreeRegressor, RandomForestRegressor, XGBRegressor from sklearn, xgboost
Metrics	root_means_squared_error, r2_score from sklearn

PYTHON PACKAGES

DATA OVERVIEW

1. Pair → BTC/INR
2. Chosen interval → 1m
3. Execution date → 07th October 2024
4. Fetched data type → json

Converted into pandas dataframe object for easy manipulation

```
df.dtypes
```

```
startTime     datetime64[ns]
open                   int64
high                   int64
low                    int64
close                  int64
endTime       datetime64[ns]
volume               float64
dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype         
---  ------     --------------  -----         
 0   startTime  1500 non-null   datetime64[ns]
 1   open       1500 non-null   int64         
 2   high       1500 non-null   int64         
 3   low        1500 non-null   int64         
 4   close      1500 non-null   int64         
 5   endTime    1500 non-null   datetime64[ns]
 6   volume     1500 non-null   float64       
dtypes: datetime64[ns](2), float64(1), int64(4)
memory usage: 82.2 KB
```

There are 1500 rows

```
df.isna().sum()
```

```
startTime    0
open         0
high         0
low          0
close        0
endTime      0
volume       0
dtype: int64
```

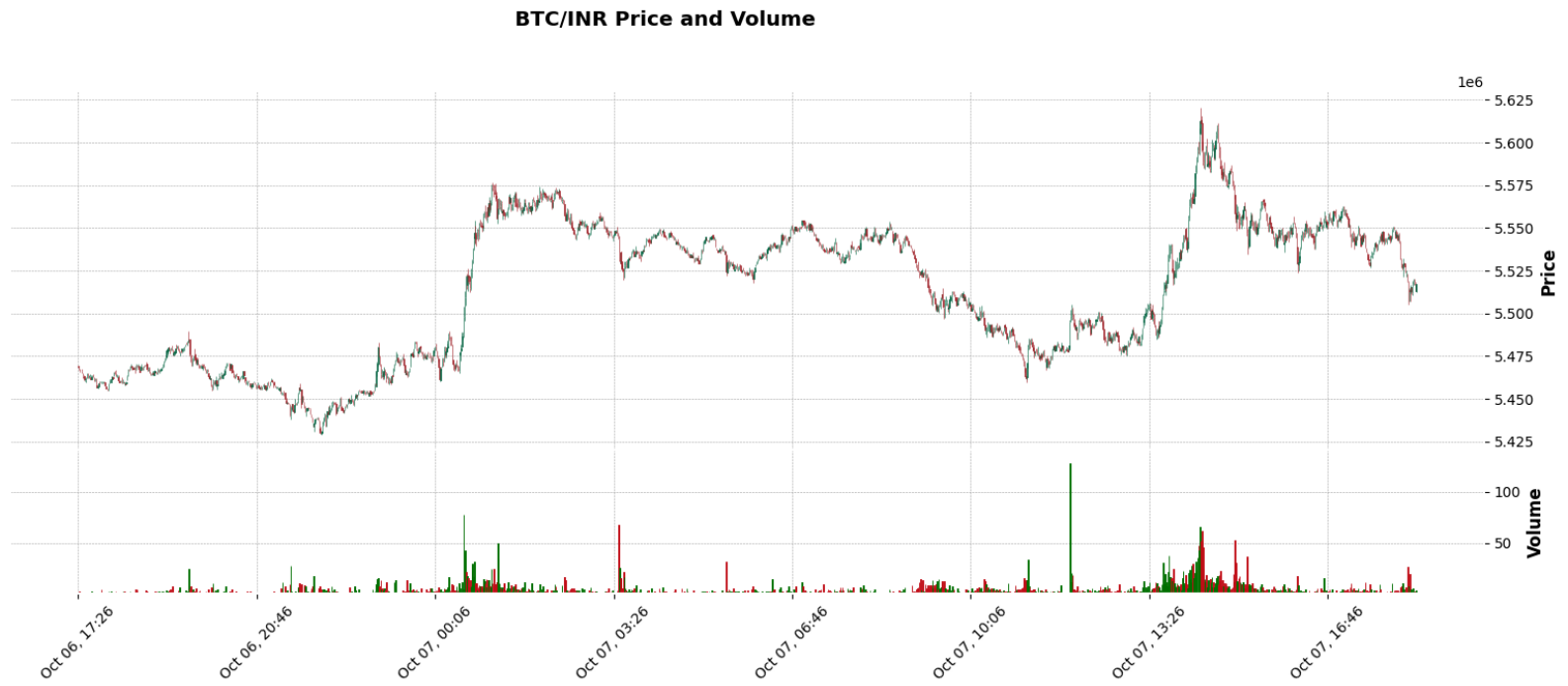
No Null values

```
df.head()
```

	startTime	open	high	low	close	endTime	volume
0	2024-10-06 17:26:00	5469150	5469211	5468452	5468452	2024-10-06 17:26:59.999	0.065
1	2024-10-06 17:27:00	5468452	5468452	5466542	5466542	2024-10-06 17:27:59.999	2.854
2	2024-10-06 17:28:00	5466368	5467467	5465714	5465714	2024-10-06 17:28:59.999	0.289
3	2024-10-06 17:29:00	5465356	5465356	5465356	5465356	2024-10-06 17:29:59.999	0.049
4	2024-10-06 17:30:00	5466115	5467414	5465147	5465147	2024-10-06 17:30:59.999	1.842

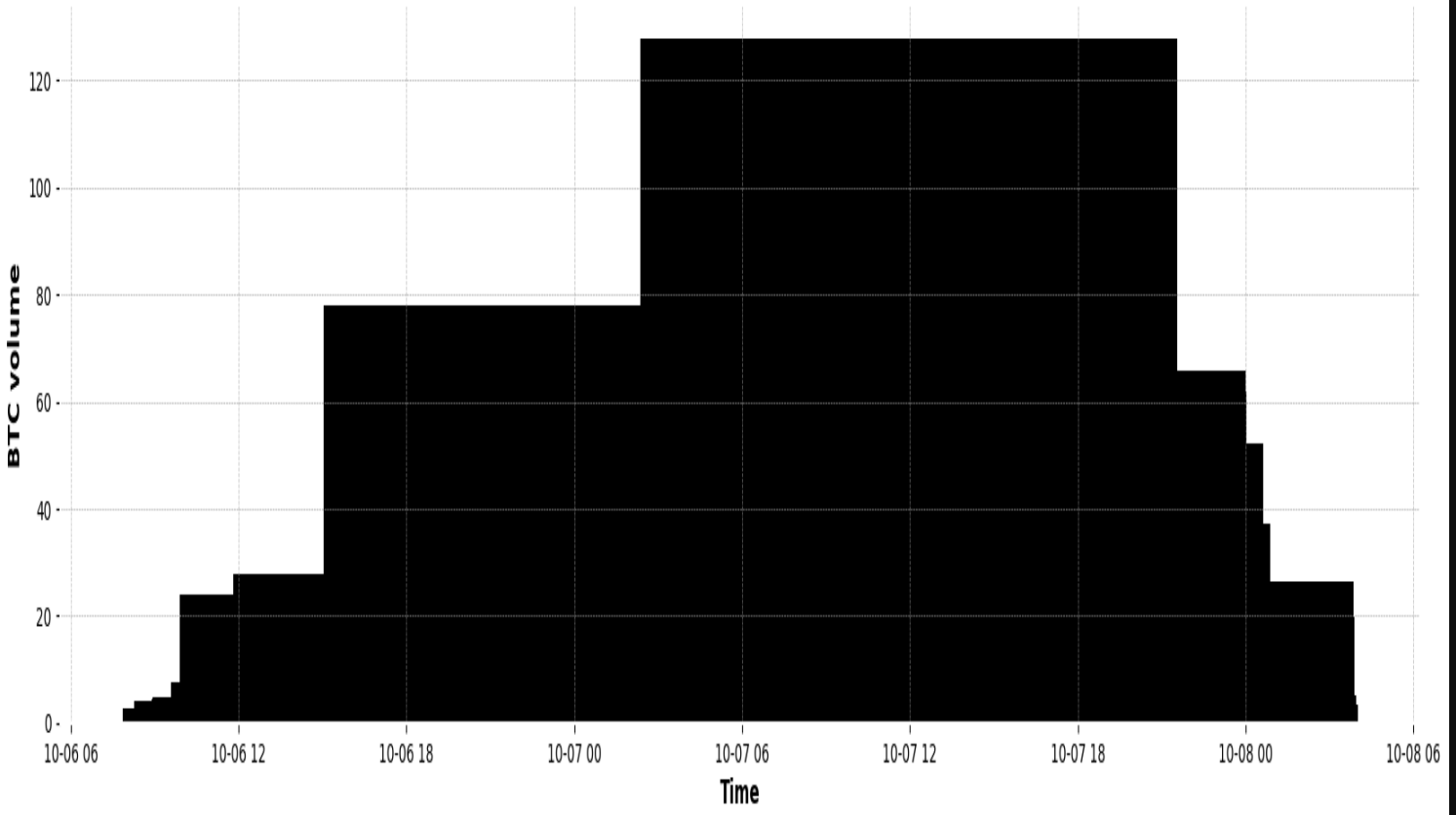
First 5 rows

DATA OVERVIEW



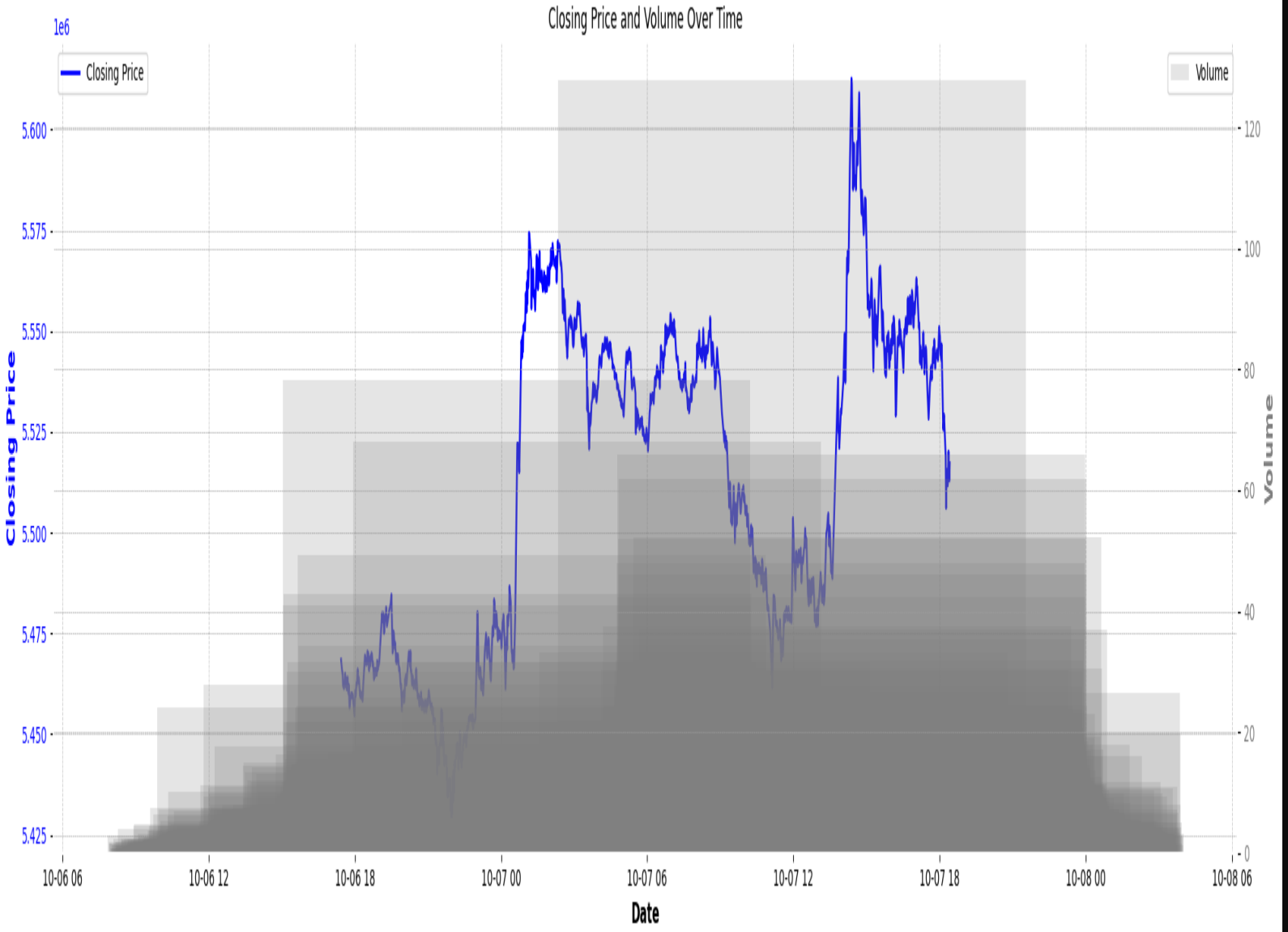
There is an overall increase of price in Oct 06, 17:26 to Oct 07, 16:45. When the volume of BTC was high and also there was an upward trend in price. This indicates bullish market. High volume means a large number of people are buying and selling indicating high interest and reliability in BTC. So, the sustainability and strength increase

EXPLORATORY DATA ANALYSIS



BTC volume traded over this day

EXPLORATORY DATA ANALYSIS



small steps of price increment is seen, however, a reversal may be started gradually.

EXPLORATORY DATA ANALYSIS

Our objective is to predict the "close" column in the dataset.

The "startTime" column is used as index of dataframe

When I have used the columns "low", "high", "open" and "volume" as features in ML models, the model was too simple to understand the hidden pattern

Then I have added 7 more features in the dataset utilizing the existing columns

WITHOUT FEATURE ENGINEERING

Feature Name	Description
RSI	Relative Strength Index → It measures the speed and change of price movement
close_lag1	Using previous time stamp value as predictor
stoch_k	Momentum indicator that compares closing price with a range of previous prices
EMA12	Exponential moving average of last 12 prices
EMA26	Exponential moving average of last 26 prices
MACD	Moving Average Convergence Divergence → Indicates momentum of market
MACD_signal	Exponential moving average of MACD for say past 9 MACD.

ENGINEERED FEATURES

SELECTION AND SPLITTING

We can not simple use train_test_split for this dataset as it is a time series data.

First 70% of data is used for training and validation. Remaining 30% is used for testing

```
# Feature selections

X = df[["RSI", "volume", "open", "high", "low", "close_lag1",  "stoch_k", "EMA12", "EMA26", "MACD", "MACD_Signal"]] # features
y = df[["close"]].shift(-1) # target column
y = y.dropna() # dropping the last NaN (due to shifting)
X = X.iloc[:-1] # dropiing the last row to make consistenst dimension
```

```
train_size = int(len(X) * 0.7)  # using 70% of data for training
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
```

I have used single untuned decision tree, tuned decision tree, tuned random forest and tuned xgboost regressor for price prediction

The 5 fold cross-validation is performed using the library TimeSeriesSplit method from sklearn. It takes care of the data sequence

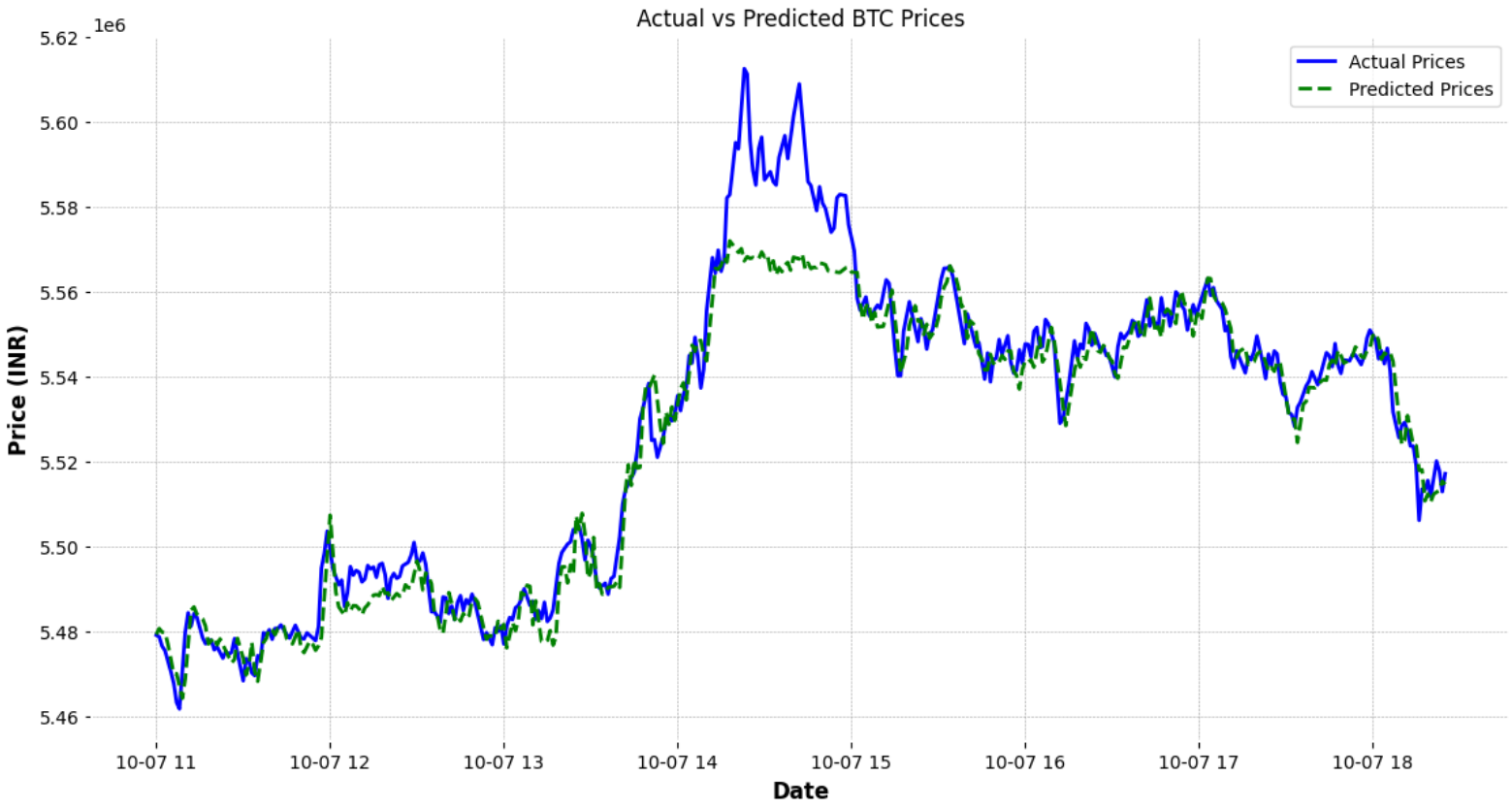
The GridSearchCV method tunes and validate and gives most optimized parameter that reduces mean_squared_error

HYPER PARAMETER TUNING

Model	RMS error (INR)	R^2 score
Un-tuned Decision Tree Regressor	10056	0.92
Tuned Decision Tree Regressor	10182	0.92
Tuned Random Forest Regressor	8698	0.94
Tuned XGBoost Regressor	8824	0.94

Both tuned random forest and tuned XGBoost regressor perform well. I have chosen XGBoost as final model

FORECASTING PERFORMANCE



FORECASTING

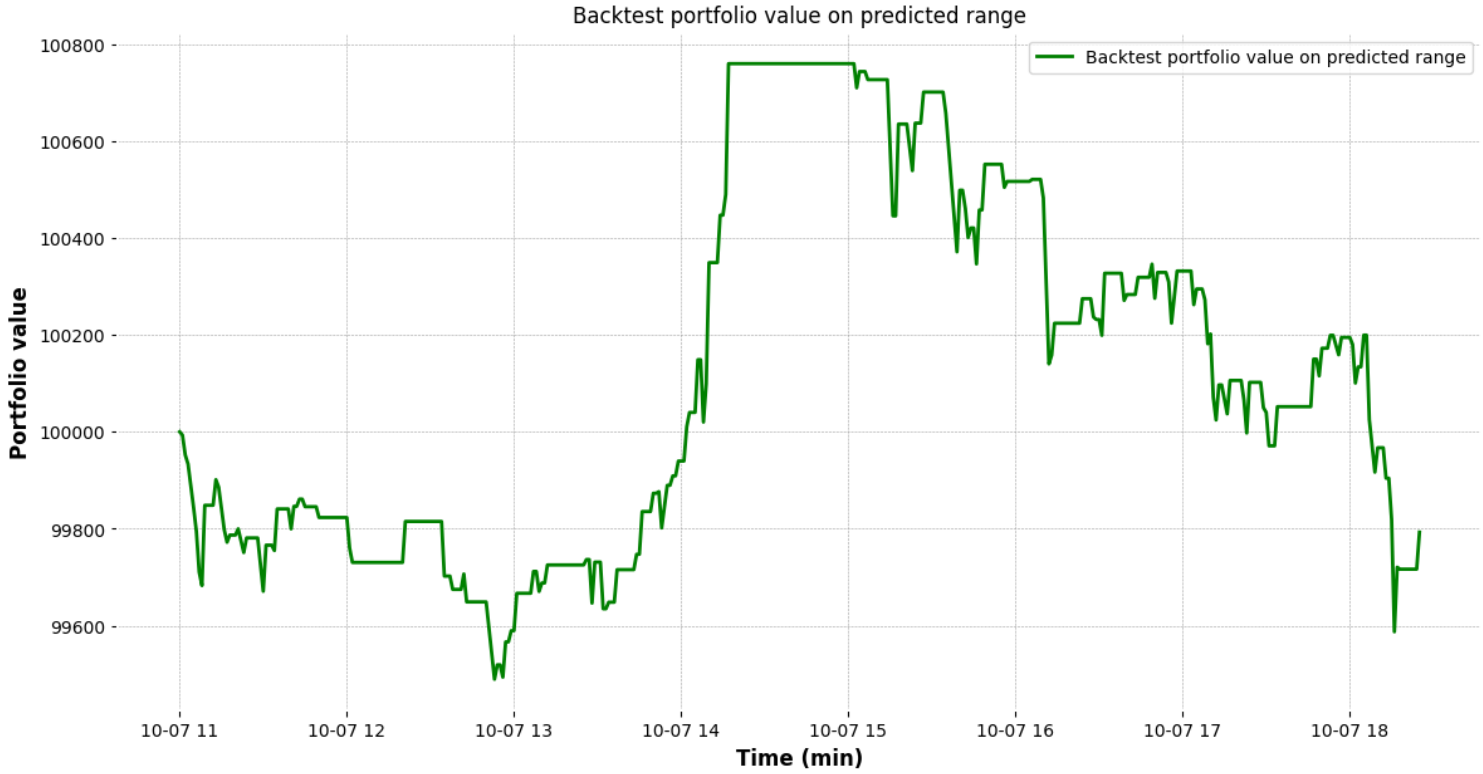
Tuned XGBoost model with RMS error INR 8824 with R^2 score 0.94

BACK TESTING

Trading strategy

1. Start with say 100000 INR
2. Start with 0 BTC holding
3. At every predicted BTC price, if predicted is more than actual, then buy BTC with all cash
4. Otherwise sell all BTC to get cash in INR
5. Track the portfolio at every transactions

BACK TESTING SIMULATION



THANK YOU

Email → gourab.pal.gpal@gmail.com

Contact → +91 7076444783

LinkedIn → www.linkedin.com/in/gourab-pal-0327801a4

GitHub → https://github.com/Gourab-Pal

Portfolio → https://sites.google.com/view/gourab-pal/home