TIME SERIES STOCK PRICE PREDICTION

COMPANY NAME : ICICI

- PROJECT GROUP NO. 6
- PRESENTED BY – NAWAJ SHAIKH
  NARESH JANGID
  HARSHAL THETE
  SAGAR NARKHEDE
  MARIUM QURESHI
  PRIYANKA SHINDE
- PRESENTED TO : MISS. AISHWARYA MATE

PROJECT MILDSTONE :

- Overview of project
- Data collection and preprocessing
- Exploratory data analysis
- Model selection and training
- Model evaluation
- Forecasting results
- Conclusion

OVERVIEW OF DATA

- ICICI Bank was originally promoted in 1994 by ICICI Limited, an Indian financial institution, and was its wholly-owned subsidiary.
- ICICI Bank's Board members include eminent individuals with a wealth of experience in international business, management consulting, banking and financial services.
- ICICI Bank offers a wide range of banking products and financial services to corporate and retail customers through a variety of delivery channels and through its group companies.
- Stock performance -
- The intrinsic value of one ICICIBANK stock under the Base Case scenario is 1 107.35 INR. Compared to the current market price of 1 037.4 INR, ICICI Bank Ltd is Undervalued by 6%.

DATA EXTRACTION

****We have extracted the stock prices for ICICIBANK from the website https://finance.yahoo.com/****

- Steps are as follows –

1. Visited the website https://finance.yahoo.com/

2. Select the ICICI Stocks by searching in the search bar for "ICICIBANK" and the BSE index.

3. The data shows up as "ICICI BANK (ICICIBANK.BO)". In order to get the historical data for stock prices for ICICI, we need to select the "Historical Data" tab.

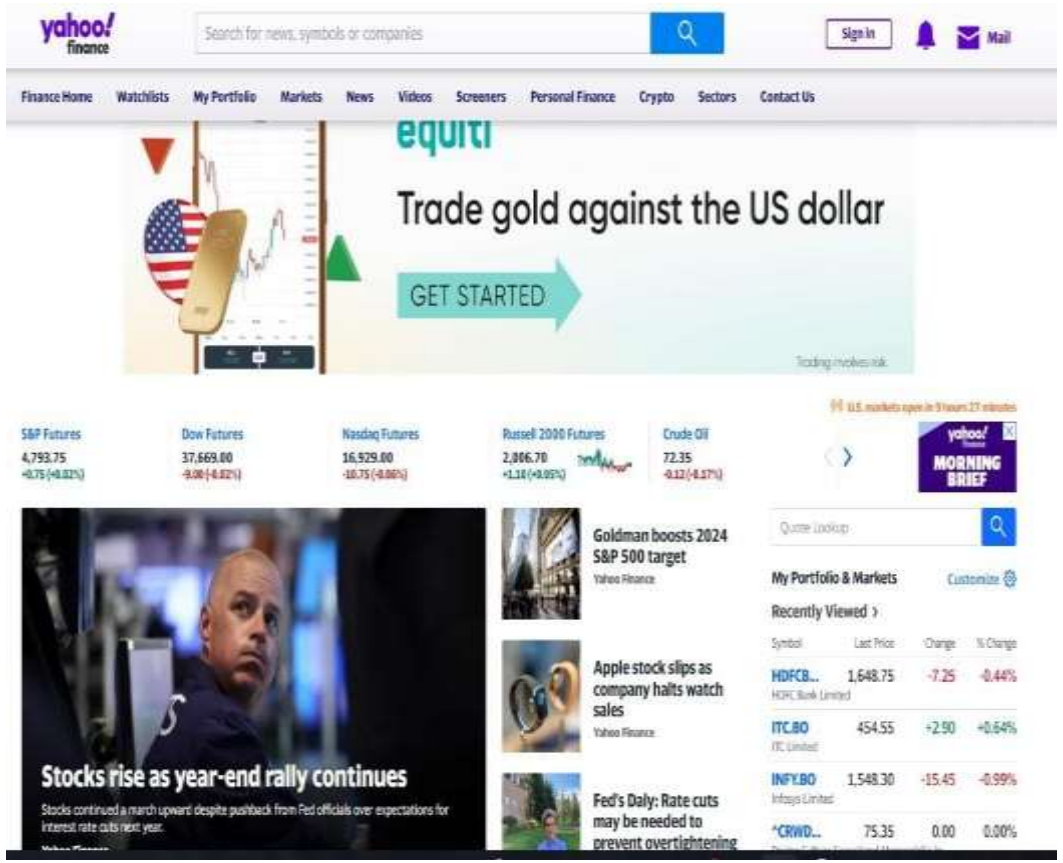
4. We need to now select the "Time Period", with the Frequency as "Daily". We can click on the start "Time Period" and click "Max". It gives the maximum Time Period for stock prices.

5. Click on the Apply button. This will give the Daily stock prices for the given maximum duration, in this case "Jan 03, 2000 - Nov 28, 2023".

6. Click on the Download button. This will give us the option to download the file in a location in the computer.

7. We have selected to download as a .csv file with the name as "ICICIBANK.csv"

DATA EXTRACTION



DATA COLLECTION AND PREPROCESSING

- **icicic records was extracted using YFinance library**.

The Stock price from **23-04-2002** to **01-12-2023** have been taken for the study. The dataset contains 6 attributes :

1.**Open:** price at which at stock started trading when the market opened on a particular day.

2. **High** : highest price at which a stock traded during the period

3. **Low**: lowest price of the period.

4. **Close**: price of an individual stock when the stock exchange closed market for the day. It represents the last buy sell order executed between 2 traders.

5. **Adj Close**: adjusted closing price is a calculation adjustment made to the stocks closing price. it is more complex and accurate than the closing price. The adjustment made to the closing price depicts the true price of the stock because the outside factors could have altered the true price.

6. **Volume**: total amount of activity during a period of time.

EXPLORATORY DATA ANALYSIS : (EDA)

- 1.Checking the basic detailing of the dataset.
- 2.Describing the data from dataset.
- 3. check the null values and drop it from the dataset.
- 4. checking the missing values form the dataset and remove it.
- 5. checking the duplicate values form dataset and removing it.
- 6.checking the outlier and replacing the outlier form the data.

EDA :

+ Code + Text

2023-12-01	936.000000	951.950012	933.250000	946.349976	946.349976	1843916.0	Dec	2023	2023

EXPLORATORY DATA ANALYSIS (EDA)

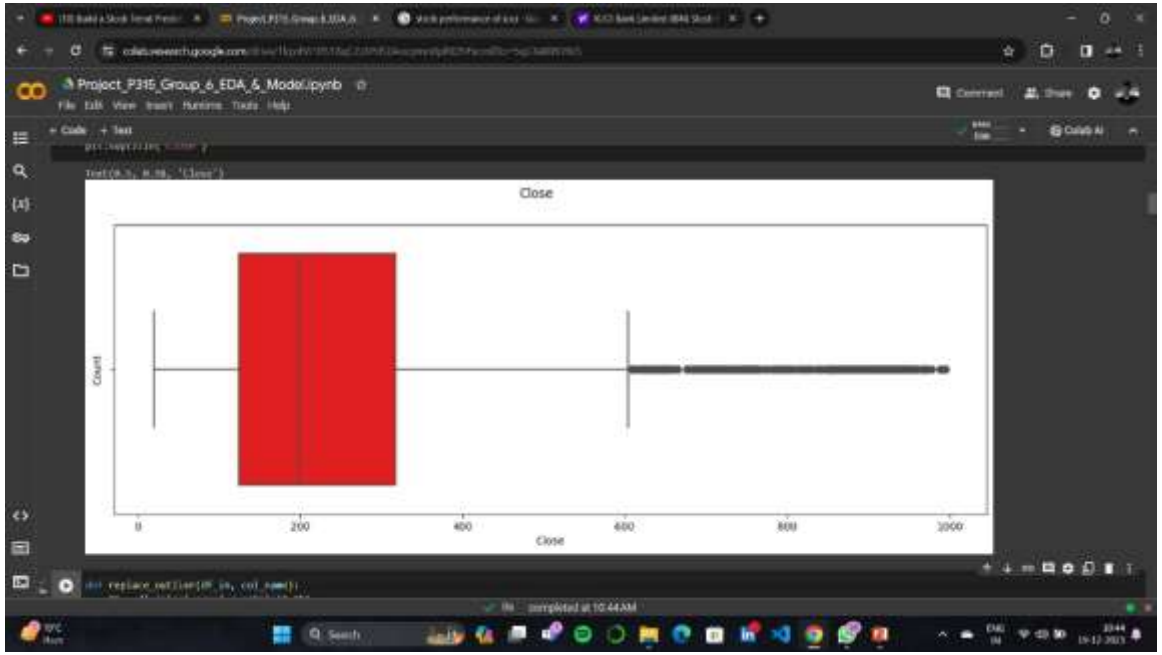
```
[79] Company_stock_prices.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5376 entries, 0 to 5375
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       5376 non-null   object
 1   Open       5341 non-null   float64
 2   High       5341 non-null   float64
 3   Low        5341 non-null   float64
 4   Close      5341 non-null   float64
 5   Adj Close  5341 non-null   float64
 6   Volume     5341 non-null   float64
dtypes: float64(6), object(1)
memory usage: 294.1+ KB
```

```
[80] Company_stock_prices.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	5341.000000	5341.000000	5341.000000	5341.000000	5341.000000	5.341000e+03
mean	271.613440	274.988165	267.923626	271.444620	239.727403	2.905065e+06
std	233.175899	235.077336	231.178346	233.193243	247.300133	9.383956e+06

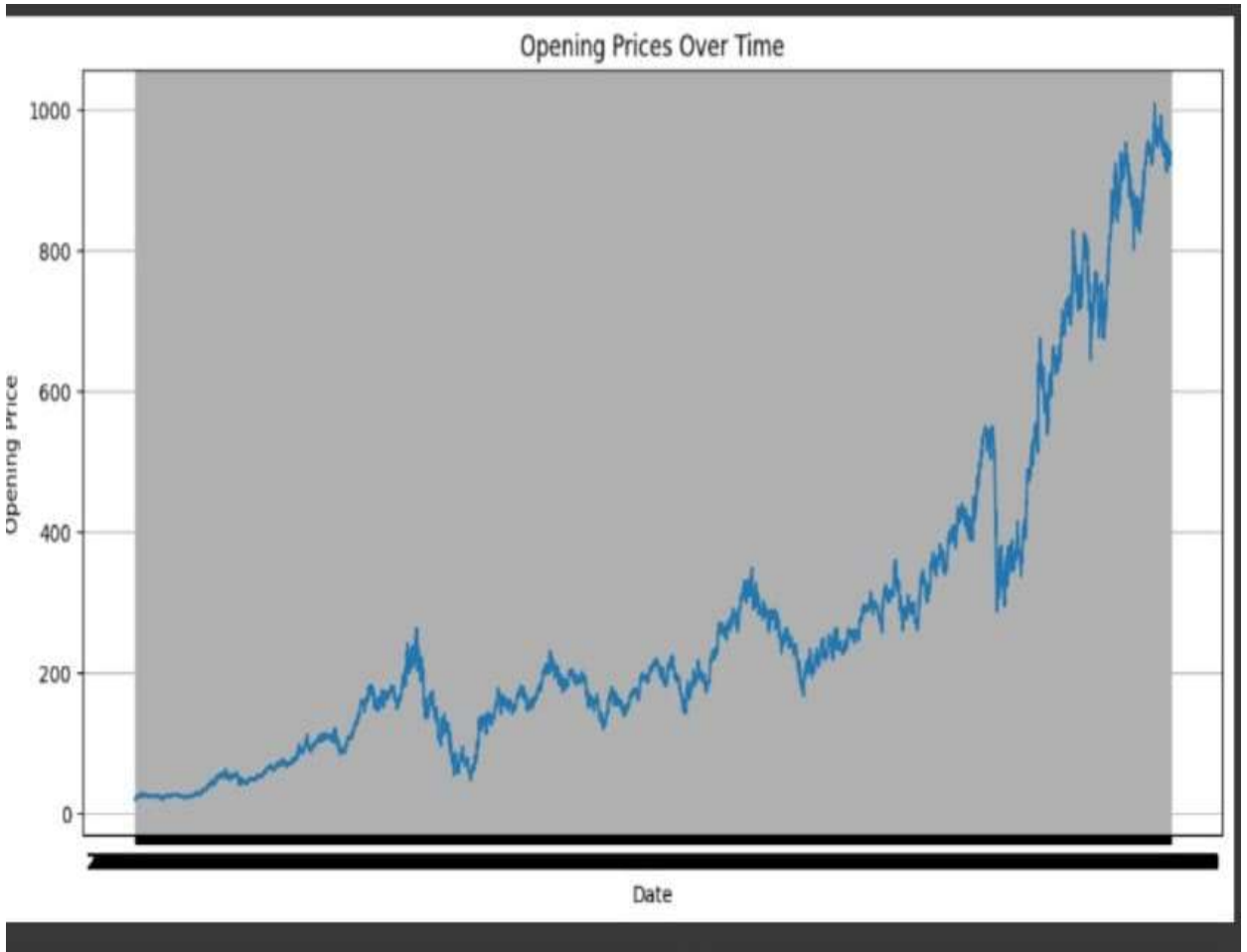
0s completed at 10:38 AM

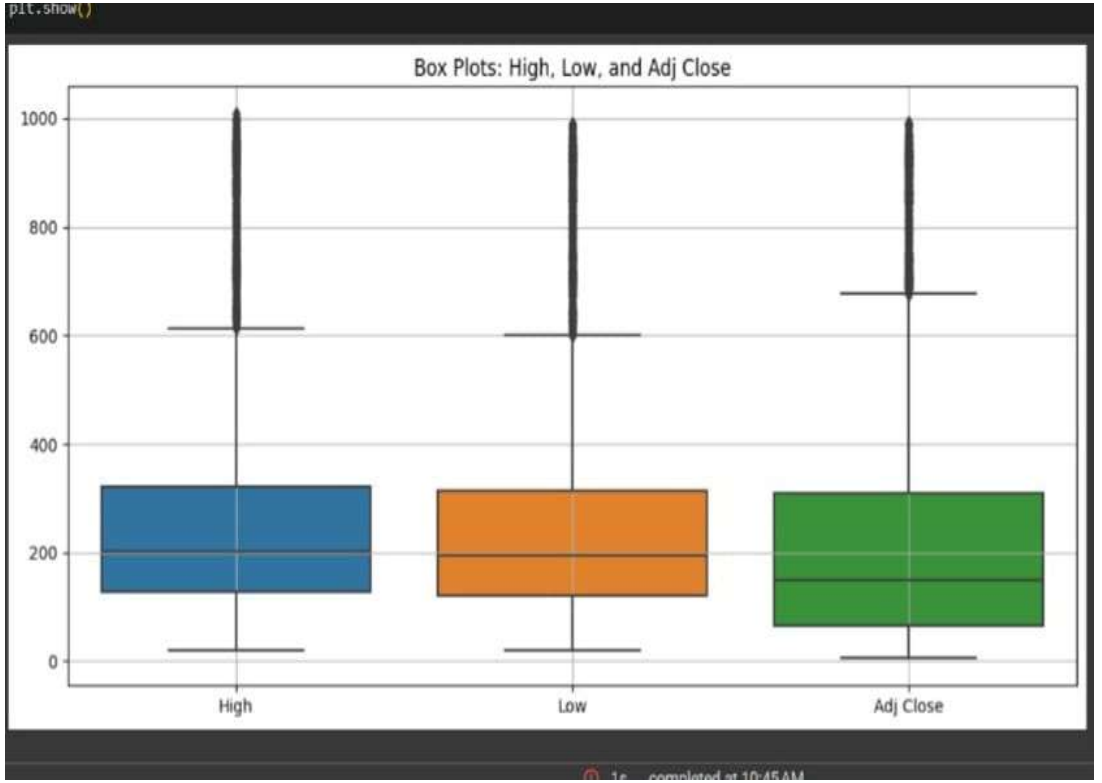


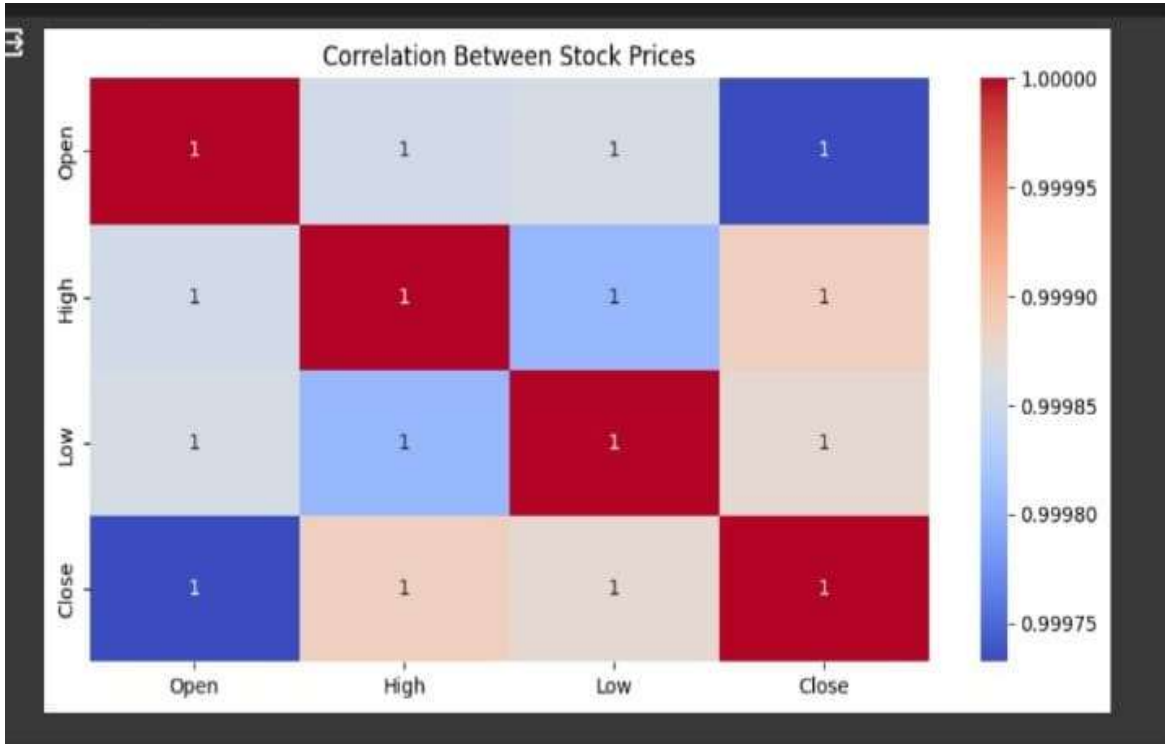
DATA VISUALIZATION :

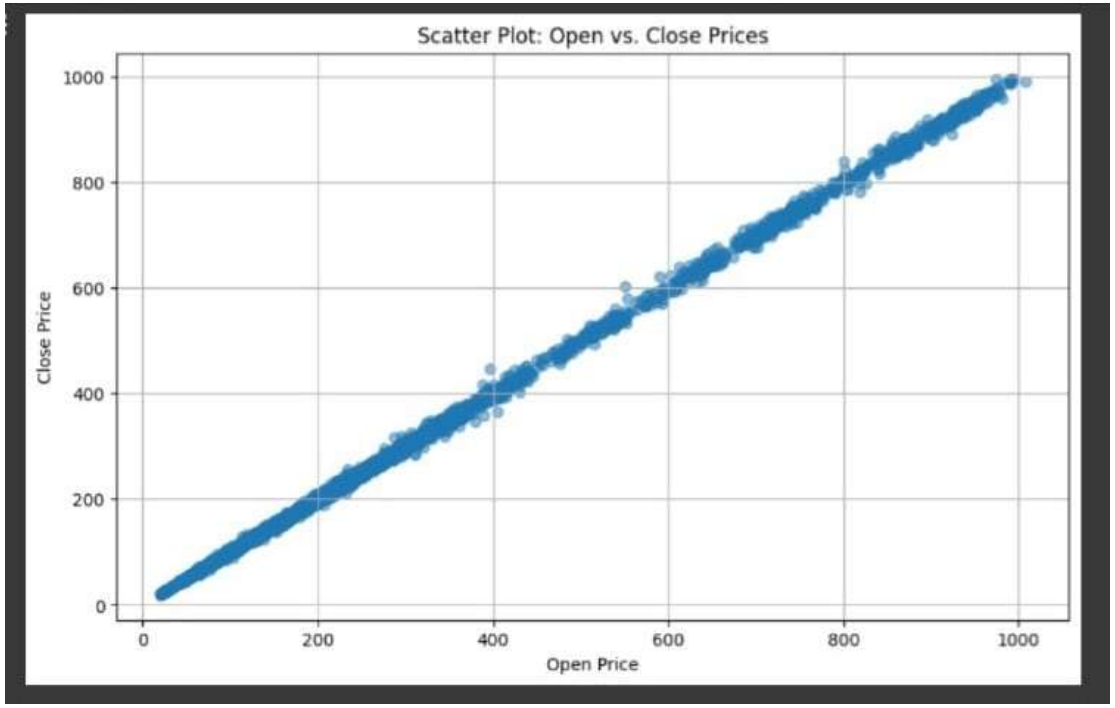
- 1. Line plot
- 2. Grid line for opening price over time
- 3. Scatter plot for opening and closing price
- 4. Box plot
- 5. Box plots for – (High , Low and Volume)
- 6. Heatmap-(correlation Between stock Prices)
- 7. Histogram – (For adj close price)
  - (Distribution of closing Stock Price)
  - (Histogram for High , Low And Volume)
- 8. Line Chart
- 9. Combination Chart

DATA VISUALIZATION :

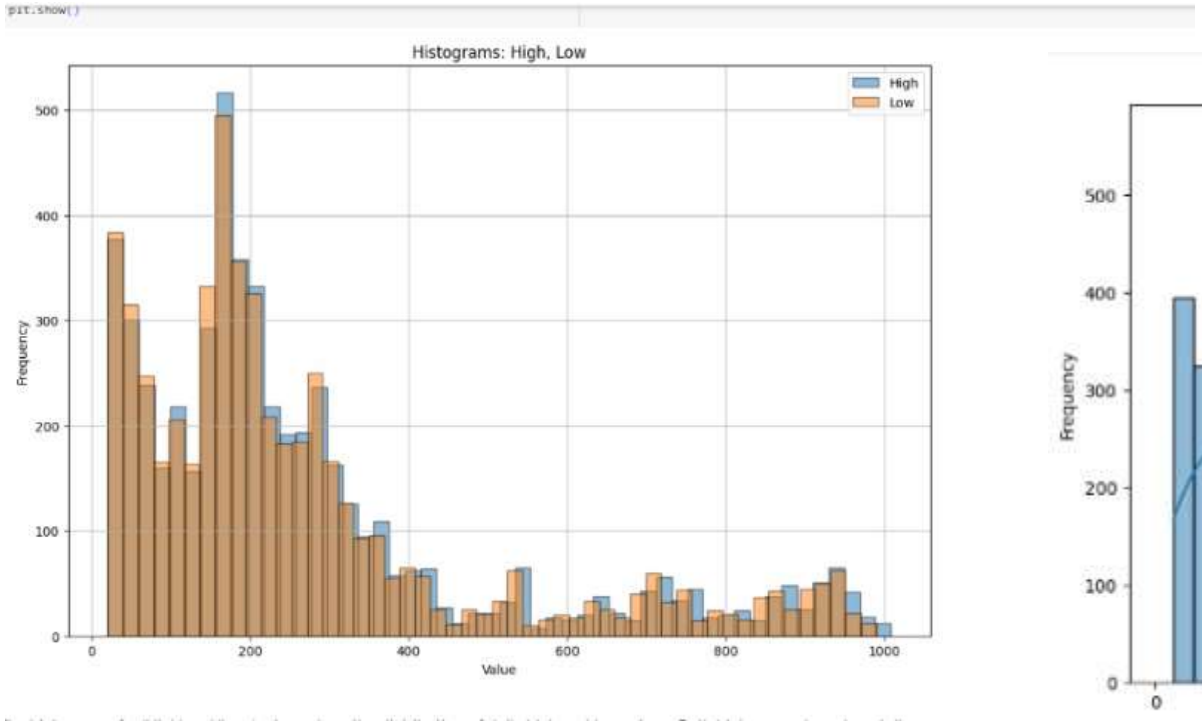


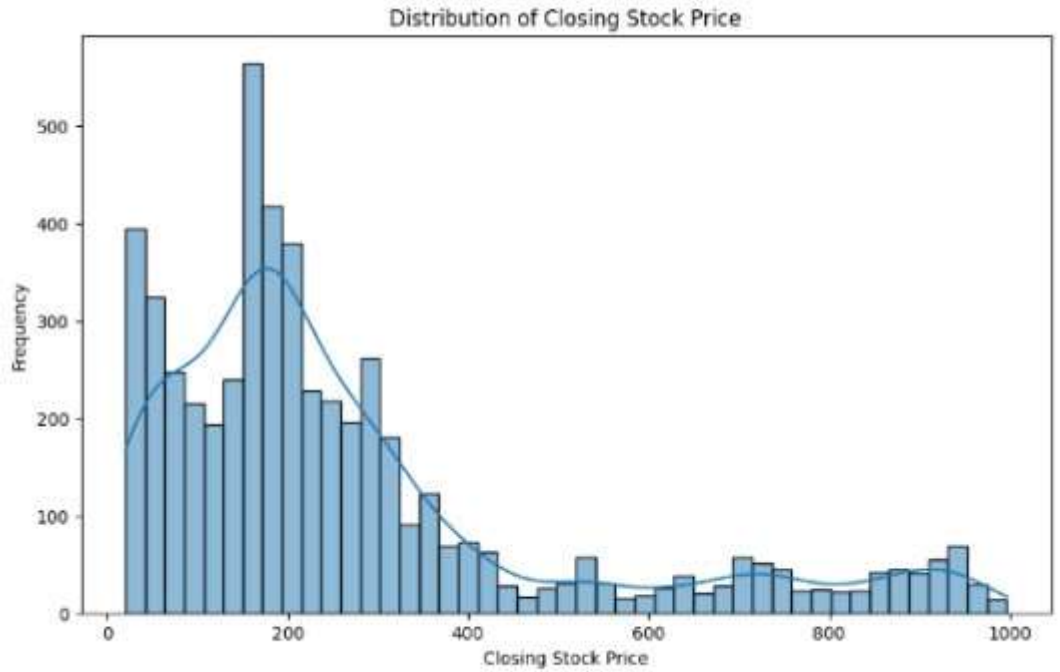



Correlation Between Stock Prices
Open
High
Low
Close
1
1.00000
0.99995
0.99990
0.99985
0.99980
0.99975

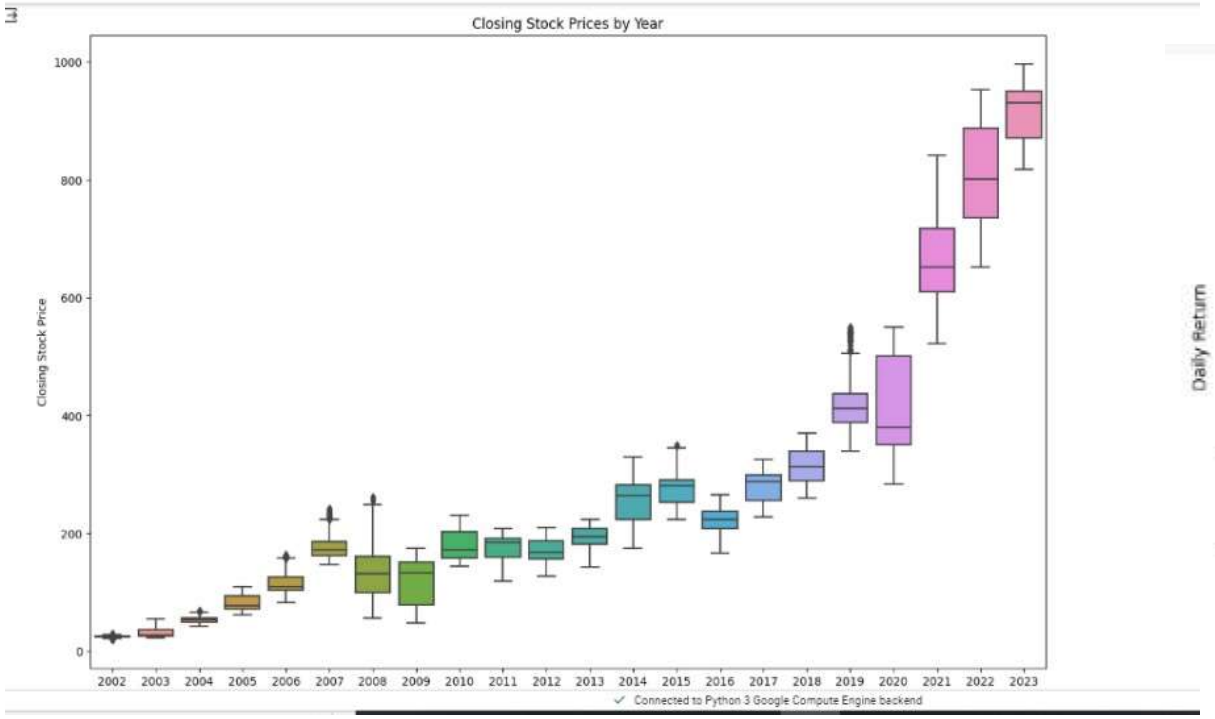

Scatter Plot: Open vs. Close Prices
Close Price
Open Price
0
200
400
600
800
1000

DATA VISULIZATION :





DATA VISULIZATION :



MODEL SELECTION AND TRAINING :

```
[125] tf.keras.backend.clear_session()
      model=Sequential()
      model.add(LSTM(32,return_sequences=True,input_shape=(time_step,1)))
      model.add(LSTM(32,return_sequences=True))
      model.add(LSTM(32))
      model.add(Dense(1))
      model.compile(loss='mean_squared_error',optimizer='adam')
```

```
[126] model.summary()

      Model: "sequential"
      _________________________________________________________________
       Layer (type)                Output Shape              Param #
      =================================================================
       lstm (LSTM)                 (None, 13, 32)            4352

       lstm_1 (LSTM)               (None, 13, 32)            8320

       lstm_2 (LSTM)               (None, 32)                8320

       dense (Dense)               (None, 1)                 33

      =================================================================
      Total params: 21025 (82.13 KB)
      Trainable params: 21025 (82.13 KB)
      Non-trainable params: 0 (0.00 Byte)
      _________________________________________________________________
```

Evaluation metrices RMSE and MAE

```
[54] import math
     from sklearn.metrics import mean_squared_error, mean_absolute_error, explained_variance_score, r2_score
     print("Train data RMSE: ", math.sqrt(mean_squared_error(original_ytrain,train_predict)))
     print("Train data MSE: ", mean_squared_error(original_ytrain,train_predict))
     print("Test data MAE: ", mean_absolute_error(original_ytrain,train_predict))
     print("-------------------------------------------------------------------------------------")
     print("Test data RMSE: ", math.sqrt(mean_squared_error(original_ytest,test_predict)))
     print("Test data MSE: ", mean_squared_error(original_ytest,test_predict))
     print("Test data MAE: ", mean_absolute_error(original_ytest,test_predict))
```

```
Train data RMSE:  4.093620612296552
Train data MSE:  16.757729717419192
Test data MAE:  3.071346856836441
-------------------------------------------------------------------------------
Test data RMSE:  36.62591924939322
Test data MSE:  1341.457960863073
Test data MAE:  27.334273100021875
```

Modeling results and predicition for next 30 days

```
forecast_df=pd.DataFrame({'Date': forecast_dates,'Close': predicted_values.flatten()})
print(forecast_df)
```

```
         Date       Close
0  2023-12-02  882.298706
1  2023-12-03  785.776001
2  2023-12-04  715.435486
3  2023-12-05  672.919556
4  2023-12-06  649.121948
5  2023-12-07  633.102478
6  2023-12-08  617.976013
7  2023-12-09  601.285095
8  2023-12-10  583.135071
9  2023-12-11  564.699219
10 2023-12-12  547.036865
11 2023-12-13  530.756226
12 2023-12-14  515.983521
13 2023-12-15  502.567322
14 2023-12-16  490.265015
15 2023-12-17  478.846527
16 2023-12-18  468.137054
17 2023-12-19  458.020874
18 2023-12-20  448.429413
19 2023-12-21  439.323181
20 2023-12-22  430.676331
21 2023-12-23  422.465759
22 2023-12-24  414.666840
23 2023-12-25  407.252411
24 2023-12-26  400.194336
25 2023-12-27  393.465393
26 2023-12-28  387.040161
27 2023-12-29  380.895905
28 2023-12-30  375.013092
```

Evaluation metrices RMSE and MAE

```
[54] import math
     from sklearn.metrics import mean_squared_error, mean_absolute_error, explained_variance_score, r2_score
     print("Train data RMSE: ", math.sqrt(mean_squared_error(original_ytrain,train_predict)))
     print("Train data MSE: ", mean_squared_error(original_ytrain,train_predict))
     print("Test data MAE: ", mean_absolute_error(original_ytrain,train_predict))
     print("-------------------------------------------------------------------------------------")
     print("Test data RMSE: ", math.sqrt(mean_squared_error(original_ytest,test_predict)))
     print("Test data MSE: ", mean_squared_error(original_ytest,test_predict))
     print("Test data MAE: ", mean_absolute_error(original_ytest,test_predict))
```

```
Train data RMSE:  4.093620612296552
Train data MSE:  16.757729717419192
Test data MAE:  3.071346856836441
-------------------------------------------------------------------------------------
Test data RMSE:  36.62591924939322
Test data MSE:  1341.457960863073
Test data MAE:  27.334273100021875
```

LSTM Model Structure

```
[ ] tf.keras.backend.clear_session()
    model=Sequential()
    model.add(LSTM(32,return_sequences=True,input_shape=(time_step,1)))
    model.add(LSTM(32,return_sequences=True))
    model.add(LSTM(32))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error',optimizer='adam')
```

```
model.summary()
```

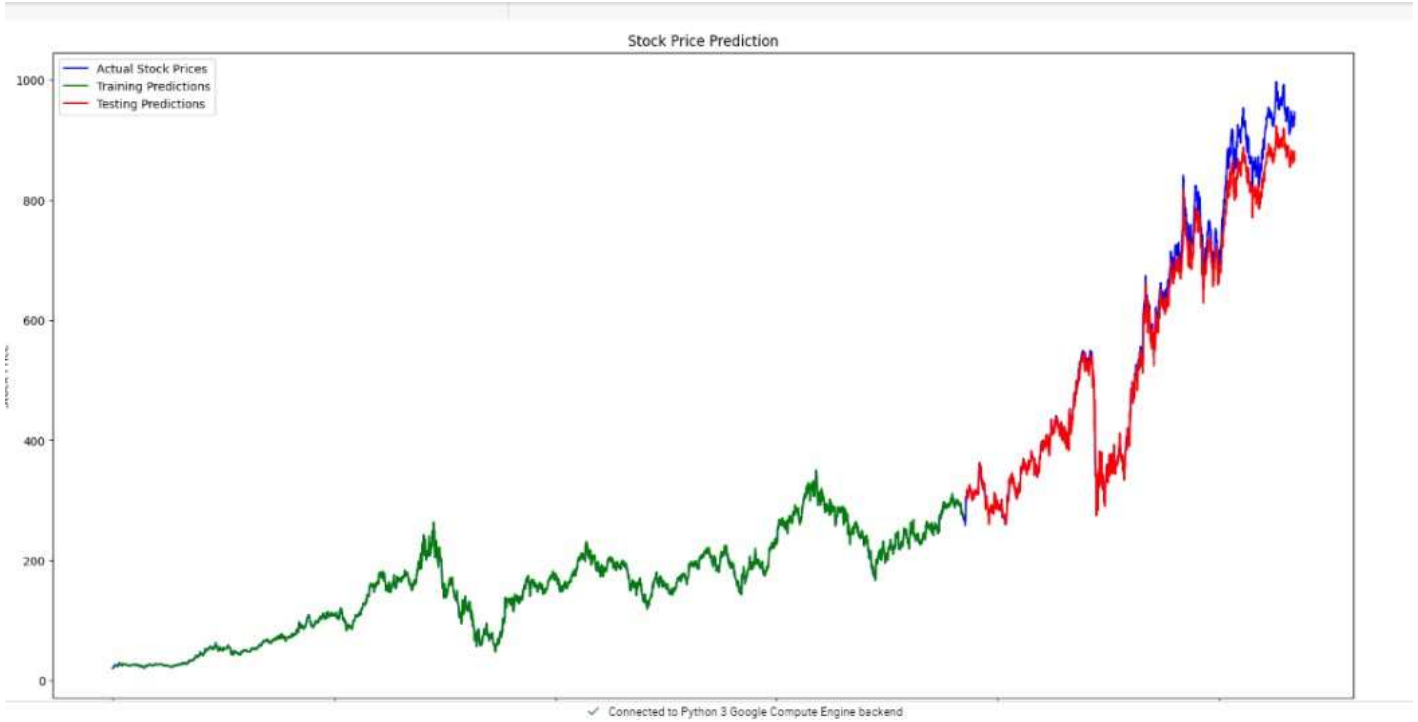
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 13, 32)            4352

 lstm_1 (LSTM)               (None, 13, 32)            8320

 lstm_2 (LSTM)               (None, 32)                8320

 dense (Dense)               (None, 1)                 33

=================================================================
Total params: 21025 (82.13 KB)
Trainable params: 21025 (82.13 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```



DEPLOYMENT:

Column Names:

	0
0	Date
1	Open
2	High
3	Low
4	Close
5	Adj Close
6	Volume

Data from 2002-2022

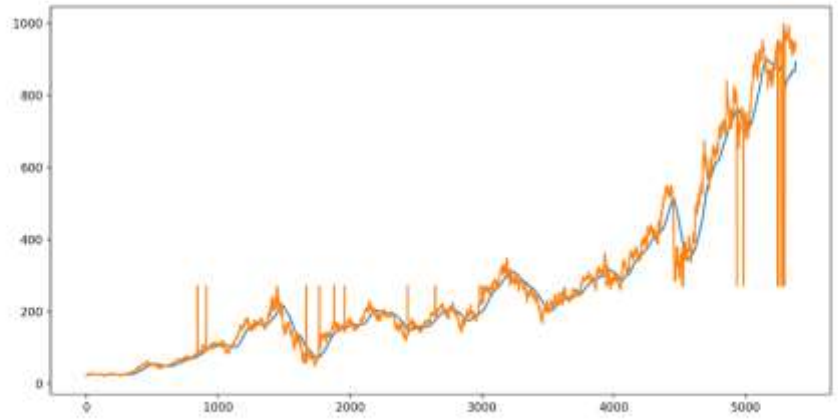
	Date	Open	High	Low	Close	Adj Close	Volume
5,366	2023-11-17 00:00:00	927.8	938.45	920.45	922	922	445,533
5,367	2023-11-20 00:00:00	922	926.55	917.45	921.45	921.45	307,256
5,368	2023-11-21 00:00:00	923.5	927.75	922	926.3	926.3	160,031
5,369	2023-11-22 00:00:00	921	925.3	914.8	922.6	922.6	186,447
5,370	2023-11-23 00:00:00	924.35	925	918.35	923.05	923.05	253,508
5,371	2023-11-24 00:00:00	922.25	930.6	920.35	929.15	929.15	906,155

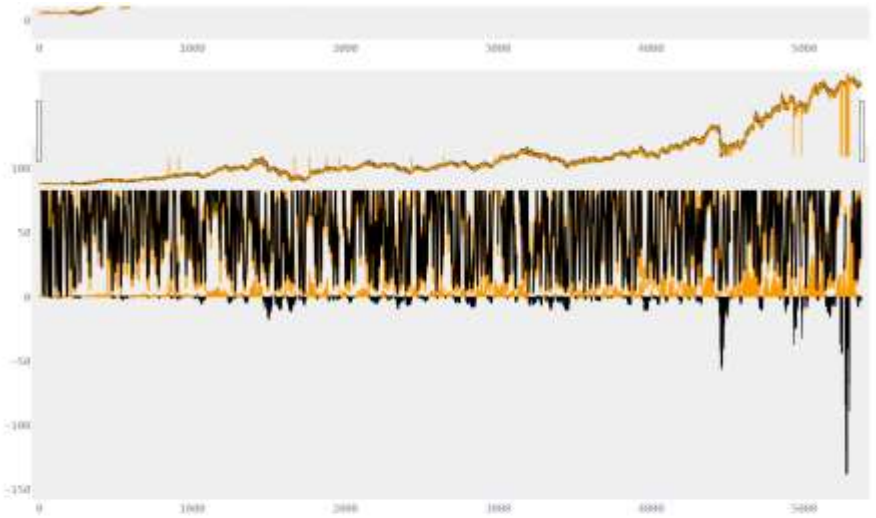
Technical Analysis

Choose Technical Analysis Type

- (●) Moving Average Chart
- ( ) Market trend
- ( ) Williams %R
- ( ) Stochastic Oscillator

Closing Price vs Time Chart with 100 MA





Enter Date in this format yyyy-mm-dd

2023/12/19

Predict

Technical Analysis

Choose Technical Analysis Type

- Moving Average Chart
- Market trend
- Williams %R
- Stochastic Oscillator

C: > Users > 92naw > OneDrive > Desktop > test.py > ...

```python
# Set the Streamlit app title and page icon
st.set_page_config(page_title="Stock Price Prediction", page_icon="")

# Custom HTML header
html_temp = """
    <div style="background-color:#8A9A5B;padding:10px;border-radius:10px">
    <h1 style="color:white;text-align:center;">Stock Price Prediction</h1>
    <h4 style="color:white;text-align:center;">COMPANY : ICICI BANK </h4>
    <h3 style="color:white;text-align:center;">Presented by: Naresh Jangid </h3>
    </div>
    """
st.markdown(html_temp, unsafe_allow_html=True)

# Load the stock price dataset
df = pd.read_csv(r'C:/Users/Naresh/Downloads/Stock predication Projects/ICICIBANK.BO.csv')

# Convert 'Date' column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Impute missing values in both 'Open' and 'Close' columns
imputer = SimpleImputer(strategy='mean')
df[['Open', 'Close']] = imputer.fit_transform(df[['Open', 'Close']])

# Display the first 10 rows of the dataset
st.subheader("Stock Price Data:")
st.write(df.head(10))

# Check the column names
st.subheader("Column Names:")
st.write(df.columns)

st.subheader('Data from 2002-2022')
#df= df.reset_index()
st.write(df.tail(10))
st.write(df.describe())
```

CONCLUSION AND PURPOSE :

- In stock market prediction, the aim is to predict the future value of the financial stocks of a company. The recent trend in stock market prediction technologies is the use of machine learning which makes predictions based on the values of current stock market indices by training on their previous values.
- Stock price prediction using machine learning helps you discover the future value of company stock and other financial assets traded on an exchange. The entire idea of predicting stock prices is to gain significant profits.

THANK YOU