Mini Project : 02

Importing necessary libraries and loading dataset

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
df = pd.read_csv('/content/3IINFOTECH.BO.csv')
```

In [ ]:
```
df.head(2)
```

Out[ ]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2006-06-30	NaN	NaN	NaN	NaN	NaN	NaN
1	2007-07-05	NaN	NaN	NaN	NaN	NaN	NaN

Data Preprocessing

In [ ]:
```
df.isnull().sum()
```

Out[ ]:

	0
Date	0
Open	7
High	7
Low	7
Close	7
Adj Close	7
Volume	7

dtype: int64

In [ ]:
```
df.dropna(inplace=True)
```

In [ ]:
```
df.isnull().sum()
```

Out[ ]:

	0
Date	0
Open	0
High	0
Low	0
Close	0
Adj Close	0
Volume	0

dtype: int64

In [ ]:
```
df.head()
```

Out[ ]:

	Date	Open	High	Low	Close	Adj Close	Volume
7	2012-06-18	9.75	9.75	8.92	9.03	9.03	1174884.0
8	2012-06-19	9.00	9.10	8.55	8.57	8.57	967500.0
9	2012-06-20	8.70	8.85	8.56	8.71	8.71	930848.0
10	2012-06-21	8.80	8.92	8.67	8.79	8.79	818401.0
11	2012-06-22	8.65	9.64	8.63	9.34	9.34	1660981.0

In [ ]:
```
df.describe()
```

Out[ ]:

	Open	High	Low	Close	Adj Close	Volume
count	1956.000000	1956.000000	1956.000000	1956.000000	1956.000000	1.956000e+03
mean	5.338175	5.469223	5.166385	5.291897	5.291897	7.335598e+05
std	2.271950	2.327745	2.196920	2.251678	2.251678	1.246651e+06
min	1.210000	1.310000	1.150000	1.210000	1.210000	0.000000e+00
25%	3.807500	3.890000	3.700000	3.780000	3.780000	2.360320e+05
50%	5.000000	5.110000	4.855000	4.980000	4.980000	3.765840e+05
75%	6.932500	7.100000	6.720000	6.882500	6.882500	6.912700e+05
max	13.480000	13.590000	12.560000	13.220000	13.220000	1.829810e+07

In [ ]:
```
df.shape
```

Out[ ]:
```
(1956, 7)
```

In [ ]:
```
df.info()
```

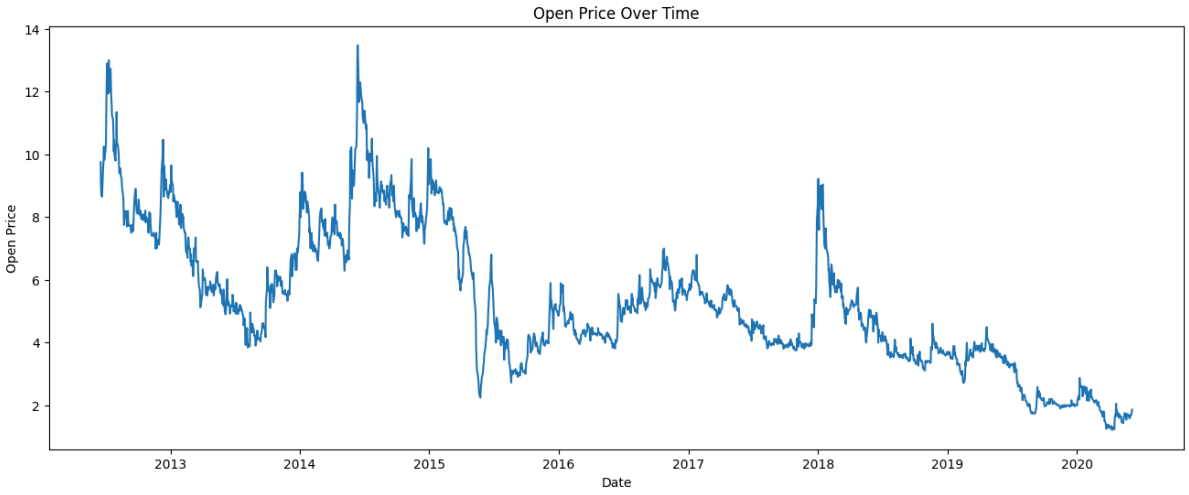
```
<class 'pandas.core.frame.DataFrame'>
Index: 1956 entries, 7 to 1962
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1956 non-null   object 
 1   Open       1956 non-null   float64
 2   High       1956 non-null   float64
 3   Low        1956 non-null   float64
 4   Close      1956 non-null   float64
 5   Adj Close  1956 non-null   float64
 6   Volume     1956 non-null   float64
dtypes: float64(6), object(1)
memory usage: 186.8+ KB
```

Exploratory Data Analysis

In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
df['Date'] = pd.to_datetime(df['Date'])

plt.figure(figsize=(16, 6))
sns.lineplot(data=df, x='Date', y='Open')
plt.title('Open Price Over Time')
plt.xlabel('Date')
plt.ylabel('Open Price')
plt.show()
```



In [ ]:
```
df = df.sort_values('Date')
```

In [ ]:
```
df.set_index('Date', inplace=True)
```

In [ ]:
```
df['SMA_20'] = df['Close'].rolling(window=20).mean()
df['SMA_50'] = df['Close'].rolling(window=50).mean()
```

In [ ]:
```
df['Per_Change'] = df['Close'].pct_change()
```

In [ ]:
```
df['Target'] = np.where(df['Per_Change'] > 0, 1, 0)
```

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
```

In [ ]:
```
from sklearn.preprocessing import StandardScaler
```

In [ ]:
```
sc = StandardScaler()
```

Splitting Data

In [ ]:
```
X = df[['SMA_20','SMA_50','Per_Change']]
```

In [ ]:
```
y = df['Target']
```

In [ ]:
```
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=42)
```

In [ ]:
```
x_train_scalled = sc.fit_transform(x_train)
x_test_scalled = sc.transform(x_test)
```

Model Training

In [ ]:
```
model = RandomForestClassifier(n_estimators=100,random_state=42)
```

In [ ]:
```
model.fit(x_train_scalled,y_train)
```

Out[ ]:
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

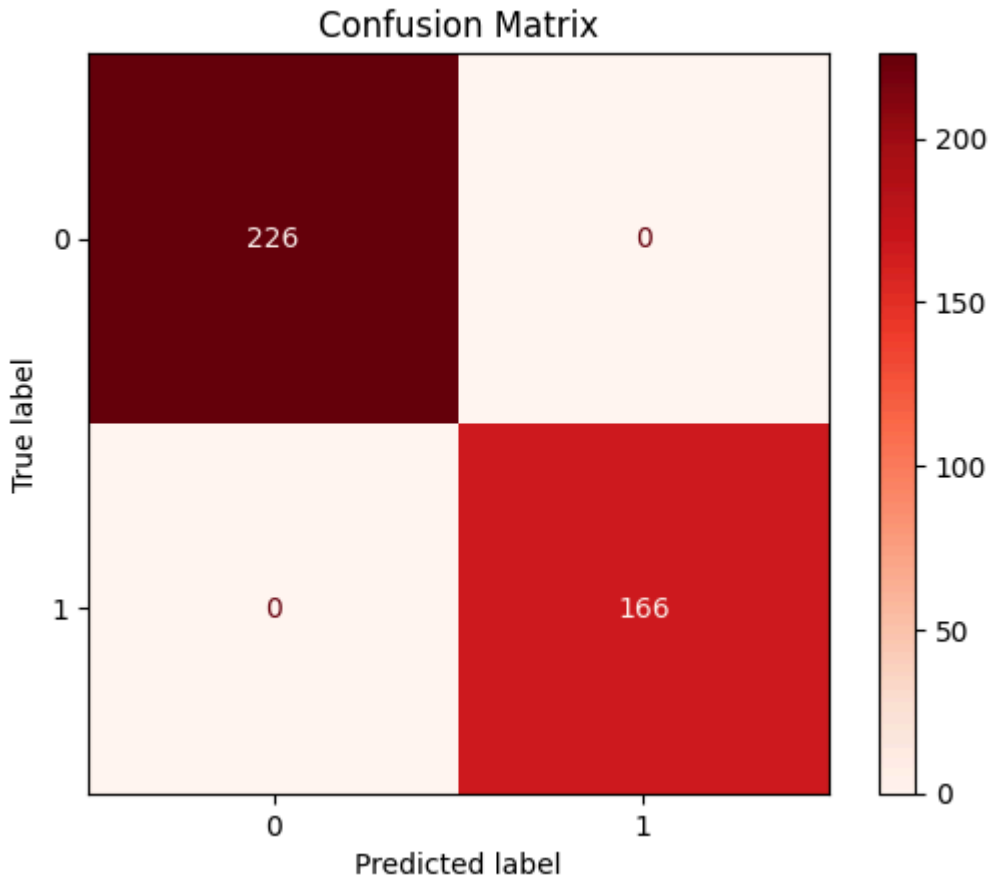
In [ ]:
```
y_pred = model.predict(x_test_scalled)
```

Classification Report

In [ ]:
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,C
```

In [ ]:
```
import matplotlib.pyplot as plt

cf = confusion_matrix(y_test,y_pred)
cm=ConfusionMatrixDisplay(cf)
cm.plot(cmap='Reds')
plt.title('Confusion Matrix')
plt.show()
```



In [ ]:
```
accuracy = accuracy_score(y_test,y_pred)
print("Accuracy:",accuracy)
```

```
Accuracy: 1.0
```

In [ ]:
```
cf = classification_report(y_test,y_pred)
print(cf)
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       226
           1       1.00      1.00      1.00       166

    accuracy                           1.00       392
   macro avg       1.00      1.00      1.00       392
weighted avg       1.00      1.00      1.00       392
```