```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn import metrics

import warnings
warnings.filterwarnings('ignore')
```

```
df=pd.read_csv("/content/INR=X (1).csv")
df.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-27	71.320000	71.635002	71.320000	71.324997	71.324997	0
1	2020-01-28	71.654999	71.654999	71.178001	71.440002	71.440002	0
2	2020-01-29	71.230103	71.425003	71.168503	71.230400	71.230400	0
3	2020-01-30	71.300003	71.711998	71.300003	71.300003	71.300003	0
4	2020-01-31	71.639999	71.639999	71.277496	71.639999	71.639999	0

```
df.shape
```

```
(262, 7)
```

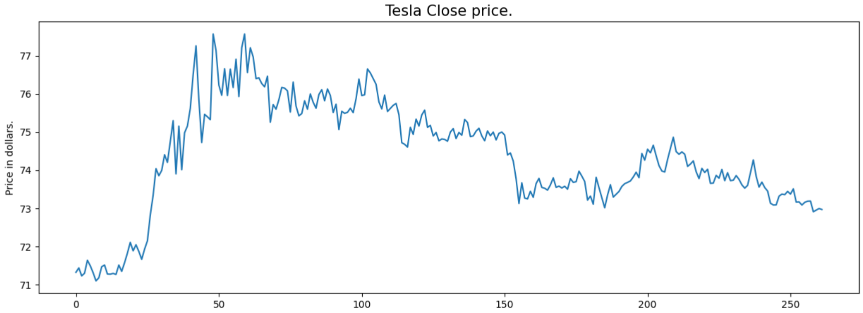
```
df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	262.000000	262.000000	262.000000	262.000000	262.000000	262.0
mean	74.373533	74.631087	74.011048	74.358489	74.358489	0.0
std	1.417620	1.494521	1.311330	1.426245	1.426245	0.0
min	71.100403	71.279999	71.064003	71.099998	71.099998	0.0
25%	73.546175	73.706577	73.202003	73.531049	73.531049	0.0
50%	74.332001	74.531300	73.881748	74.275799	74.275799	0.0
75%	75.484551	75.737499	75.067053	75.489424	75.489424	0.0
max	77.684998	77.754997	76.496300	77.570000	77.570000	0.0

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 262 entries, 0 to 261
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       262 non-null    object 
 1   Open       262 non-null    float64
 2   High       262 non-null    float64
 3   Low        262 non-null    float64
 4   Close      262 non-null    float64
 5   Adj Close  262 non-null    float64
 6   Volume     262 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 14.5+ KB
```

```
plt.figure(figsize=(15,5))
plt.plot(df['Close'])
plt.title('Tesla Close price.', fontsize=15)
plt.ylabel('Price in dollars.')
plt.show()
```



```
df.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-27	71.320000	71.635002	71.320000	71.324997	71.324997	0
1	2020-01-28	71.654999	71.654999	71.178001	71.440002	71.440002	0
2	2020-01-29	71.230103	71.425003	71.168503	71.230400	71.230400	0
3	2020-01-30	71.300003	71.711998	71.300003	71.300003	71.300003	0
4	2020-01-31	71.639999	71.639999	71.277496	71.639999	71.639999	0

```
df[df['Close'] == df['Adj Close']].shape
```

```
(262, 7)
```

```
df[df['Close'] == df['Adj Close']].shape
```

```
(262, 7)
```

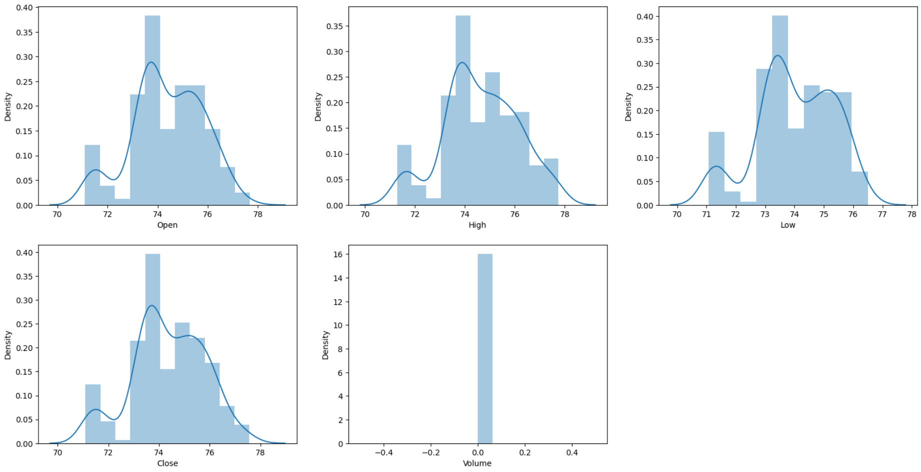
```
df.isnull().sum()
```

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

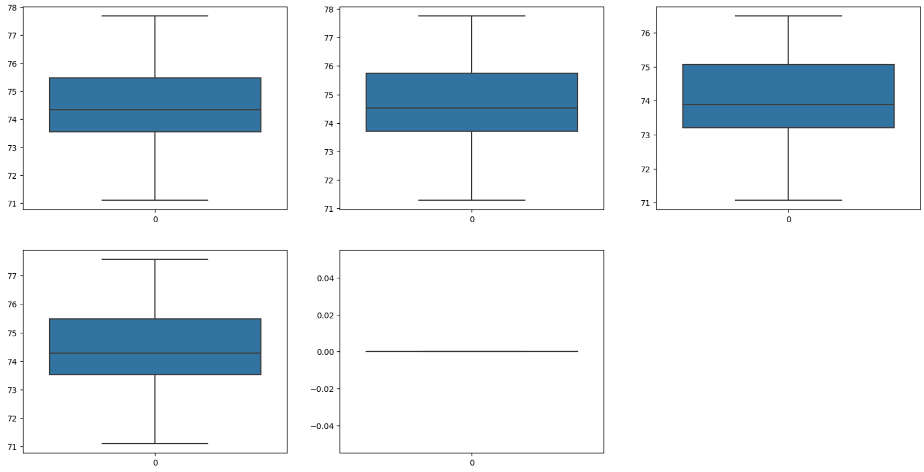
```
features = ['Open', 'High', 'Low', 'Close', 'Volume']

plt.subplots(figsize=(20,10))

for i, col in enumerate(features):
  plt.subplot(2,3,i+1)
  sb.distplot(df[col])
plt.show()
```

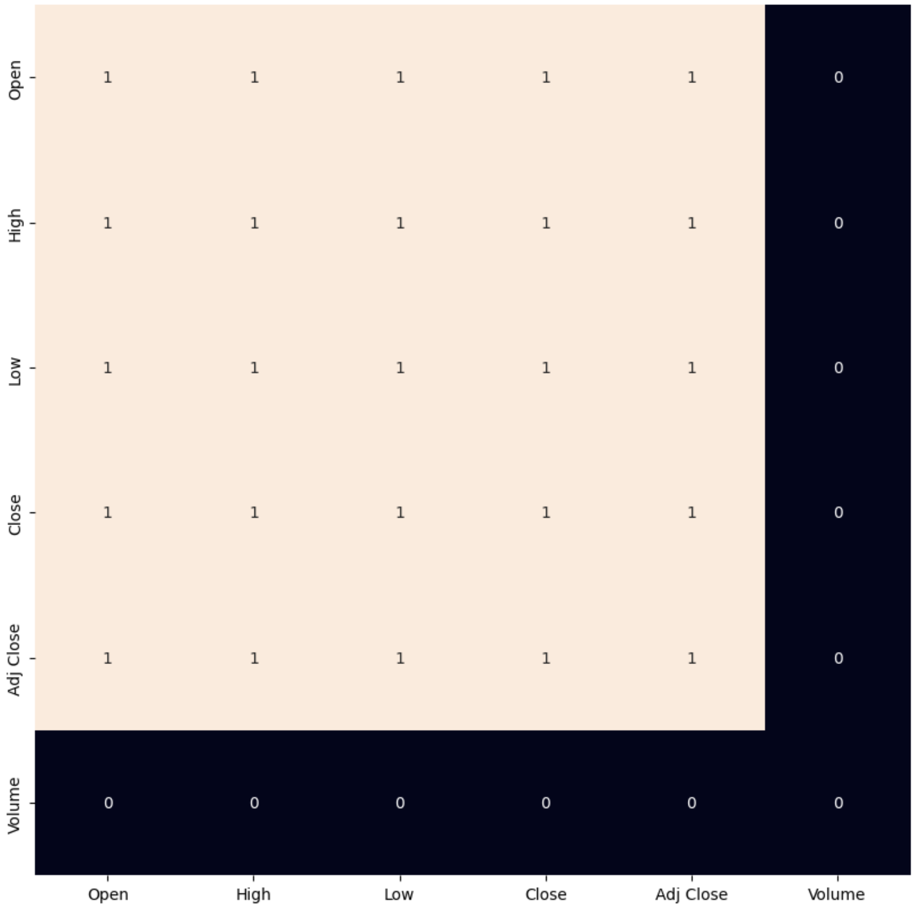


```
plt.subplots(figsize=(20,10))
for i, col in enumerate(features):
 plt.subplot(2,3,i+1)
 sb.boxplot(df[col])
plt.show()
```



```
plt.figure(figsize=(10, 10))

# As our concern is with the highly
# correlated features only so, we will visualize
# our heatmap as per that criteria only.
sb.heatmap(df.corr() > 0.9, annot=True, cbar=False)
plt.show()
```


Open
High
Low
Close
Adj Close
Volume
1 1 1 1 1 0
1 1 1 1 1 0
1 1 1 1 1 0
1 1 1 1 1 0
1 1 1 1 1 0
0 0 0 0 0 0
Open
High
Low
Close
Adj Close
Volume