m4wwh4tpd

March 5, 2024

[1]:
```
from pandas_datareader import data as pdr
```

[2]:
```
import pandas as pd

df= pd.read_csv("Group4_train_test.csv")
print(df.columns)
```

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'],
dtype='object')
```

[3]:
```
df.shape
```

[3]:
```
(639, 7)
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 639 entries, 0 to 638
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       639 non-null    object
 1   Open       639 non-null    float64
 2   High       639 non-null    float64
 3   Low        639 non-null    float64
 4   Close      639 non-null    float64
 5   Adj Close  639 non-null    float64
 6   Volume     639 non-null    int64
dtypes: float64(5), int64(1), object(1)
memory usage: 35.1+ KB
```

[5]:
```
df=df.drop("Volume",axis=1)
```

[6]:
```
df.head()
```

[6]:

	Date	Open	High	Low	Close	Adj Close
0	2021-01-01	535.549988	537.000000	526.099976	527.500000	514.302124
1	2021-01-04	532.299988	535.000000	524.299988	531.700012	518.397034

```
2  2021-01-05  526.650024  539.950012  523.000000  537.250000  523.808167
3  2021-01-06  538.750000  550.599976  535.849976  546.700012  533.021790
4  2021-01-07  552.150024  554.400024  539.750000  541.099976  527.561890
```

[ ]:

[7]:
```
from pandas_datareader import data as pdr
import yfinance as yf
yf.pdr_override()
import yfinance as yf
```

```
C:\Users\Administrator\anaconda3\lib\site-packages\yfinance\base.py:48:
FutureWarning: The default dtype for empty Series will be 'object' instead of
'float64' in a future version. Specify a dtype explicitly to silence this
warning.
  _empty_series = pd.Series()
```

[8]:
```
pip install yfinance
```

```
Requirement already satisfied: yfinance in
c:\users\administrator\anaconda3\lib\site-packages (0.2.36)
Requirement already satisfied: appdirs>=1.4.4 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (1.4.4)
Requirement already satisfied: pandas>=1.3.0 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (1.4.4)
Requirement already satisfied: peewee>=3.16.2 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (3.17.0)
Requirement already satisfied: lxml>=4.9.1 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (4.9.1)
Requirement already satisfied: beautifulsoup4>=4.11.1 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (4.11.1)
Requirement already satisfied: html5lib>=1.1 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (1.1)
Requirement already satisfied: numpy>=1.16.5 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (1.21.5)
Requirement already satisfied: multitasking>=0.0.7 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (0.0.11)
Requirement already satisfied: pytz>=2022.5 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (2023.4)
Requirement already satisfied: frozendict>=2.3.4 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (2.4.0)
Requirement already satisfied: requests>=2.31 in
c:\users\administrator\anaconda3\lib\site-packages (from yfinance) (2.31.0)
Requirement already satisfied: soupsieve>1.2 in
c:\users\administrator\anaconda3\lib\site-packages (from
beautifulsoup4>=4.11.1->yfinance) (2.3.1)
Requirement already satisfied: six>=1.9 in
```

```
c:\users\administrator\anaconda3\lib\site-packages (from
html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in
c:\users\administrator\anaconda3\lib\site-packages (from
html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance) (2022.9.14)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance) (2.0.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance) (1.26.11)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance) (3.3)
Note: you may need to restart the kernel to use updated packages.
```

[9]:
```
pip install fix_yahoo_finance
```

```
Requirement already satisfied: fix_yahoo_finance in
c:\users\administrator\anaconda3\lib\site-packages (0.1.37)
Requirement already satisfied: yfinance in
c:\users\administrator\anaconda3\lib\site-packages (from fix_yahoo_finance)
(0.2.36)
Requirement already satisfied: frozendict>=2.3.4 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (2.4.0)
Requirement already satisfied: numpy>=1.16.5 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (1.21.5)
Requirement already satisfied: lxml>=4.9.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (4.9.1)
Requirement already satisfied: pandas>=1.3.0 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (1.4.4)
Requirement already satisfied: peewee>=3.16.2 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (3.17.0)
Requirement already satisfied: beautifulsoup4>=4.11.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (4.11.1)
Requirement already satisfied: pytz>=2022.5 in
```

```
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (2023.4)
Requirement already satisfied: requests>=2.31 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (2.31.0)
Requirement already satisfied: appdirs>=1.4.4 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (1.4.4)
Requirement already satisfied: html5lib>=1.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (1.1)
Requirement already satisfied: multitasking>=0.0.7 in
c:\users\administrator\anaconda3\lib\site-packages (from
yfinance->fix_yahoo_finance) (0.0.11)
Requirement already satisfied: soupsieve>1.2 in
c:\users\administrator\anaconda3\lib\site-packages (from
beautifulsoup4>=4.11.1->yfinance->fix_yahoo_finance) (2.3.1)
Requirement already satisfied: six>=1.9 in
c:\users\administrator\anaconda3\lib\site-packages (from
html5lib>=1.1->yfinance->fix_yahoo_finance) (1.16.0)
Requirement already satisfied: webencodings in
c:\users\administrator\anaconda3\lib\site-packages (from
html5lib>=1.1->yfinance->fix_yahoo_finance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
pandas>=1.3.0->yfinance->fix_yahoo_finance) (2.8.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance->fix_yahoo_finance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance->fix_yahoo_finance) (3.3)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance->fix_yahoo_finance) (1.26.11)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\administrator\anaconda3\lib\site-packages (from
requests>=2.31->yfinance->fix_yahoo_finance) (2022.9.14)
Note: you may need to restart the kernel to use updated packages.
```

[10]:
```
import yfinance as yf

nf= yf.download('^NSEI', '2021-01-01', '2023-07-31')
nf=nf.reset_index(drop=True)
print(nf.columns)
nf.info()
```

```
[**********************100%%**********************]  1 of 1 completed
```

```
Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 638 entries, 0 to 637
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       638 non-null    float64
 1   High       638 non-null    float64
 2   Low        638 non-null    float64
 3   Close      638 non-null    float64
 4   Adj Close  638 non-null    float64
 5   Volume     638 non-null    int64
dtypes: float64(5), int64(1)
memory usage: 30.0 KB
```

[11]:
```
nf=nf.rename(columns={"Open": "nf_open", "High": "nf_high","Low":
  ↪"nf_low","Close":"nf_close","Adj Close":"nf_adj_close"})
```

[12]:
```
nf.head()
```

[12]:

	nf_open	nf_high	nf_low	nf_close	nf_adj_close	\
0	13996.099609	14049.849609	13991.349609	14018.500000	14018.500000	
1	14104.349609	14147.950195	13953.750000	14132.900391	14132.900391	
2	14075.150391	14215.599609	14048.150391	14199.500000	14199.500000	
3	14240.950195	14244.150391	14039.900391	14146.250000	14146.250000	
4	14253.750000	14256.250000	14123.099609	14137.349609	14137.349609	

	Volume
0	358100
1	495000
2	492500
3	632300
4	559200

[13]:
```
nb= yf.download('^NSEBANK', '2021-01-01', '2023-07-31')
print(nb.columns)
nb=nb.reset_index(drop=True)
nb.info()
```

```
[**********************100%%**********************]  1 of 1 completed

Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 636 entries, 0 to 635
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
```

```
 0   Open       636 non-null    float64
 1   High       636 non-null    float64
 2   Low        636 non-null    float64
 3   Close      636 non-null    float64
 4   Adj Close  636 non-null    float64
 5   Volume     636 non-null    int64
dtypes: float64(5), int64(1)
memory usage: 29.9 KB
```

[14]:
```
nb=nb.rename(columns={"Open": "nb_open", "High": "nb_high","Low":
 ↪"nb_low","Close":"nb_close","Adj Close":"nb_adj_close"})
```

[15]:
```
nb.head()
```

[15]:

	nb_open	nb_high	nb_low	nb_close	nb_adj_close	Volume
0	31485.150391	31489.599609	30893.650391	31212.449219	31212.085938	0
1	31041.099609	31767.650391	30935.550781	31722.250000	31721.880859	0
2	31839.949219	31982.300781	31548.150391	31797.900391	31797.531250	0
3	32129.800781	32177.400391	31911.500000	31956.000000	31955.628906	0
4	32298.050781	32298.050781	32002.949219	32084.199219	32083.826172	0

[16]:
```
be= yf.download('BSE.NS', '2021-01-01', '2023-07-31')
print(be.columns)
be=be.reset_index(drop=True)
be.info()
```

```
[**********************100%%**********************]  1 of 1 completed

Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 638 entries, 0 to 637
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       638 non-null    float64
 1   High       638 non-null    float64
 2   Low        638 non-null    float64
 3   Close      638 non-null    float64
 4   Adj Close  638 non-null    float64
 5   Volume     638 non-null    int64
```

```
dtypes: float64(5), int64(1)
memory usage: 30.0 KB
```

[17]:
```
be=be.rename(columns={"Open": "be_open", "High": "be_high","Low":
 ↪"be_low","Close":"be_close","Adj Close":"be_adj_close"})
```

[18]:
```
df1=pd.DataFrame(nf)
df2=pd.DataFrame(nb)
df3=pd.DataFrame(be)
df=pd.DataFrame(df)
df2
```

[18]:

	nb_open	nb_high	nb_low	nb_close	nb_adj_close	\
0	31485.150391	31489.599609	30893.650391	31212.449219	31212.085938	
1	31041.099609	31767.650391	30935.550781	31722.250000	31721.880859	
2	31839.949219	31982.300781	31548.150391	31797.900391	31797.531250	
3	32129.800781	32177.400391	31911.500000	31956.000000	31955.628906	
4	32298.050781	32298.050781	32002.949219	32084.199219	32083.826172	
..	…	…	…	…	…	
631	46131.898438	46191.500000	45858.750000	45923.050781	45923.050781	
632	46154.699219	46156.101562	45622.851562	45845.000000	45845.000000	
633	45935.148438	46096.601562	45804.699219	46062.351562	46062.351562	
634	46285.851562	46310.101562	45570.648438	45679.300781	45679.300781	
635	45560.898438	45727.750000	45238.800781	45468.101562	45468.101562	

	Volume
0	0
1	0
2	0
3	0
4	0
..	…
631	225800
632	228100
633	298400
634	320400
635	194700

[636 rows x 6 columns]

[19]:
```
a=[df,df1,df2,df3]
merged_df=pd.concat(a,axis=1)
```

[20]:
```
merged_df.info
```

[20]:
```
<bound method DataFrame.info of            Date         Open         High
Low       Close   Adj Close  \
0    2021-01-01   535.549988   537.000000  526.099976  527.500000  514.302124
1    2021-01-04   532.299988   535.000000  524.299988  531.700012  518.397034
2    2021-01-05   526.650024   539.950012  523.000000  537.250000  523.808167
3    2021-01-06   538.750000   550.599976  535.849976  546.700012  533.021790
4    2021-01-07   552.150024   554.400024  539.750000  541.099976  527.561890
..          …          …          …          …          …          …
634  2023-07-25   999.000000   999.000000  988.250000  994.700012  978.489502
635  2023-07-26   993.099976   997.750000  990.000000  996.450012  980.210999
636  2023-07-27  1002.549988  1006.500000  987.200012  990.250000  974.112061
637  2023-07-28   988.000000   998.700012  977.049988  996.200012  979.965088
638  2023-07-31   996.000000   999.400024  988.599976  998.299988  982.030884

          nf_open       nf_high        nf_low      nf_close  …  \
0    13996.099609  14049.849609  13991.349609  14018.500000  …
1    14104.349609  14147.950195  13953.750000  14132.900391  …
2    14075.150391  14215.599609  14048.150391  14199.500000  …
3    14240.950195  14244.150391  14039.900391  14146.250000  …
4    14253.750000  14256.250000  14123.099609  14137.349609  …
..            …          …          …          …  …
634  19729.349609  19729.349609  19615.949219  19680.599609  …
635  19733.349609  19825.599609  19716.699219  19778.300781  …
636  19850.900391  19867.550781  19603.550781  19659.900391  …
637  19659.750000  19695.900391  19563.099609  19646.050781  …
638           NaN           NaN           NaN           NaN  …

           nb_low      nb_close  nb_adj_close    Volume     be_open  \
0    30893.650391  31212.449219  31212.085938       0.0  206.966660
1    30935.550781  31722.250000  31721.880859       0.0  216.666672
2    31548.150391  31797.900391  31797.531250       0.0  215.000000
3    31911.500000  31956.000000  31955.628906       0.0  213.666672
4    32002.949219  32084.199219  32083.826172       0.0  215.000000
..            …          …          …       …          …
634  45570.648438  45679.300781  45679.300781  320400.0  732.900024
635  45238.800781  45468.101562  45468.101562  194700.0  739.500000
636           NaN           NaN           NaN       NaN  771.299988
637           NaN           NaN           NaN       NaN  782.000000
638           NaN           NaN           NaN       NaN         NaN

        be_high      be_low    be_close  be_adj_close     Volume
0    214.333328  206.966660  211.250000    200.081833  3749283.0
1    219.333328  213.983337  215.716660    204.312363  2935995.0
2    215.899994  211.716660  213.483337    202.197098  1299579.0
3    216.666672  210.399994  213.300003    202.023453  1499637.0
4    217.649994  213.000000  214.233337    202.907440  2451396.0
..          …          …          …           …        …
```

```
634  744.000000  728.000000  737.750000    727.186829   832761.0
635  774.000000  738.849976  768.849976    757.841492  3198014.0
636  797.900024  771.000000  779.650024    768.486938  2807890.0
637  810.000000  766.000000  802.049988    790.566162  3099291.0
638         NaN         NaN         NaN           NaN        NaN

[639 rows x 24 columns]>
```

[21]:
```
merged_df.head()
```

[21]:
```
         Date        Open        High         Low       Close   Adj Close  \
0  2021-01-01  535.549988  537.000000  526.099976  527.500000  514.302124
1  2021-01-04  532.299988  535.000000  524.299988  531.700012  518.397034
2  2021-01-05  526.650024  539.950012  523.000000  537.250000  523.808167
3  2021-01-06  538.750000  550.599976  535.849976  546.700012  533.021790
4  2021-01-07  552.150024  554.400024  539.750000  541.099976  527.561890

        nf_open       nf_high        nf_low      nf_close  …        nb_low  \
0  13996.099609  14049.849609  13991.349609  14018.500000  …  30893.650391
1  14104.349609  14147.950195  13953.750000  14132.900391  …  30935.550781
2  14075.150391  14215.599609  14048.150391  14199.500000  …  31548.150391
3  14240.950195  14244.150391  14039.900391  14146.250000  …  31911.500000
4  14253.750000  14256.250000  14123.099609  14137.349609  …  32002.949219

       nb_close  nb_adj_close  Volume     be_open     be_high      be_low  \
0  31212.449219  31212.085938     0.0  206.966660  214.333328  206.966660
1  31722.250000  31721.880859     0.0  216.666672  219.333328  213.983337
2  31797.900391  31797.531250     0.0  215.000000  215.899994  211.716660
3  31956.000000  31955.628906     0.0  213.666672  216.666672  210.399994
4  32084.199219  32083.826172     0.0  215.000000  217.649994  213.000000

     be_close  be_adj_close     Volume
0  211.250000    200.081833  3749283.0
1  215.716660    204.312363  2935995.0
2  213.483337    202.197098  1299579.0
3  213.300003    202.023453  1499637.0
4  214.233337    202.907440  2451396.0

[5 rows x 24 columns]
```

[22]:
```
merged_df.columns
```

[22]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'nf_open',
       'nf_high', 'nf_low', 'nf_close', 'nf_adj_close', 'Volume', 'nb_open',
       'nb_high', 'nb_low', 'nb_close', 'nb_adj_close', 'Volume', 'be_open',
       'be_high', 'be_low', 'be_close', 'be_adj_close', 'Volume'],
      dtype='object')
```

[23]:
```
data=merged_df
```

[24]:
```
data["Open_Close"]=data['Open']-data['Close']
data["High_Low"]=data['High']-data['Low']
data["High/Low"]=data['High']/data['Low']
```

[25]:
```
data["Next_close"]=data["Close"].shift(-1)
```

[26]:
```
data["Next_Open"]=data["Open"].shift(-1)
```

[27]:
```
data.info
```

[27]:
```
<bound method DataFrame.info of            Date         Open         High
Low       Close   Adj Close  \
0    2021-01-01   535.549988   537.000000  526.099976  527.500000  514.302124
1    2021-01-04   532.299988   535.000000  524.299988  531.700012  518.397034
2    2021-01-05   526.650024   539.950012  523.000000  537.250000  523.808167
3    2021-01-06   538.750000   550.599976  535.849976  546.700012  533.021790
4    2021-01-07   552.150024   554.400024  539.750000  541.099976  527.561890
..          …          …          …          …          …          …
634  2023-07-25   999.000000   999.000000  988.250000  994.700012  978.489502
635  2023-07-26   993.099976   997.750000  990.000000  996.450012  980.210999
636  2023-07-27  1002.549988  1006.500000  987.200012  990.250000  974.112061
637  2023-07-28   988.000000   998.700012  977.049988  996.200012  979.965088
638  2023-07-31   996.000000   999.400024  988.599976  998.299988  982.030884

          nf_open       nf_high        nf_low      nf_close  …     be_high  \
0    13996.099609  14049.849609  13991.349609  14018.500000  …  214.333328
1    14104.349609  14147.950195  13953.750000  14132.900391  …  219.333328
2    14075.150391  14215.599609  14048.150391  14199.500000  …  215.899994
3    14240.950195  14244.150391  14039.900391  14146.250000  …  216.666672
4    14253.750000  14256.250000  14123.099609  14137.349609  …  217.649994
..            …          …          …          …  …          …
634  19729.349609  19729.349609  19615.949219  19680.599609  …  744.000000
635  19733.349609  19825.599609  19716.699219  19778.300781  …  774.000000
636  19850.900391  19867.550781  19603.550781  19659.900391  …  797.900024
637  19659.750000  19695.900391  19563.099609  19646.050781  …  810.000000
638           NaN           NaN           NaN           NaN  …         NaN

         be_low    be_close  be_adj_close     Volume  Open_Close   High_Low  \
0    206.966660  211.250000    200.081833  3749283.0    8.049988  10.900024
1    213.983337  215.716660    204.312363  2935995.0    0.599976  10.700012
2    211.716660  213.483337    202.197098  1299579.0  -10.599976  16.950012
3    210.399994  213.300003    202.023453  1499637.0   -7.950012  14.750000
4    213.000000  214.233337    202.907440  2451396.0   11.050049  14.650024
..          …          …          …          …          …          …
634  728.000000  737.750000    727.186829   832761.0    4.299988  10.750000
```

```
635  738.849976  768.849976    757.841492  3198014.0   -3.350037   7.750000
636  771.000000  779.650024    768.486938  2807890.0   12.299988  19.299988
637  766.000000  802.049988    790.566162  3099291.0   -8.200012  21.650024
638         NaN         NaN           NaN        NaN   -2.299988  10.800049

     High/Low  Next_close    Next_Open
0    1.020719  531.700012   532.299988
1    1.020408  537.250000   526.650024
2    1.032409  546.700012   538.750000
3    1.027526  541.099976   552.150024
4    1.027142  542.049988   547.000000
..        …         …           …
634  1.010878  996.450012   993.099976
635  1.007828  990.250000  1002.549988
636  1.019550  996.200012   988.000000
637  1.022159  998.299988   996.000000
638  1.010925         NaN          NaN

[639 rows x 29 columns]>
```

[28]:
```
data=data.dropna()
```

[29]:
```
data.shape
```

[29]:
```
(636, 29)
```

[30]:
```
data.corr()["Next_close"]
```

[30]:
```
Open            0.992101
High            0.994225
Low             0.994223
Close           0.995394
Adj Close       0.995442
nf_open         0.893137
nf_high         0.894740
nf_low          0.896759
nf_close        0.896709
nf_adj_close    0.896709
Volume         -0.566929
nb_open         0.929348
nb_high         0.932229
nb_low          0.934108
nb_close        0.935307
nb_adj_close    0.935305
Volume          0.040913
be_open         0.530321
be_high         0.519460
```

```
be_low          0.542564
be_close        0.531849
be_adj_close    0.557150
Volume         -0.186121
Open_Close     -0.012787
High_Low       -0.058769
High/Low       -0.385208
Next_close      1.000000
Next_Open       0.996706
Name: Next_close, dtype: float64
```

[31]:
```
x=data[["Open","High","Adj Close","Next_Open","nb_open","nb_adj_close"]]
```

[32]:
```
y=data["Next_close"]
```

[33]:
```
from sklearn.model_selection import train_test_split
```

[34]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,
                                               test_size=0.2,random_state=10)
x_train.head()
```

[34]:
```
            Open        High   Adj Close   Next_Open       nb_open  \
311  746.799988  755.450012  725.358154  732.299988  37711.300781
505  864.700012  874.900024  859.215515  877.250000  42241.199219
84   612.000000  614.799988  591.910583  611.000000  33175.300781
105  648.000000  648.000000  626.327454  644.400024  35114.250000
339  712.349976  717.000000  691.671692  693.099976  33461.000000

     nb_adj_close
311  37632.363281
505  42235.050781
84   33142.011719
105  34800.093750
339  33315.261719
```

[35]:
```
x_train.shape
```

[35]:
```
(508, 6)
```

[36]:
```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(x_train,y_train)
```

[36]:
```
LinearRegression()
```

[37]:
```
y_pred=model.predict(x_test)
```

[38]:
```
print(y_pred)
```

```
[630.19632335 937.58916244 756.7022818  715.31074909 915.91898551
 756.55935235 898.78861718 737.72335936 720.30027693 609.12666925
 788.04085483 727.1395777  916.82642853 870.7186899  901.44561537
 579.88724845 694.65567996 860.09448637 724.08539934 788.2284666
 673.8157951  878.74058552 588.4789094  611.26878979 804.12419509
 842.36921659 945.70582979 855.81760601 942.42757101 575.95208184
 708.47598988 754.17732615 619.47384857 852.79058881 750.73188577
 892.77713788 713.15135924 738.35304841 717.47934197 905.73315949
 735.95322713 703.50981505 982.55807235 628.64284496 826.0363773
 906.06682805 842.54185708 553.29879103 851.1139264  838.7749241
 877.41942517 886.29276328 913.12687622 875.59779227 710.78878943
 942.27519436 642.82897229 753.5501709  550.82422016 647.76784005
 804.94108922 690.59963913 733.57335115 655.165794   809.58925785
 635.24187074 750.69597886 700.68860799 732.92751058 799.17418121
 946.75988662 883.6280002  870.97645093 801.12071131 771.41874306
 922.91112342 921.25272909 857.15330969 622.74938759 756.51283541
 572.55494531 706.20676651 935.68659254 614.77188349 727.17355887
 854.13598002 681.79949972 522.02230933 552.38673396 751.96075574
 713.24135516 914.42167727 807.38081181 708.11651343 761.54286945
 572.23223078 879.06603699 655.12578204 755.19850487 803.03453994
 775.93101144 913.5961955  569.30699318 742.72362672 877.27347416
 715.16996699 668.97524217 743.91701264 680.25228253 864.01691885
 613.39890355 867.59689752 711.95797198 614.95306487 872.60660199
 818.24982185 879.19962557 758.18630132 706.91240515 606.12347229
 764.65342922 921.40940666 642.34398276 684.03405846 932.75768074
 856.34906784 910.31566835 824.58483895]
```

[39]:
```
dataframe=pd.DataFrame({"ACTUAL":y_test,"PREDICTED":y_pred})
```

[40]:
```
dataframe
```

[40]:

	ACTUAL	PREDICTED
27	630.650024	630.196323
475	933.599976	937.589162
309	746.599976	756.702282
238	720.349976	715.310749
571	914.950012	915.918986
..	…	…
142	681.400024	684.034058
604	943.549988	932.757681
429	852.549988	856.349068
486	909.700012	910.315668
392	817.750000	824.584839

[128 rows x 2 columns]

[41]:
```
sample_dataframe=dataframe.head(10)
```

[42]:
```
model.score(x_test,y_test)
```

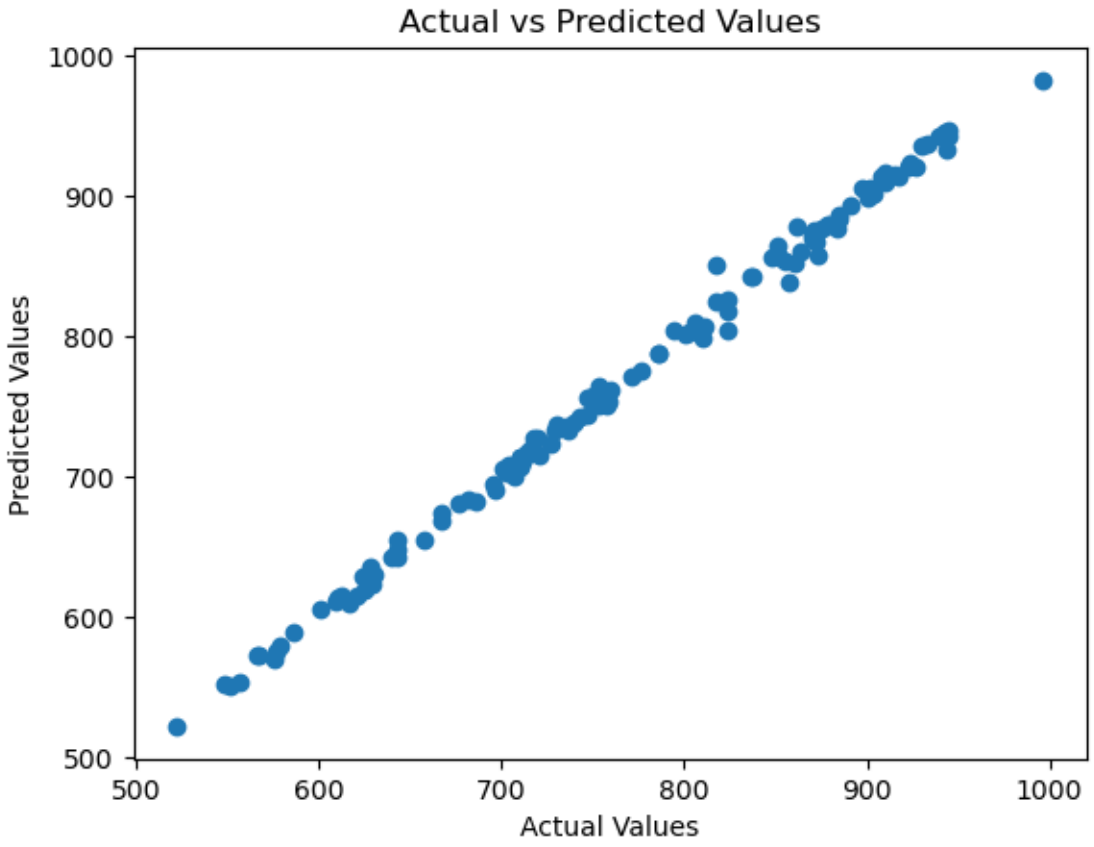
[42]: 0.99668863244443

[43]:
```
from sklearn.metrics import mean_squared_error
mean_squared_error(y_test,y_pred)
```

[43]: 42.39696454624645

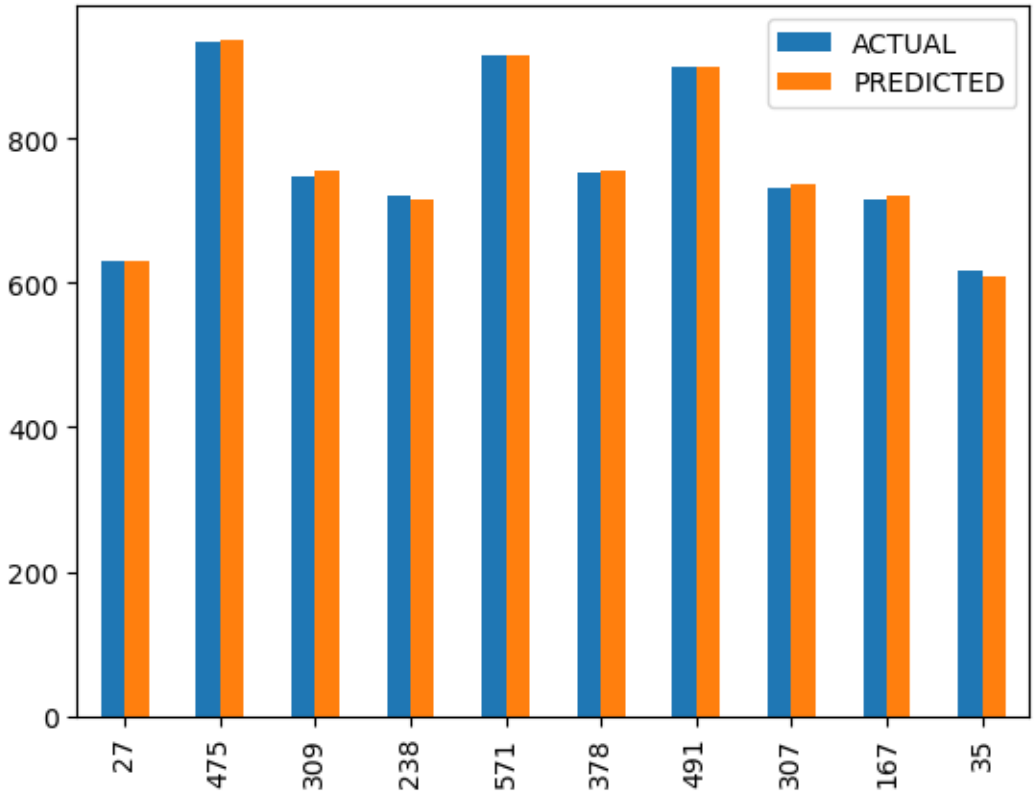
[44]:
```
import matplotlib.pyplot as plt

plt.scatter(y_test, y_pred)
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs Predicted Values')
plt.show()
```



[45]:
```
sample_dataframe.plot(kind='bar')
```

[45]: <AxesSubplot:>



[46]:
```
from sklearn.linear_model import ElasticNet


model2 = ElasticNet()
model2.fit(x_train, y_train)
```

```
C:\Users\Administrator\anaconda3\lib\site-
packages\sklearn\linear_model\_coordinate_descent.py:647: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 7.678e+03, tolerance: 7.185e+02
  model = cd_fast.enet_coordinate_descent(
```

[46]: ElasticNet()

[47]:
```
model2.score(x_test,y_test)
```

[47]: 0.996648529935843

[48]:
```
from sklearn.linear_model import Ridge

model3 = Ridge()
model3.fit(x_train, y_train)
```

[48]:
```
Ridge()
```

[49]:
```
model3.score(x_test,y_test)
```

[49]:
```
0.9966886699666904
```

[50]:
```
from sklearn.linear_model import Lasso

model4 = Lasso()
model4.fit(x_train, y_train)

model4.score(x_test,y_test)
```

```
C:\Users\Administrator\anaconda3\lib\site-
packages\sklearn\linear_model\_coordinate_descent.py:647: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 5.720e+03, tolerance: 7.185e+02
  model = cd_fast.enet_coordinate_descent(
```

[50]:
```
0.9966643672229265
```

[51]:
```
val= pd.read_csv("Group4_validation.csv")
print(val.columns)
```

```
Index(['Date', 'Open', 'High', 'Low', 'Adj Close', 'Volume'], dtype='object')
```

[52]:
```
val.head(10)
```

[52]:

	Date	Open	High	Low	Adj Close	Volume
0	2023-08-02	985.250000	994.099976	977.500000	970.865845	15787996
1	2023-08-03	976.549988	983.000000	960.049988	949.273560	30994707
2	2023-08-04	968.650024	976.000000	961.250000	954.683899	20582882
3	2023-08-07	970.950012	981.500000	968.200012	959.750000	16686062
4	2023-08-08	975.750000	986.599976	972.299988	971.900024	23784778
5	2023-08-09	974.950012	975.299988	963.599976	972.700012	19317331
6	2023-08-10	971.549988	973.650024	959.450012	964.099976	24858441
7	2023-08-11	963.400024	963.500000	950.000000	952.849976	17162177
8	2023-08-14	949.000000	962.900024	946.000000	959.549988	19026691
9	2023-08-16	954.450012	959.799988	946.599976	956.500000	17930805

[53]:
```
val.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       20 non-null     object
 1   Open       20 non-null     float64
 2   High       20 non-null     float64
 3   Low        20 non-null     float64
 4   Adj Close  20 non-null     float64
 5   Volume     20 non-null     int64
dtypes: float64(4), int64(1), object(1)
memory usage: 1.1+ KB
```

[54]:
```
val=val.dropna()
```

[55]:
```
val.head(10)
```

[55]:

	Date	Open	High	Low	Adj Close	Volume
0	2023-08-02	985.250000	994.099976	977.500000	970.865845	15787996
1	2023-08-03	976.549988	983.000000	960.049988	949.273560	30994707
2	2023-08-04	968.650024	976.000000	961.250000	954.683899	20582882
3	2023-08-07	970.950012	981.500000	968.200012	959.750000	16686062
4	2023-08-08	975.750000	986.599976	972.299988	971.900024	23784778
5	2023-08-09	974.950012	975.299988	963.599976	972.700012	19317331
6	2023-08-10	971.549988	973.650024	959.450012	964.099976	24858441
7	2023-08-11	963.400024	963.500000	950.000000	952.849976	17162177
8	2023-08-14	949.000000	962.900024	946.000000	959.549988	19026691
9	2023-08-16	954.450012	959.799988	946.599976	956.500000	17930805

[56]:
```
val['Date'] = pd.to_datetime(val['Date'])
val.set_index('Date', inplace=True)
```

[57]:
```
val
```

[57]:

Date	Open	High	Low	Adj Close	Volume
2023-08-02	985.250000	994.099976	977.500000	970.865845	15787996
2023-08-03	976.549988	983.000000	960.049988	949.273560	30994707
2023-08-04	968.650024	976.000000	961.250000	954.683899	20582882
2023-08-07	970.950012	981.500000	968.200012	959.750000	16686062
2023-08-08	975.750000	986.599976	972.299988	971.900024	23784778
2023-08-09	974.950012	975.299988	963.599976	972.700012	19317331
2023-08-10	971.549988	973.650024	959.450012	964.099976	24858441
2023-08-11	963.400024	963.500000	950.000000	952.849976	17162177
2023-08-14	949.000000	962.900024	946.000000	959.549988	19026691
2023-08-16	954.450012	959.799988	946.599976	956.500000	17930805

```
2023-08-17  956.950012  957.799988  946.450012  951.650024  27077611
2023-08-18  947.900024  955.000000  946.000000  950.650024  13257508
2023-08-21  950.700012  959.299988  948.400024  955.200012  13358017
2023-08-22  956.400024  956.950012  949.299988  952.000000  16872719
2023-08-23  954.000000  968.049988  949.849976  966.849976  14707331
2023-08-24  972.500000  982.400024  966.750000  968.950012  28389740
2023-08-25  962.950012  972.599976  960.200012  970.400024  10275758
2023-08-28  969.950012  975.000000  965.049988  970.549988  10481574
2023-08-29  973.299988  977.250000  965.650024  967.750000  13209913
2023-08-30  970.450012  975.900024  955.400024  958.900024  13936165
```

[58]:
```
val["Next_Open"]=val["Open"].shift(-1)
```

[59]:
```
nb_v= yf.download('^NSEBANK', '2023-08-02', '2023-08-31')
print(nb_v.columns)
nb_v.info()
```

```
[**********************100%%**********************]  1 of 1 completed

Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2023-08-02 to 2023-08-30
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       20 non-null     float64
 1   High       20 non-null     float64
 2   Low        20 non-null     float64
 3   Close      20 non-null     float64
 4   Adj Close  20 non-null     float64
 5   Volume     20 non-null     int64
dtypes: float64(5), int64(1)
memory usage: 1.1 KB
```

[60]:
```
val
```

[60]:
```
                 Open        High         Low   Adj Close    Volume  \
Date
2023-08-02  985.250000  994.099976  977.500000  970.865845  15787996
2023-08-03  976.549988  983.000000  960.049988  949.273560  30994707
2023-08-04  968.650024  976.000000  961.250000  954.683899  20582882
2023-08-07  970.950012  981.500000  968.200012  959.750000  16686062
2023-08-08  975.750000  986.599976  972.299988  971.900024  23784778
2023-08-09  974.950012  975.299988  963.599976  972.700012  19317331
2023-08-10  971.549988  973.650024  959.450012  964.099976  24858441
2023-08-11  963.400024  963.500000  950.000000  952.849976  17162177
```

```
2023-08-14  949.000000  962.900024  946.000000  959.549988  19026691
2023-08-16  954.450012  959.799988  946.599976  956.500000  17930805
2023-08-17  956.950012  957.799988  946.450012  951.650024  27077611
2023-08-18  947.900024  955.000000  946.000000  950.650024  13257508
2023-08-21  950.700012  959.299988  948.400024  955.200012  13358017
2023-08-22  956.400024  956.950012  949.299988  952.000000  16872719
2023-08-23  954.000000  968.049988  949.849976  966.849976  14707331
2023-08-24  972.500000  982.400024  966.750000  968.950012  28389740
2023-08-25  962.950012  972.599976  960.200012  970.400024  10275758
2023-08-28  969.950012  975.000000  965.049988  970.549988  10481574
2023-08-29  973.299988  977.250000  965.650024  967.750000  13209913
2023-08-30  970.450012  975.900024  955.400024  958.900024  13936165

             Next_Open
Date
2023-08-02  976.549988
2023-08-03  968.650024
2023-08-04  970.950012
2023-08-07  975.750000
2023-08-08  974.950012
2023-08-09  971.549988
2023-08-10  963.400024
2023-08-11  949.000000
2023-08-14  954.450012
2023-08-16  956.950012
2023-08-17  947.900024
2023-08-18  950.700012
2023-08-21  956.400024
2023-08-22  954.000000
2023-08-23  972.500000
2023-08-24  962.950012
2023-08-25  969.950012
2023-08-28  973.299988
2023-08-29  970.450012
2023-08-30         NaN
```

[61]:
```
c=nb_v.drop(["High","Low","Close","Volume"],axis = 1)
c=c.rename(columns={"Open":"nb_open","Adj Close":"nb_adj_close"})
```

[62]:
```
c
```

[62]:
```
                nb_open  nb_adj_close
Date
2023-08-02  45234.648438  44995.699219
2023-08-03  44862.851562  44513.449219
2023-08-04  44754.750000  44879.500000
2023-08-07  44993.699219  44837.500000
```

```
2023-08-08  44888.949219  44964.449219
2023-08-09  44973.449219  44880.699219
2023-08-10  44797.648438  44541.800781
2023-08-11  44568.148438  44199.101562
2023-08-14  44066.601562  44090.949219
2023-08-16  43726.250000  43946.398438
2023-08-17  43897.250000  43891.351562
2023-08-18  43724.750000  43851.050781
2023-08-21  43952.851562  44002.000000
2023-08-22  44125.000000  43993.250000
2023-08-23  44064.500000  44479.050781
2023-08-24  44704.199219  44496.199219
2023-08-25  44276.199219  44231.449219
2023-08-28  44253.648438  44494.648438
2023-08-29  44655.750000  44495.250000
2023-08-30  44706.550781  44232.601562
```

[63]:
```
val.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2023-08-02 to 2023-08-30
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       20 non-null     float64
 1   High       20 non-null     float64
 2   Low        20 non-null     float64
 3   Adj Close  20 non-null     float64
 4   Volume     20 non-null     int64
 5   Next_Open  19 non-null     float64
dtypes: float64(5), int64(1)
memory usage: 1.1 KB
```

[64]:
```
c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2023-08-02 to 2023-08-30
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   nb_open       20 non-null     float64
 1   nb_adj_close  20 non-null     float64
dtypes: float64(2)
memory usage: 480.0 bytes
```

[65]:
```
val=pd.DataFrame(val)
```

[66]:
```
v = pd.concat([val,c],axis=1)
```

[67]:
```
v
```

[67]:

Date	Open	High	Low	Adj Close	Volume	Next_Open	nb_open	nb_adj_close
2023-08-02	985.250000	994.099976	977.500000	970.865845	15787996	976.549988	45234.648438	44995.699219
2023-08-03	976.549988	983.000000	960.049988	949.273560	30994707	968.650024	44862.851562	44513.449219
2023-08-04	968.650024	976.000000	961.250000	954.683899	20582882	970.950012	44754.750000	44879.500000
2023-08-07	970.950012	981.500000	968.200012	959.750000	16686062	975.750000	44993.699219	44837.500000
2023-08-08	975.750000	986.599976	972.299988	971.900024	23784778	974.950012	44888.949219	44964.449219
2023-08-09	974.950012	975.299988	963.599976	972.700012	19317331	971.549988	44973.449219	44880.699219
2023-08-10	971.549988	973.650024	959.450012	964.099976	24858441	963.400024	44797.648438	44541.800781
2023-08-11	963.400024	963.500000	950.000000	952.849976	17162177	949.000000	44568.148438	44199.101562
2023-08-14	949.000000	962.900024	946.000000	959.549988	19026691	954.450012	44066.601562	44090.949219
2023-08-16	954.450012	959.799988	946.599976	956.500000	17930805	956.950012	43726.250000	43946.398438
2023-08-17	956.950012	957.799988	946.450012	951.650024	27077611	947.900024	43897.250000	43891.351562
2023-08-18	947.900024	955.000000	946.000000	950.650024	13257508	950.700012	43724.750000	43851.050781
2023-08-21	950.700012	959.299988	948.400024	955.200012	13358017	956.400024	43952.851562	44002.000000
2023-08-22	956.400024	956.950012	949.299988	952.000000	16872719	954.000000	44125.000000	43993.250000
2023-08-23	954.000000	968.049988	949.849976	966.849976	14707331	972.500000	44064.500000	44479.050781
2023-08-24	972.500000	982.400024	966.750000	968.950012	28389740	962.950012	44704.199219	44496.199219
2023-08-25	962.950012	972.599976	960.200012	970.400024	10275758	969.950012	44276.199219	44231.449219
2023-08-28	969.950012	975.000000	965.049988	970.549988	10481574	973.299988	44253.648438	44494.648438
2023-08-29	973.299988	977.250000	965.650024	967.750000	13209913			
2023-08-30	970.450012	975.900024	955.400024	958.900024	13936165			

```
2023-08-29  970.450012  44655.750000  44495.250000
2023-08-30         NaN  44706.550781  44232.601562
```

[68]:
```
v=v.drop(["Volume","Low"],axis=1)
```

[69]:
```
v.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2023-08-02 to 2023-08-30
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Open          20 non-null     float64
 1   High          20 non-null     float64
 2   Adj Close     20 non-null     float64
 3   Next_Open     19 non-null     float64
 4   nb_open       20 non-null     float64
 5   nb_adj_close  20 non-null     float64
dtypes: float64(6)
memory usage: 1.1 KB
```

[70]:
```
v.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2023-08-02 to 2023-08-30
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Open          20 non-null     float64
 1   High          20 non-null     float64
 2   Adj Close     20 non-null     float64
 3   Next_Open     19 non-null     float64
 4   nb_open       20 non-null     float64
 5   nb_adj_close  20 non-null     float64
dtypes: float64(6)
memory usage: 1.1 KB
```

[71]:
```
v=v.dropna()
```

[72]:
```
pred=model.predict(v)
```

[73]:
```
print(pred)
```

```
[972.84386019 962.56463351 973.08105892 972.60172806 976.59147186
 970.56584107 959.66350564 943.70627333 955.05912456 961.11256741
 948.79590862 953.20289818 957.30589295 952.14489082 979.13842559
 960.14730578 969.02427641 977.69558017 967.96470913]
```

[74]:
```
v["predicted closing"]=pred
```

[75]:
```
v
```

[75]:

Date	Open	High	Adj Close	Next_Open	nb_open	nb_adj_close	predicted closing
2023-08-02	985.250000	994.099976	970.865845	976.549988	45234.648438	44995.699219	972.843860
2023-08-03	976.549988	983.000000	949.273560	968.650024	44862.851562	44513.449219	962.564634
2023-08-04	968.650024	976.000000	954.683899	970.950012	44754.750000	44879.500000	973.081059
2023-08-07	970.950012	981.500000	959.750000	975.750000	44993.699219	44837.500000	972.601728
2023-08-08	975.750000	986.599976	971.900024	974.950012	44888.949219	44964.449219	976.591472
2023-08-09	974.950012	975.299988	972.700012	971.549988	44973.449219	44880.699219	970.565841
2023-08-10	971.549988	973.650024	964.099976	963.400024	44797.648438	44541.800781	959.663506
2023-08-11	963.400024	963.500000	952.849976	949.000000	44568.148438	44199.101562	943.706273
2023-08-14	949.000000	962.900024	959.549988	954.450012	44066.601562	44090.949219	955.059125
2023-08-16	954.450012	959.799988	956.500000	956.950012	43726.250000	43946.398438	961.112567
2023-08-17	956.950012	957.799988	951.650024	947.900024	43897.250000	43891.351562	948.795909
2023-08-18	947.900024	955.000000	950.650024	950.700012	43724.750000	43851.050781	953.202898
2023-08-21	950.700012	959.299988	955.200012	956.400024	43952.851562	44002.000000	957.305893
2023-08-22	956.400024	956.950012	952.000000	954.000000	44125.000000	43993.250000	952.144891
2023-08-23	954.000000	968.049988	966.849976	972.500000	44064.500000	44479.050781	979.138426
2023-08-24	972.500000	982.400024	968.950012	962.950012	44704.199219	44496.199219	960.147306
2023-08-25	962.950012	972.599976	970.400024	969.950012	44276.199219	44231.449219	969.024276
2023-08-28	969.950012	975.000000	970.549988	973.299988	44253.648438	44494.648438	977.695580
2023-08-29	973.299988	977.250000	967.750000	970.450012	44655.750000	44495.250000	967.964709

[76]:
```
v["predicted closing"]=pred
```

[77]:
```
v.to_excel("Group4_validation_with_predicted_closing.xlsx")
```