In [133]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
```

In [138]:
```
data = pd.read_csv('C:/Users/Lenovo/Downloads/new.csv')
data.drop('date',axis=1,inplace=True)
data.drop('symbol',axis=1,inplace=True)
print(data.head(10))
data.head()
```

```
       open       close         low        high   volume
0  123.430000  125.839996  122.309998  126.250000  2163600
1  125.239998  119.980003  119.940002  125.540001  2386400
2  116.379997  114.949997  114.930000  119.739998  2489500
3  115.480003  116.620003  113.500000  117.440002  2006300
4  117.010002  114.970001  114.089996  117.330002  1408600
5  115.510002  115.550003  114.500000  116.059998  1098000
6  116.459999  112.849998  112.589996  117.070000   949600
7  113.510002  114.379997  110.050003  115.029999   785300
8  113.350002  112.529999  111.919998  114.879997  1093700
9  113.660004  110.379997  109.870003  115.870003  1523500
```

Out[138]:

	open	close	low	high	volume
0	123.430000	125.839996	122.309998	126.250000	2163600
1	125.239998	119.980003	119.940002	125.540001	2386400
2	116.379997	114.949997	114.930000	119.739998	2489500
3	115.480003	116.620003	113.500000	117.440002	2006300
4	117.010002	114.970001	114.089996	117.330002	1408600

In [123]:
```
print(df.describe())
```

```
            open  close  low  high  volume
count    0.0    0.0    0.0  0.0   0.0     0.0
mean     NaN    NaN    NaN  NaN   NaN     NaN
std      NaN    NaN    NaN  NaN   NaN     NaN
min      NaN    NaN    NaN  NaN   NaN     NaN
25%      NaN    NaN    NaN  NaN   NaN     NaN
50%      NaN    NaN    NaN  NaN   NaN     NaN
75%      NaN    NaN    NaN  NaN   NaN     NaN
max      NaN    NaN    NaN  NaN   NaN     NaN
```

In [124]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3482 entries, 0 to 3481
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   open    3482 non-null   float64
 1   close   3482 non-null   float64
 2   low     3482 non-null   float64
 3   high    3482 non-null   float64
 4   volume  3482 non-null   int64
dtypes: float64(4), int64(1)
memory usage: 136.1 KB
None
```

In [96]:
```
data1 = pd.read_csv('C:/Users/Lenovo/Downloads/new1.csv')
print(data1.head(20))
```

```
    Unnamed: 0        date symbol       open       close        low  \
0            0  05-01-2016 00:00   WLTW  123.430000  125.839996  122.309998
1            1  06-01-2016 00:00   WLTW  125.239998  119.980003  119.940002
2            2  07-01-2016 00:00   WLTW  116.379997  114.949997  114.930000
3            3  08-01-2016 00:00   WLTW  115.480003  116.620003  113.500000
4            4  11-01-2016 00:00   WLTW  117.010002  114.970001  114.089996
5            5  12-01-2016 00:00   WLTW  115.510002  115.550003  114.500000
6            6  13-01-2016 00:00   WLTW  116.459999  112.849998  112.589996
7            7  14-01-2016 00:00   WLTW  113.510002  114.379997  110.050003
8            8  15-01-2016 00:00   WLTW  113.330002  112.529999  111.919998
9            9  19-01-2016 00:00   WLTW  113.660004  110.379997  109.870003
10          10  20-01-2016 00:00   WLTW  109.059998  109.300003  108.320000
11          11  21-01-2016 00:00   WLTW  109.730003  110.000000  108.320000
12          12  22-01-2016 00:00   WLTW  111.879997  111.949997  110.190002
13          13  25-01-2016 00:00   WLTW  111.320000  110.120003  110.000000
14          14  26-01-2016 00:00   WLTW  110.419998  111.000000  107.300003
15          15  27-01-2016 00:00   WLTW  110.769997  110.709999  109.019997
16          16  28-01-2016 00:00   WLTW  110.900002         NaN  109.900002
17          17  29-01-2016 00:00   WLTW  113.349998  114.470001  111.669998
18          18  01-02-2016 00:00   WLTW  114.000000  114.500000  112.900002
19          19  02-02-2016 00:00   WLTW  113.250000  110.559998  109.750000

        high   volume
0  126.250000  2163600
1  125.540001  2386400
2  119.739998  2489500
3  117.440002  2006300
4         NaN  1408600
5  116.059998  1098000
6  117.070000   949600
7  115.029999   785300
8  114.879997  1093700
9  115.870003  1523500
10         NaN  1653900
11 110.580002   944300
12 112.949997   744900
13 114.629997   703800
14 111.400002   563100
15 112.570000   896100
16 112.970001   688400
17 114.589996   749900
18 114.849998   574200
19         NaN   694800
```

In [97]:
```
print(data1.describe())
```

```
       Unnamed: 0        open       close         low        high  \
count   50.000000   50.000000   49.000000   50.000000   47.000000
mean    24.50000  113.906800  113.966122  112.023400  115.461915
std     14.57738    3.892047    3.907816    3.857870    3.718608
min      0.00000  105.629997  107.129997  104.110001  109.260002
25%     12.25000  111.207498  111.000000  109.657499  112.685001
50%     24.50000  113.364998  113.320000  111.180000  114.879997
75%     36.75000  116.439999  116.620003  114.822500  117.405002
max     49.00000  125.239998  125.839996  122.309998  126.250000

             volume
count  5.000000e+01
mean   9.988880e+05
std    5.144256e+05
min    4.112000e+05
25%    6.806250e+05
50%    8.962000e+05
75%    1.178725e+06
max    2.489500e+06
```

In [98]:
```
print(data1.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  50 non-null     int64
 1   date        50 non-null     object
 2   symbol      50 non-null     object
 3   open        50 non-null     float64
 4   close       49 non-null     float64
 5   low         50 non-null     float64
 6   high        47 non-null     float64
 7   volume      50 non-null     int64
dtypes: float64(4), int64(2), object(2)
memory usage: 3.2+ KB
None
```

In [126]:
```
data1.dropna(inplace=True)
data1.fillna(-9999,inplace=True)
data1['high'] = data1['high'].fillna(data1['high'].mean())
print(data1.head())
```

```
   Unnamed: 0        date symbol       open       close        low  \
0           0  05-01-2016 00:00   WLTW  123.430000  125.839996  122.309998
1           1  06-01-2016 00:00   WLTW  125.239998  119.980003  119.940002
2           2  07-01-2016 00:00   WLTW  116.379997  114.949997  114.930000
3           3  08-01-2016 00:00   WLTW  115.480003  116.620003  113.500000
5           5  12-01-2016 00:00   WLTW  115.510002  115.550003  114.500000

        high   volume
0  126.250000  2163600
1  125.540001  2386400
2  119.739998  2489500
3  117.440002  2006300
5  116.059998  1098000
```

In [91]:
```
#fill the value by fillna()
data1.dropna(inplace=True)
data1.fillna(9999,inplace=True)
print(data1.head(10))
```

```
   Unnamed: 0        date symbol       open       close        low  \
0           0  05-01-2016 00:00   WLTW  123.430000  125.839996  122.309998
1           1  06-01-2016 00:00   WLTW  125.239998  119.980003  119.940002
2           2  07-01-2016 00:00   WLTW  116.379997  114.949997  114.930000
3           3  08-01-2016 00:00   WLTW  115.480003  116.620003  113.500000
5           5  12-01-2016 00:00   WLTW  115.510002  115.550003  114.500000
6           6  13-01-2016 00:00   WLTW  116.459999  112.849998  112.589996
7           7  14-01-2016 00:00   WLTW  113.510002  114.379997  110.050003
8           8  15-01-2016 00:00   WLTW  113.330002  112.529999  111.919998
9           9  19-01-2016 00:00   WLTW  113.660004  110.379997  109.870003
11         11  21-01-2016 00:00   WLTW  109.730003  110.000000  108.320000

        high   volume
0  126.250000  2163600
1  125.540001  2386400
2  119.739998  2489500
3  117.440002  2006300
5  116.059998  1098000
7  115.029999   785300
8  114.879997  1093700
9  115.870003  1523500
11 110.580002   944300
```

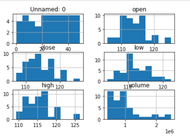
In [100]:
```
#fill the value with mean
data1['high'] = data1['high'].fillna(data1['high'].mean())
print(data1.head())
```

```
   Unnamed: 0        date symbol       open       close        low  \
0           0  05-01-2016 00:00   WLTW  123.430000  125.839996  122.309998
1           1  06-01-2016 00:00   WLTW  125.239998  119.980003  119.940002
2           2  07-01-2016 00:00   WLTW  116.379997  114.949997  114.930000
3           3  08-01-2016 00:00   WLTW  115.480003  116.620003  113.500000
5           5  12-01-2016 00:00   WLTW  115.510002  115.550003  114.500000

        high   volume
0  126.250000  2163600
1  125.540001  2386400
2  119.739998  2489500
3  117.440002  2006300
5  116.059998  1098000
```

In [ ]:

In [110]:
```
data.hist()
plt.show()
```



In [111]:
```
plt.scatter(data.close,data.open,color="blue")
plt.show()
```

In [115]:
```
#to find the correlation value for each column
cor =data.corr()
print(cor['open'].sort_values(ascending=True))
```

```
Unnamed: 0    0.070603
volume        0.161164
close         0.927620
low           0.949430
high          0.966265
open          1.000000
Name: open, dtype: float64
```

TO PREDICT THE STOCK MARKET AFTER 10 DAYS

In [120]:
```
forecast =10
data['label']=data['open'].shift(-forecast)
print(data.head())
```

```
   Unnamed: 0       open       close         low        high  volume  \
0           0  123.430000  125.839996  122.309998  126.250000  2163600
1           1  125.239998  119.980003  119.940002  125.540001  2386400
2           2  116.379997  114.949997  114.930000  119.739998  2489500
3           3  115.480003  116.620003  113.500000  117.440002  2006300
5           5  115.510002  115.550003  114.500000  116.059998  1098000

        label
0  111.879997
1  111.320000
2  110.419998
3  110.769997
5  113.349998
```

In [127]:
```
data.dropna(inplace=True)
print(data.head())
```

```
       open       close         low        high   volume
0  123.430000  125.839996  122.309998  126.250000  2163600
1  125.239998  119.980003  119.940002  125.540001  2386400
2  116.379997  114.949997  114.930000  119.739998  2489500
3  115.480003  116.620003  113.500000  117.440002  2006300
4  117.010002  114.970001  114.089996  117.330002  1408600
```

In [137]:
```
forecast =10
data['label']=data['open'].shift(-forecast)
print(data.head())
x =np.array(data.drop(['label'],1))
y = np.array(data['label'])
x =preprocessing.scale(x)
#print(x)
#print(y)
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.15,random_state=4)
print(len(x_train))
print(len(x_test))
```

```
       open       close         low        high   volume       label
0  123.430000  125.839996  122.309998  126.250000  2163600  109.059998
1  125.239998  119.980003  119.940002  125.540001  2386400  109.730003
2  116.379997  114.949997  114.930000  119.739998  2489500  111.879997
3  115.480003  116.620003  113.500000  117.440002  2006300  111.320000
4  117.010002  114.970001  114.089996  117.330002  1408600  110.419998
2959
523
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_64300\1823541486.py:4: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' wi
ll be keyword-only.
  x =np.array(data.drop(['label'],1))
```

In [ ]: