Project1: Stock marketing prediction

In [38]:
```
import numpy as np #import the numerical python for mathematics calculate
import pandas as pd #import pandas for dataframes
import matplotlib.pyplot as plt #import matplotlib for visulization
```

In [39]:
```
tesla=pd.read_csv("tesla.csv")#using tesla stock marketing prediction
tesla
```

Out[39]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	29-06-2010	19.000000	25.000000	17.540001	23.889999	23.889999	18766300
1	30-06-2010	25.790001	30.420000	23.299999	23.830000	23.830000	17187100
2	01-07-2010	25.000000	25.920000	20.270000	21.959999	21.959999	8218800
3	02-07-2010	23.000000	23.100000	18.709999	19.200001	19.200001	5139800
4	06-07-2010	20.000000	20.000000	15.830000	16.110001	16.110001	6866900
...	...	...	...	...	...	...	...
2188	11-03-2019	283.519989	291.279999	280.500000	290.920013	290.920013	7392300
2189	12-03-2019	286.489990	288.070007	281.059998	283.359985	283.359985	7504100
2190	13-03-2019	283.899994	291.989990	282.700012	288.959991	288.959991	6844700
2191	14-03-2019	292.450012	295.390015	288.290009	289.959991	289.959991	7074200
2192	15-03-2019	283.510010	283.723999	274.399994	275.429993	275.429993	14758243

2193 rows × 7 columns

In [40]:
```
tesla.info#information about data
```

Out[40]:
```
<bound method DataFrame.info of            Date        Open        High         Low       Close   Adj Close  \
0     29-06-2010   19.000000   25.000000   17.540001   23.889999   23.889999
1     30-06-2010   25.790001   30.420000   23.299999   23.830000   23.830000
2     01-07-2010   25.000000   25.920000   20.270000   21.959999   21.959999
3     02-07-2010   23.000000   23.100000   18.709999   19.200001   19.200001
4     06-07-2010   20.000000   20.000000   15.830000   16.110001   16.110001
...          ...         ...         ...         ...         ...         ...
2188  11-03-2019  283.519989  291.279999  280.500000  290.920013  290.920013
2189  12-03-2019  286.489990  288.070007  281.059998  283.359985  283.359985
2190  13-03-2019  283.899994  291.989990  282.700012  288.959991  288.959991
2191  14-03-2019  292.450012  295.390015  288.290009  289.959991  289.959991
2192  15-03-2019  283.510010  283.723999  274.399994  275.429993  275.429993

        Volume
0     18766300
1     17187100
2      8218800
3      5139800
4      6866900
...        ...
2188   7392300
2189   7504100
2190   6844700
2191   7074200
2192  14758243
```

In [41]:
```
tesla.describe()#describe the dataset
```
Out[41]:

	Open	High	Low	Close	Adj Close	Volume
count	2193.000000	2193.000000	2193.000000	2193.000000	2193.000000	2.193000e+03
mean	175.652882	178.710262	172.412075	175.648555	175.648555	5.077449e+06
std	115.580903	117.370092	113.654794	115.580771	115.580771	4.545398e+06
min	16.139999	16.629999	14.980000	15.800000	15.800000	1.185000e+05
25%	33.110001	33.910000	32.459999	33.160000	33.160000	1.577800e+06
50%	204.990005	208.160004	201.669998	204.990005	204.990005	4.171700e+06
75%	262.000000	265.329987	256.209991	261.739990	261.739990	6.885600e+06
max	386.690002	389.609985	379.350006	385.000000	385.000000	3.716390e+07

In [42]:
```
from sklearn.model_selection import train_test_split#import train_test_split

from sklearn.preprocessing import MinMaxScaler#import Minimum and maximum scaler
from sklearn.preprocessing import StandardScaler#import StandardScaler
from sklearn.metrics import mean_squared_error as mse#import mean squared error
from sklearn.metrics import r2_score# import r2_score find the accurancy
```
In [43]:
```
X=np.array(tesla.index).reshape(-1,1)#slpit dataset into train data and test data
Y=tesla['Close']
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state=101)
```
In [44]:
```
X_train# X_train dataset
```
Out[44]:
```
array([[ 365],
       [1111],
       [1581],
       ...,
       [1361],
       [1547],
       [ 863]], dtype=int64)
```
In [45]:
```
X_test# X_test dataset
```
Out[45]:
```
array([[ 925],
       [1151],
       [1378],
       [2079],
       [ 762],
       [ 330],
       [ 254],
       [ 900],
       [1328],
       [1440],
       [1869],
       [ 452],
       [ 482],
       [1660],
       [ 240],
       [1373],
       [1084],
       [1243],
       [1258],
```

In [46]:
```
Y_train#Y_train dataset
```

Out[46]:
```
365      34.189999
1111    248.089996
1581    196.610001
1990    277.850006
1753    375.339996
           ...
599      31.610001
1599    188.020004
1361    217.750000
1547    225.000000
863     124.169998
Name: Close, Length: 1535, dtype: float64
```

In [47]:
```
Y_test#Y_test dataset
```

Out[47]:
```
925     254.839996
1151    206.550003
1378    233.389999
2079    310.700012
762     122.269997
           ...
1786    325.890015
193      25.830000
162      23.600000
1063    263.820007
543      29.950001
Name: Close, Length: 658, dtype: float64
```

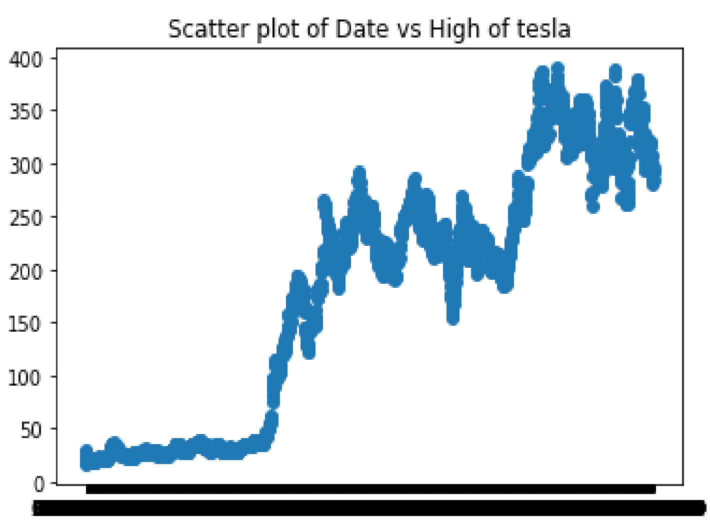
In [48]:
```
scaler=StandardScaler().fit(X_train)
```

In [49]:
```
from sklearn.linear_model import LinearRegression
```

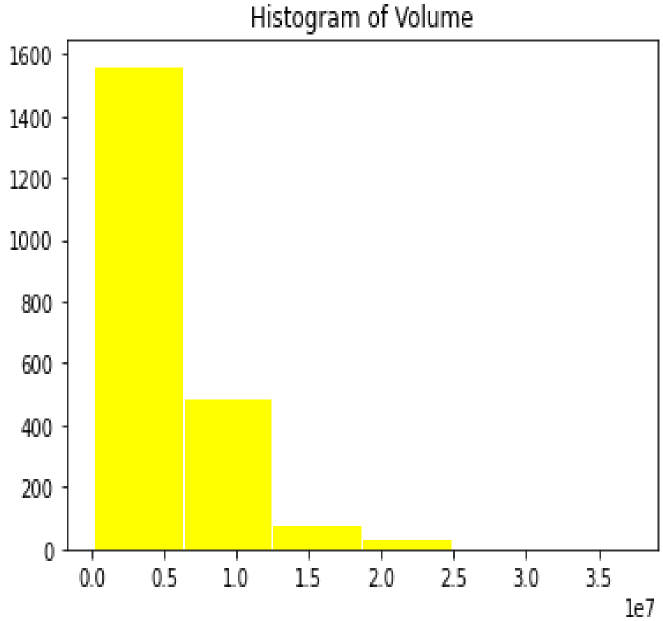
In [50]:
```
lm=LinearRegression()
lm.fit(X_train,Y_train)
```

Out[50]:
```
LinearRegression()
```

In [58]:
```
plt.scatter(x=tesla['Date'],y=tesla['High'])
plt.title('Scatter plot of Date vs High of tesla')
plt.show()
```



In [59]:
```
plt.hist(tesla['Volume'],color='yellow',edgecolor='white',bins=6)
plt.title('Histogram of Volume')
plt.show()
```



In [60]:
```
scores=f'''
{'Metric'.ljust(10)}{'Train'.center(20)}{'Test'.center(20)}
{'r2_score'.ljust(10)}{r2_score(Y_train,lm.predict(X_train))}\t{r2_score(Y_test,lm.predict(X_test))}
{'MSE'.ljust(10)}{mse(Y_train,lm.predict(X_train))}\t{mse(Y_test,lm.predict(X_test))}
'''
print(scores)
```

```
Metric             Train                Test
r2_score  0.8658871776828707    0.8610649253244574
MSE       1821.3833862936174    1780.987539418845
```