In [8]:

```
import matplotlib.pyplot as plt
from sklearn import metrics
import seaborn as sns
import numpy as np
import pandas as pd


df = pd.read_csv("G:\iNurture_Rathinam\Machine Learning - M.Sc DSBA\Lab\Bitcoin Prediciton\
```

In [10]:

```
df.head()
```

Out[10]:

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
0	1325317920	4.39	4.39	4.39	4.39	0.455581	2.0	4.39
1	1325317980	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1325318040	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1325318100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1325318160	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [11]:

```
df.shape
```

Out[11]:

```
(4857377, 8)
```

In [12]:

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

Out[12]:

```
Timestamp                  0
Open                 1243608
High                 1243608
Low                  1243608
Close                1243608
Volume_(BTC)         1243608
Volume_(Currency)    1243608
Weighted_Price       1243608
dtype: int64
```

In [14]:

```
# convert date string to timestamp

df['Dates'] = pd.to_datetime(df['Timestamp'], unit='s')
df.head()
```

Out[14]:

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price	
0	1325317920	4.39	4.39	4.39	4.39	0.455581	2.0	4.39	07
1	1325317980	NaN	NaN	NaN	NaN	NaN	NaN	NaN	07
2	1325318040	NaN	NaN	NaN	NaN	NaN	NaN	NaN	07
3	1325318100	NaN	NaN	NaN	NaN	NaN	NaN	NaN	07
4	1325318160	NaN	NaN	NaN	NaN	NaN	NaN	NaN	07

In [20]:

```
# remove missing values

df.dropna(inplace=True)
df.shape
```

Out[20]:

```
(3613769, 9)
```

In [21]:

```
df.head()
```

Out[21]:

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
0	1325317920	4.39	4.39	4.39	4.39	0.455581	2.000000	4.390000
478	1325346600	4.39	4.39	4.39	4.39	48.000000	210.720000	4.390000
547	1325350740	4.50	4.57	4.50	4.57	37.862297	171.380338	4.526411
548	1325350800	4.58	4.58	4.58	4.58	9.000000	41.220000	4.580000
1224	1325391360	4.58	4.58	4.58	4.58	1.502000	6.879160	4.580000

In [28]:

```
features = ['Open', 'High', 'Low', 'Volume_(BTC)', 'Volume_(Currency)', 'Weighted_Price']
target = 'Close'
```

In [36]:

```
# Train and Test set

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df[features],df[target],test_size = 0.3

print("xtrain shape : ", X_train.shape)
print("xtest shape  : ", X_test.shape)
print("ytrain shape : ", y_train.shape)
print("ytest shape  : ", y_test.shape)
```

```
xtrain shape :  (2529638, 6)
xtest shape  :  (1084131, 6)
ytrain shape :  (2529638,)
ytest shape  :  (1084131,)
```

In [44]:

```
# Linear Regression

from sklearn.linear_model import LinearRegression

Linear_Model = LinearRegression()
Linear_Model_Fit = Linear_Model.fit(X_train, y_train)

print("Actual values are : \n",y_test)

# # predicting the test set result
pred = Linear_Model.predict(X_test)
print("\nPredicted values are : \n",pred)
```

```
Actual values are : 
 587056        23.75
863914       112.18
601037        29.89
3938893    10298.82
1371166      586.66
             ...   
2492210      604.70
3595276     6399.30
2861850     2757.57
1993848      265.82
1037180      685.00
Name: Close, Length: 1084131, dtype: float64

Predicted values are : 
 [  23.71498299  112.17397448   29.88067928 ... 2756.27687043  265.80576865
  684.19279839]
```

In [46]:

```
df[features]
```

Out[46]:

	Open	High	Low	Volume_(BTC)	Volume_(Currency)	Weighted_Price
0	4.39	4.39	4.39	0.455581	2.000000	4.390000
478	4.39	4.39	4.39	48.000000	210.720000	4.390000
547	4.50	4.57	4.50	37.862297	171.380338	4.526411
548	4.58	4.58	4.58	9.000000	41.220000	4.580000
1224	4.58	4.58	4.58	1.502000	6.879160	4.580000
...	...	...	...	...	...	...
4857372	58714.31	58714.31	58686.00	1.384487	81259.372187	58692.753339
4857373	58683.97	58693.43	58683.97	7.294848	428158.146640	58693.226508
4857374	58693.43	58723.84	58693.43	1.705682	100117.070370	58696.198496
4857375	58742.18	58770.38	58742.18	0.720415	42332.958633	58761.866202
4857376	58767.75	58778.18	58755.97	2.712831	159417.751000	58764.349363

3613769 rows × 6 columns

In [54]:

```
# Coefficients:

lreg_coefficient = pd.DataFrame()
lreg_coefficient["Columns"] = features
lreg_coefficient['Coefficient Estimate'] = pd.Series(Linear_Model.coef_)
print("\n Coefficients Estimate by Linear Model \n")
print(lreg_coefficient)
print("\n")
print("Intercept:", Linear_Model.intercept_)
```

```
 Coefficients Estimate by Linear Model 

            Columns  Coefficient Estimate
0              Open         -4.846740e-01
1              High          6.745226e-01
2               Low          5.767766e-01
3      Volume_(BTC)         -1.368262e-03
4  Volume_(Currency)          8.857309e-08
5    Weighted_Price          2.333281e-01


Intercept: -0.004433348483871669
```

In [45]:

```
# Metrics

from sklearn.metrics import mean_squared_error, r2_score

MSE = mean_squared_error(y_test, pred)
print("Mean Square Error : ", MSE)

R2 = r2_score(y_test, pred)
print("R2 Error: ",R2)

print('RMSE:', np.sqrt(mean_squared_error(y_test, pred)))
```

```
Mean Square Error :  50.812355776440214
R2 Error:  0.999999372987545
RMSE: 7.1282785984022965
```

In [ ]: