In [74]:

```
#import all necessary Libraries

import pandas as pd
import numpy as np
import sklearn as sl
from sklearn.decomposition import PCA
from sklearn import metrics
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn import preprocessing
```

In [18]:

```
pip install xgboost
```

```
Requirement already satisfied: xgboost in c:\programdata\ssh\lib\site-pack
ages (1.2.0)
Requirement already satisfied: scipy in c:\programdata\ssh\lib\site-packag
es (from xgboost) (1.5.0)
Requirement already satisfied: numpy in c:\programdata\ssh\lib\site-packag
es (from xgboost) (1.18.5)
Note: you may need to restart the kernel to use updated packages.
```

In [19]:

```
import xgboost as xgb
```

In [20]:

```
#reading the datasets and explore it
#train dataset

df_train=pd.read_csv("train.csv")
df_test=pd.read_csv("test.csv")
print('train shape',df_train.shape)
print('test shape',df_test.shape)
```

```
train shape (4209, 378)
test shape (4209, 377)
```

In [21]:

```
df_train.head()
```

Out[21]:

	ID	y	X0	X1	X2	X3	X4	X5	X6	X8	...	X375	X376	X377	X378	X379	X380
0	0	130.81	k	v	at	a	d	u	j	o	...	0	0	1	0	0	0
1	6	88.53	k	t	av	e	d	y	l	o	...	1	0	0	0	0	0
2	7	76.26	az	w	n	c	d	x	j	x	...	0	0	0	0	0	0
3	9	80.62	az	t	n	f	d	x	l	e	...	0	0	0	0	0	0
4	13	78.02	az	v	n	f	d	h	d	n	...	0	0	0	0	0	0

5 rows × 378 columns

In [22]:

```
df_test.head()
```

Out[22]:

	ID	X0	X1	X2	X3	X4	X5	X6	X8	X10	...	X375	X376	X377	X378	X379	X380	X3
0	1	az	v	n	f	d	t	a	w	0	...	0	0	0	1	0	0	
1	2	t	b	ai	a	d	b	g	y	0	...	0	0	1	0	0	0	
2	3	az	v	as	f	d	a	j	j	0	...	0	0	0	1	0	0	
3	4	az	l	n	f	d	z	l	n	0	...	0	0	0	1	0	0	
4	5	w	s	as	c	d	y	i	m	0	...	1	0	0	0	0	0	

5 rows × 377 columns

In [23]:

```
df_train.dtypes
```

Out[23]:

```
ID         int64
y        float64
X0        object
X1        object
X2        object
          ...
X380       int64
X382       int64
X383       int64
X384       int64
X385       int64
Length: 378, dtype: object
```

In [24]:

```
df_test.dtypes
```

Out[24]:

```
ID         int64
X0        object
X1        object
X2        object
X3        object
           ...  
X380       int64
X382       int64
X383       int64
X384       int64
X385       int64
Length: 377, dtype: object
```

In [25]:

```
#q2checking for the null value
df_train.isnull().sum()
```

Out[25]:

```
ID      0
y       0
X0      0
X1      0
X2      0
       ..
X380    0
X382    0
X383    0
X384    0
X385    0
Length: 378, dtype: int64
```

In [26]:

```
df_test.isnull().sum()
```

Out[26]:

```
ID      0
X0      0
X1      0
X2      0
X3      0
       ..
X380    0
X382    0
X383    0
X384    0
X385    0
Length: 377, dtype: int64
```

In [27]:

```
dtype_df = df_train.dtypes.reset_index()
dtype_df.columns = ["Count", "Column Type"]
dtype_df.groupby("Column Type").aggregate('count').reset_index()
```

Out[27]:

	Column Type	Count
0	int64	369
1	float64	1
2	object	8

In [28]:

```
dtype_df = df_test.dtypes.reset_index()
dtype_df.columns = ["Count", "Column Type"]
dtype_df.groupby("Column Type").aggregate('count').reset_index()
```

Out[28]:

	Column Type	Count
0	int64	369
1	object	8

In [29]:

```
#datatype of columns
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4209 entries, 0 to 4208
Columns: 378 entries, ID to X385
dtypes: float64(1), int64(369), object(8)
memory usage: 12.1+ MB
```

In [31]:

```
#datatype of columns
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4209 entries, 0 to 4208
Columns: 377 entries, ID to X385
dtypes: int64(369), object(8)
memory usage: 12.1+ MB
```

In [33]:

```
#splitting the dataset into X and y variable
from sklearn.model_selection import train_test_split



X_train = df_train.drop(['y','ID'],axis = 1)
y_train = df_train['y']
X=X_train
y=y_train
```

In [34]:

```
print('Train dataset shape = ', X_train.shape)
print('Train labels shape = ', y_train.shape)
```

```
Train dataset shape =  (4209, 376)
Train labels shape =  (4209,)
```

In [35]:

```
X_test = df_test.drop(['ID'], axis = 1)
y_test = df_test['ID']
```

In [37]:

```
print('Test dataset shape = ', X_test.shape)
print('Test labels shape = ', y_test.shape)
```

```
Test dataset shape =  (4209, 376)
Test labels shape =  (4209,)
```

In [43]:

```
X_train.corr()
```

Out[43]:

	X10	X12	X13	X14	X15	X16	X17	X18
X10	1.000000	-0.033084	-0.028806	-0.100474	-0.002532	-0.005944	-0.010164	-0.010323
X12	-0.033084	1.000000	0.214825	-0.246513	-0.006212	-0.014584	-0.024937	-0.025327
X13	-0.028806	0.214825	1.000000	-0.083141	-0.005409	-0.012698	-0.021713	-0.010525
X14	-0.100474	-0.246513	-0.083141	1.000000	-0.018865	-0.044291	0.012713	-0.076916
X15	-0.002532	-0.006212	-0.005409	-0.018865	1.000000	-0.001116	-0.001908	-0.001938
...	...	...	...	...	...	...	...	...
X380	-0.010479	-0.005566	0.023045	0.007743	-0.001968	-0.004619	-0.007899	-0.008022
X382	-0.010164	-0.024937	-0.021713	0.012713	-0.001908	-0.004480	1.000000	0.085256
X383	-0.004740	-0.011628	-0.010125	0.023604	-0.000890	-0.002089	-0.003572	0.062481
X384	-0.002532	-0.006212	0.041242	0.025199	-0.000475	-0.001116	-0.001908	-0.001938
X385	-0.004387	-0.010765	-0.009373	0.043667	-0.000824	-0.001934	-0.003307	-0.003359

356 rows × 356 columns

In [44]:

```
X_test.corr()
```

Out[44]:

	X10	X11	X12	X13	X14	X15	X16	X17
X10	1.000000	-0.002146	-0.039453	-0.035496	-0.120379	-0.003717	-0.007125	-0.013108
X11	-0.002146	1.000000	-0.004369	-0.003931	0.017825	-0.000412	-0.000789	-0.001452
X12	-0.039453	-0.004369	1.000000	0.283228	-0.245127	-0.007570	-0.014509	-0.016991
X13	-0.035496	-0.003931	0.283228	1.000000	-0.076145	-0.006811	-0.013054	-0.024015
X14	-0.120379	0.017825	-0.245127	-0.076145	1.000000	-0.023097	-0.044269	0.000864
...	...	...	...	...	...	...	...	...
X380	-0.012561	-0.001391	-0.025578	0.054582	0.007787	-0.002410	-0.004619	-0.008498
X382	-0.013108	-0.001452	-0.016991	-0.024015	0.000864	-0.002515	-0.004821	1.000000
X383	-0.003035	-0.000336	-0.006180	-0.005560	0.025212	-0.000582	-0.001116	-0.002053
X384	-0.003717	-0.000412	-0.007570	-0.006811	0.030881	-0.000713	-0.001367	-0.002515
X385	-0.005681	-0.000629	-0.011569	-0.010408	0.047195	-0.001090	-0.002089	-0.003844

368 rows × 368 columns

In [47]:

```
print(X_train.shape)
print (X_test.shape)
print (y_train.shape)
print(y_test.shape)
```

```
(4209, 364)
(4209, 376)
(4209,)
(4209,)
```

In [48]:

```
#label encodcer

from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [77]:

```
df_train.apply(LabelEncoder().fit_transform)
```

Out[77]:

	ID	y	X0	X1	X2	X3	X4	X5	X6	X8	...	X375	X376	X377	X378	X379	X3
0	0	2466	32	23	17	0	3	24	9	14	...	0	0	1	0	0	
1	1	366	32	21	19	4	3	28	11	14	...	1	0	0	0	0	
2	2	69	20	24	34	2	3	27	9	23	...	0	0	0	0	0	
3	3	133	20	21	34	5	3	27	11	4	...	0	0	0	0	0	
4	4	106	20	23	34	5	3	12	3	13	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4204	4204	1657	8	20	16	2	3	0	3	16	...	1	0	0	0	0	
4205	4205	1766	31	16	40	3	3	0	7	7	...	0	1	0	0	0	
4206	4206	1801	8	23	38	0	3	0	6	4	...	0	0	1	0	0	
4207	4207	280	9	19	25	5	3	0	11	20	...	0	0	0	0	0	
4208	4208	1921	46	19	3	2	3	0	6	22	...	1	0	0	0	0	

4209 rows × 378 columns

In [50]:

```
df_test.apply(LabelEncoder().fit_transform)
```

Out[50]:

	ID	X0	X1	X2	X3	X4	X5	X6	X8	X10	...	X375	X376	X377	X378	X379	X38
0	0	21	23	34	5	3	26	0	22	0	...	0	0	0	1	0	
1	1	42	3	8	0	3	9	6	24	0	...	0	0	1	0	0	
2	2	21	23	17	5	3	0	9	9	0	...	0	0	0	1	0	
3	3	21	13	34	5	3	31	11	13	0	...	0	0	0	1	0	
4	4	45	20	17	2	3	30	8	12	0	...	1	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4204	4204	6	9	17	5	3	1	9	4	0	...	0	0	0	0	0	
4205	4205	42	1	8	3	3	1	9	24	0	...	0	1	0	0	0	
4206	4206	47	23	17	5	3	1	3	22	0	...	0	0	0	0	0	
4207	4207	7	23	17	0	3	1	2	16	0	...	0	0	1	0	0	
4208	4208	42	1	8	2	3	1	6	17	0	...	1	0	0	0	0	

4209 rows × 377 columns

In [ ]:

```

```

In [ ]:

```
#Dta preprocessing
```

In [72]:

```
def one_hot(train_data,test_data,columns):

    for i,column in enumerate(columns):
        Xtrain = train_data[str(column)].T
        Xtest = test_data[str(column)].T

        # train_df
        lb=preprocessing.LabelBinarizer()
        lb.fit(Xtrain)
        X_classes = len(lb.classes_)
        Xenc = lb.transform(Xtrain)
        Xtrain_enc = pd.DataFrame(data = Xenc, columns = lb.classes_)
        train_data.drop([str(column)], axis =1, inplace=True)

        # test_df
        Xenc = lb.transform(Xtest)
        Xtest_enc = pd.DataFrame(data = Xenc, columns = lb.classes_)
        test_data.drop([str(column)], axis =1, inplace=True)

        print('Number of classes in '+str(column)+ ' = '+ str(X_classes))
        train_data = pd.concat((train_data,Xtrain_enc),axis=1)
        test_data = pd.concat((test_data,Xtest_enc),axis=1)
    return train_data,test_data
```

In [75]:

```
train_data , test_data = one_hot(X_train,X_test,['X0', 'X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X8'])
```

```
Number of classes in X0 = 47
Number of classes in X1 = 27
Number of classes in X2 = 44
Number of classes in X3 = 7
Number of classes in X4 = 4
Number of classes in X5 = 29
Number of classes in X6 = 12
Number of classes in X8 = 25
```

In [83]:

```
pca = PCA(random_state=21)
pca.fit(train_data)
pca_train = pca.transform(train_data)
```

In [84]:

```
X = pca_train
y = y_train
X.shape,y.shape
```

Out[84]:

```
((4209, 551), (4209,))
```

In [98]:

```
#building model

from sklearn.linear_model import LinearRegression

model_lr=LinearRegression()
model_lr=LinearRegression.fit(model_lr,X,y)
model_lr.fit(X,y)
```

Out[98]:

```
LinearRegression()
```

from sklearn.linear_model import LinearRegression

model_lr=LinearRegression() model_lr=LinearRegression.fit(model_lr,X,y) model_lr.fit()

In [99]:

```
print ("The intercept of the LR model is: ", model_lr.intercept_ )
```

```
The intercept of the LR model is:  100.67076162730552
```

In [100]:

```
print ("The regression coefficient value for all the features are : ", model_lr.coef_)
```

```
The regression coefficient value for all the features are :  [-1.19791113e
+00  2.80540466e+00  1.42630589e+00  8.42887878e-01
  2.61870015e+00 -3.08058000e+00  3.02907097e+00 -8.22961450e-01
  2.71383882e+00 -4.27813202e+00  2.54132225e+00  1.28410299e+00
  1.01693833e+00  1.99650437e-01  1.33830146e+00 -1.32973552e-01
  6.56755276e-01  2.26468833e-01 -1.16097629e+00 -1.53115573e+00
  1.29875479e+00  8.32181424e-02  7.35318359e-01 -6.12625271e-01
 -7.30581310e-01 -7.11957425e-01 -5.14735196e-01 -2.95661896e-01
 -1.42307436e+00  8.36950533e-01 -9.24282908e-01 -5.40952764e-01
 -2.33931690e-01  7.01770373e-02  9.60962921e-02 -8.35632108e-01
  9.37783778e-01 -1.44660750e+00  3.01958621e-03  1.79151818e-01
  1.02480870e+00  1.63362779e+00 -2.09326654e+00 -1.08415608e+00
 -1.23397484e-01  3.42372015e-01  3.09955835e-01 -1.21379673e+00
 -1.63088235e+00 -2.83592343e-02  2.73846179e+00  3.50950524e-01
  1.52844778e+00 -4.53091599e-01  1.55177417e+00 -3.42069902e-01
  1.36710104e+00  2.41039617e+00 -5.07413931e-02  1.59504043e+00
 -1.29199222e-01  1.36340420e+00 -1.81057096e-01  1.02164443e+00
 -1.62224542e+00  3.37068520e-01 -1.06990264e+00  2.37138027e-01
  1.85196322e-01 -1.61215179e-02 -3.60828090e-01  1.88732010e+00
 -1.79782967e-01  4.12670020e-01 -2.53285836e-01  7.24768807e-01
  1.94126035e+00 -2.18742859e+00 -1.08629860e+00  2.12512471e+00
  1.55364723e+00  1.03775845e+00  2.78958406e-01 -8.00408661e-01
  8.40508571e-01 -1.08156480e+00  5.50154392e-01  2.16198257e+00
 -2.32791573e-01  3.06322062e+00  6.47464639e-01 -1.00441024e+00
  2.80751541e-01 -1.17119125e+00  2.47721713e-01 -8.97610545e-01
 -3.23539402e+00 -1.20605617e+00 -2.40303870e-01 -6.81703314e-02
  1.82063641e+00  1.06197495e+00  1.93358825e+00  1.25684143e+00
  2.38713011e-01  2.51662797e+00 -1.19384849e+00  6.98761073e-01
  6.43716402e-01  7.41222233e-01 -1.21191700e+00 -3.80670905e-01
 -1.04730299e+00  1.32933420e+00 -2.74220653e+00  1.44471893e+00
 -1.70490892e+00 -1.59384197e+00  1.81662606e+00  1.58768694e+00
  6.56452965e-01 -3.13625486e+00  2.01773310e+00  8.16467851e-01
  1.30450267e+00  1.22217549e+00  1.60097610e+00 -1.33897626e+00
 -1.59502731e+00 -5.22662401e-01  2.33198291e+00 -5.47145605e-01
 -1.53972930e+00  1.45429276e-01 -2.29708426e-01 -2.05565691e+00
 -1.61065027e+00 -1.42469552e+00 -2.87517115e-01 -1.97023511e-01
 -1.27771953e+00  5.65739125e-02 -7.30202816e-01  2.16371375e+00
 -1.50691003e+00 -1.15528275e+00 -1.18045390e+00 -2.53295102e-02
  6.32417172e-01 -2.37735528e+00  8.95146258e-01 -6.03295028e-01
 -1.93691450e+00 -9.36172264e-01  3.01427381e+00  1.06890815e+00
 -1.31942675e-01 -1.19868035e-01 -2.46737283e-01  3.44126076e-01
 -3.42703946e-01  1.74482211e-01  1.32662182e+00  4.04559418e+00
  3.03557156e+00  1.40743700e+00 -1.39618292e-02  3.70901644e-01
 -8.54687855e-01 -1.59320951e-01  1.02932158e+00 -4.68323682e-01
  7.80954808e-01 -8.20936352e-01  8.13281029e-01 -9.16840270e-01
 -4.64952044e-01 -1.54617330e+00 -3.77222925e-01 -1.80948744e-01
  8.20128962e-01 -3.45088924e+00 -1.59114963e+00 -2.68932020e+00
 -1.68145949e+00  4.07977611e-01  1.34790567e+00 -9.47992206e-02
  1.54268998e+00 -1.59677309e+00 -6.49481609e-01  2.35050168e+00
 -1.40591367e+00 -5.43945760e-01 -1.13485505e+00 -1.14969183e+00
  1.02162373e+00  1.58740067e+00  1.11032015e+00  3.52728063e+00
 -1.23566420e+00  1.26533225e+00 -6.36682376e-01 -8.13181758e-01
  2.78104162e+00  4.94690489e-01 -1.30501702e+00 -7.12802559e-01
  3.32903787e+00 -1.20390844e+00 -2.35618837e-01 -4.48440795e+00
 -1.91257552e+00 -1.87197629e+00  4.80822623e-01 -4.64672387e-01
  1.64412731e+00  4.06077254e+00  4.40325354e+00  1.07774422e+00
 -5.12655698e-01 -8.65945637e-01  8.40271264e-01 -3.41412866e+00
 -4.99867272e-01 -6.91178218e-01  1.86030968e+00 -1.07217279e+00
  3.17163109e+00 -2.57625908e+00 -6.53686172e+00  3.78948643e+00
  2.33433023e-02  4.50386670e+00  2.39071301e+00 -4.79748015e+00
  1.55308500e+00 -2.55765507e+00 -6.60296461e+00 -4.80899102e+00
 -6.21957477e+00 -1.47079585e+00  2.16042948e+00 -1.36476350e+00
  3.96613565e+00 -2.59618655e+00  2.32210360e+00 -5.41820094e-01
  1.33399767e+00 -1.14755282e+00  3.21068585e+00 -4.08883527e+00
 -1.86363447e+00 -3.94388474e+00  3.20821227e+00  1.40307081e+00
  8.19641262e-01 -3.38603672e+00 -2.22831190e-01 -3.92824650e+00
 -1.48885414e+00  2.09384993e+00  3.81948087e+00  2.08750119e+00
 -3.65762046e+00  8.23322460e-01 -5.26706934e-01 -4.46854739e+00
 -3.48423302e+00  5.13791330e+00  9.59837094e-01  5.19175007e+00
 -9.49122578e-01  2.97482193e-01  4.80868736e+00  4.62334736e+00
 -8.22674949e-01  1.28595321e+00  6.32141329e+00 -4.63373111e+00
  2.53350756e+00  4.73380201e+00  1.80059344e-01  4.53987479e-01
  2.22159477e+00  9.37635258e-01 -2.01101375e+00  1.60566569e+00
  4.18354884e+00 -3.40217448e+00 -2.16314679e+00 -3.88354734e+00
  2.22832598e+00  6.79544166e-01  1.80983918e+00  4.59174821e+00
  1.15714635e+01 -2.23505629e+00  4.39157748e+00 -5.43631221e+00
 -8.83365501e+00  1.38521753e+00  7.93257652e+00  4.59383732e+00
 -1.90007402e+00  6.08095878e+00 -4.39958085e+00 -3.24487975e+00
  8.04973948e+00  4.07853729e+00 -2.14321952e+00  2.12288254e+00
 -1.62044692e+00  1.17287415e+00 -1.72821850e+00  6.61434725e-01
 -1.67033479e+00  1.05900749e+01  1.40923712e+00 -3.71034936e-01
 -7.98241258e+00  6.02726701e+00 -4.34990180e+00 -3.34662411e+00
  7.77704047e+00 -6.73281066e-01  2.98816143e+00 -4.30591932e+00
  7.42114252e+00  5.28998487e+00  1.52100410e+01 -2.93376287e+00
 -6.34294893e+00  7.37053864e-01 -1.01063713e+00  2.49707443e+00
 -3.14743311e+00  3.92134992e+00 -4.92231211e+00 -3.60965999e+00
 -7.55576031e+00  9.57282065e+00  1.54959997e+00 -6.26024120e-01
 -8.86012056e+00 -4.25985927e+00  1.17879420e+01 -3.73873290e+00
 -1.05374908e+01  4.37304385e+00  1.47181139e+00 -1.14553122e+01
  9.62784166e+00  1.18250773e+00  8.19524495e+00 -1.46877369e+01
 -2.84240710e+00 -5.25622947e+00 -5.60639134e+00  1.57834619e+00
 -8.57029893e-02  2.27652445e+01 -1.71299726e+00  4.07017475e+00
  2.61078352e+00  7.91817948e+00  3.20857935e+00 -8.21022280e+00
 -9.86444704e+00 -4.13923772e+00  9.94087586e+00 -1.82374024e+00
 -1.19536734e+01 -3.07280464e+01 -1.13374414e+01  9.07592492e+00
 -1.02274664e+01  4.57934022e+01 -1.54188892e+01  1.14666146e+00
 -1.68025633e+12  1.02080765e+12 -2.45548652e+12 -3.95641690e+11
  9.35046259e+10 -1.26487175e+12 -1.73031524e+12  1.99378419e+11
  8.33895871e+11  2.17105050e+12 -6.29412997e+12  4.59217610e+12
 -1.01163852e+13  3.21471266e+11  1.04248142e+13  4.98218021e+12
  2.51482044e+12 -1.22656550e+12 -1.92292312e+11 -2.89657221e+11
  5.73875790e+12  2.94224144e+12 -8.72572367e+11  5.44486693e+11
  1.28018435e+12  1.63301696e+12  2.34082711e+12  1.30885387e+12
 -1.10211251e+12  3.26148479e+11  9.67879404e+11  1.80340473e+12
 -2.22092978e+11 -8.71174788e+11  7.81783674e+11 -6.18245421e+11
 -4.43119314e+11 -4.18471186e+11 -1.01390580e+12 -1.61212802e+12
  1.08970759e+11  7.66451214e+11 -1.91719916e+11 -1.38409982e+12
  1.32983710e+11  3.65523086e+11  4.88176456e+12 -5.02429878e+11
  5.01961966e+11  5.91912591e+11  7.68206266e+11 -8.04202204e+11
 -1.15243363e+12  3.81356195e+11  6.62941512e+11 -4.82458947e+11
  3.81130262e+11 -5.49480606e+11  9.39175426e+11  1.01456547e+11
  5.67712519e+11 -4.61588203e+11  8.56949944e+11 -5.42750425e+11
 -5.87913109e+11 -2.97510521e+11  5.98304175e+11  1.35214491e+12
 -1.15656456e+11 -9.54035080e+11  7.76742802e+11  4.75683827e+11
  6.70457208e+11 -3.68254436e+11  5.09673055e+11  9.20778805e+11
  3.88794919e+12  3.37859270e+11  3.82002689e+11  5.67637951e+11
  7.38715310e+11  2.01598598e+10  8.89480317e+11 -4.79769081e+11
 -7.43410638e+11  1.32017987e+12 -4.20189685e+11  4.33243406e+11
 -3.74956884e+11 -1.13794407e+11  8.04155506e+10  4.33560041e+11
 -7.07134401e+10 -2.57934669e+11  7.30354920e+11  4.84078245e+11
  1.96934635e+12 -1.56223892e+12  5.88199795e+10  2.20638690e+11
 -1.42193150e+11  1.64642584e+11  6.49114868e+11  1.12686224e+12
```

```
  1.01238619e+12 -6.32501653e+11 -5.63564976e+11  2.75881181e+11
  2.54778454e+10  6.49168580e+11 -6.34285829e+11  4.84079943e+11
  6.13763203e+11 -2.85237091e+11  1.06914817e+12 -3.14052948e+11
  4.68947111e+11  2.97884616e+11  3.95760315e+11  3.77570777e+11
 -8.20788045e+11  1.04222508e+12  1.63161778e+11  4.85059525e+10
  2.82303169e+11 -1.27841621e+11  4.63923449e+11 -6.06685609e+11
 -9.92305221e+10 -1.29838348e+10  1.06797038e+12  1.23034916e+11
 -2.84265529e+11  2.89368209e+11 -3.81063180e+09  2.61104090e+11
 -1.30671533e+11  3.22083272e+11 -1.32358010e+11  5.02536997e+11
 -7.01286354e+10  4.84552818e+11  8.62692010e+11 -3.16324822e+10
  3.61337559e+11  2.41445491e+11  6.00544721e+11 -6.24770220e+11
 -7.50406137e+11 -2.43275686e+11 -5.12939675e+11 -2.47046754e+11
  4.61493319e+11 -1.13878501e+11  8.93694917e+11 -3.18432879e+10
  6.37367452e+11  9.59351506e+11  9.47609257e+11  1.17859581e+12
 -5.98153270e+11 -5.17609845e+11  4.39661275e+11 -8.95032814e+11
 -6.52475477e+11 -2.44781726e+12  8.91263044e+11  3.09405734e+12
  4.09873042e+11 -3.40864925e+10 -1.29387836e+12]
```

In [102]:

```
#eveluate the regression modcel

y_pred_test= model_lr.predict(X)
```

In [103]:

```
#compare preditced value by actual
pd.DataFrame({'Actual y_test': y_test, 'Predicted y_test': y_pred_test})
```

Out[103]:

	Actual y_test	Predicted y_test
0	1	130.804612
1	2	88.524436
2	3	74.658690
3	4	82.216194
4	5	78.016202
...	...	...
4204	8410	109.012347
4205	8411	107.187696
4206	8413	110.938100
4207	8414	89.036535
4208	8416	95.319629

4209 rows × 2 columns

In [ ]:

In [105]:

```
from sklearn.metrics import mean_squared_error
import numpy as np
print('RMSE value of testion dataset')

print (np.sqrt(mean_squared_error(y_test,y_pred_test)))
```

```
RMSE value of testion dataset
4771.643063124692
```

In [145]:

```
#XGBoost

xg_reg=xgb.XGBRegressor(objective='reg:linear', colsample_bytree=0.3,learning_rate=0.1,
max_depth=5,alpha=10,n_estimator=10)
```

In [109]:

```
xg_reg.fit(X,y)
```

```
[18:59:28] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release
_1.2.0/src/objective/regression_obj.cu:174: reg:linear is now deprecated i
n favor of reg:squarederror.
[18:59:29] WARNING: C:\Users\Administrator\workspace\xgboost-win64_release
_1.2.0\src\learner.cc:516:
Parameters: { n_estimator } might not be used.

  This may not be accurate due to some parameters are only used in languag
e bindings but
  passed down to XGBoost core.  Or some parameters are not used but slip t
hrough this
  verification. Please open an issue if you find above cases.


[18:59:34] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release
_1.2.0/src/objective/regression_obj.cu:174: reg:linear is now deprecated i
n favor of reg:squarederror.
```

Out[109]:

```
XGBRegressor(alpha=10, base_score=0.5, booster='gbtree', colsample_bylevel
=1,
             colsample_bynode=1, colsample_bytree=0.3, gamma=0, gpu_id=-1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.1, max_delta_step=0, max_depth=5,
             min_child_weight=1, missing=nan, monotone_constraints='()',
             n_estimator=10, n_estimators=100, n_jobs=0, num_parallel_tree
=1,
             objective='reg:linear', random_state=0, reg_alpha=10, reg_lam
bda=1,
             scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
```

In [ ]:

```

```

In [ ]:

In [ ]:

In [ ]: