NAME: VEENA

ROLL NO: 225229145

LAB 4 - ML

In [90]:
```
#step1
import pandas as pd
```

In [51]:
```
f = pd.read_csv('Ames_House_Sales_Cropped1.csv')
print(f)
```

```
1357          0           0           1           0  ...
1358        616           0           0           0  ...
1359       1336           0           1           0  ...
1360        600           0           1           0  ...
1361        315         110           0           0  ...
1362          0           0           0           0  ...
1363        697           0           1           0  ...
1364        765           0           1           0  ...
1365          0           0           0           0  ...
1366          0           0           0           0  ...
1367        187         627           1           0  ...
1368        593           0           0           0  ...
1369       1079           0           1           0  ...
1370          0           0           0           0  ...
1371          0           0           0           0  ...
1372        547           0           1           0  ...
1373        410           0           1           0  ...
1374          0           0           0           0  ...
1375        790         163           1           0  ...
1376        275           0           0           0  ...
```

In [52]:
```
#properties
#head
f.head()
```

Out[52]:

	BldgType	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	...	OverallQual	PoolArea	Scr
0	1Fam	Y	856	854	0	3	706	0	1	0	...	7	0	
1	1Fam	Y	1262	0	0	3	978	0	0	1	...	6	0	
2	1Fam	Y	920	866	0	3	486	0	1	0	...	7	0	
3	1Fam	Y	961	756	0	3	216	0	1	0	...	7	0	
4	1Fam	Y	1145	1053	0	4	655	0	1	0	...	8	0	

5 rows × 39 columns

In [53]:
```
#shape
f.shape
```

Out[53]: (1379, 39)

In [55]:
```
#columns
f.shape[0]
```

Out[55]: 1379

In [56]:
```
#dtype
type(f)
```

Out[56]: pandas.core.frame.DataFrame

In [11]:
```
#info
f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 82 columns):
Order              2930 non-null int64
PID                2930 non-null int64
MS SubClass        2930 non-null int64
MS Zoning          2930 non-null object
Lot Frontage       2440 non-null float64
Lot Area           2930 non-null int64
Street             2930 non-null object
Alley              198 non-null object
Lot Shape          2930 non-null object
Land Contour       2930 non-null object
Utilities          2930 non-null object
Lot Config         2930 non-null object
Land Slope         2930 non-null object
Neighborhood       2930 non-null object
Condition 1        2930 non-null object
Condition 2        2930 non-null object
Bldg Type          2930 non-null object
House Style        2930 non-null object
Overall Qual       2930 non-null int64
Overall Cond       2930 non-null int64
Year Built         2930 non-null int64
Year Remod/Add     2930 non-null int64
Roof Style         2930 non-null object
Roof Matl          2930 non-null object
Exterior 1st       2930 non-null object
Exterior 2nd       2930 non-null object
Mas Vnr Type       2907 non-null object
Mas Vnr Area       2907 non-null float64
Exter Qual         2930 non-null object
Exter Cond         2930 non-null object
Foundation         2930 non-null object
Bsmt Qual          2850 non-null object
Bsmt Cond          2850 non-null object
Bsmt Exposure      2847 non-null object
BsmtFin Type 1     2850 non-null object
BsmtFin SF 1       2929 non-null float64
BsmtFin Type 2     2849 non-null object
BsmtFin SF 2       2929 non-null float64
Bsmt Unf SF        2929 non-null float64
Total Bsmt SF      2929 non-null float64
Heating            2930 non-null object
Heating QC         2930 non-null object
Central Air        2930 non-null object
Electrical         2929 non-null object
1st Flr SF         2930 non-null int64
2nd Flr SF         2930 non-null int64
Low Qual Fin SF    2930 non-null int64
Gr Liv Area        2930 non-null int64
Bsmt Full Bath     2928 non-null float64
Bsmt Half Bath     2928 non-null float64
Full Bath          2930 non-null int64
Half Bath          2930 non-null int64
Bedroom AbvGr      2930 non-null int64
Kitchen AbvGr      2930 non-null int64
Kitchen Qual       2930 non-null object
TotRms AbvGrd      2930 non-null int64
Functional         2930 non-null object
Fireplaces         2930 non-null int64
Fireplace Qu       1508 non-null object
Garage Type        2773 non-null object
Garage Yr Blt      2771 non-null float64
Garage Finish      2771 non-null object
Garage Cars        2929 non-null float64
Garage Area        2929 non-null float64
Garage Qual        2771 non-null object
Garage Cond        2771 non-null object
Paved Drive        2930 non-null object
Wood Deck SF       2930 non-null int64
Open Porch SF      2930 non-null int64
Enclosed Porch     2930 non-null int64
3Ssn Porch         2930 non-null int64
Screen Porch       2930 non-null int64
Pool Area          2930 non-null int64
Pool QC            13 non-null object
Fence              572 non-null object
Misc Feature       106 non-null object
Misc Val           2930 non-null int64
Mo Sold            2930 non-null int64
Yr Sold            2930 non-null int64
Sale Type          2930 non-null object
Sale Condition     2930 non-null object
SalePrice          2930 non-null int64
```

```
dtypes: float64(11), int64(28), object(43)
memory usage: 1.8+ MB
```

In [57]:
```
#value_counts
# find the value counts
f.sum()
```

Out[57]:
```
BldgType          1Fam1Fam1Fam1Fam1Fam1Fam1Fam1Fam2fmCon1Fam...
CentralAir        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNYYYYYYYYYYYYY...
1stFlrSF                                                1623262
2ndFlrSF                                                 487373
3SsnPorch                                                  4978
BedroomAbvGr                                               3951
BsmtFinSF1                                               628243
BsmtFinSF2                                                66333
BsmtFullBath                                                594
BsmtHalfBath                                                 81
BsmtUnfSF                                                787085
EnclosedPorch                                             29013
Fireplaces                                                  885
FullBath                                                   2179
GarageArea                                               690551
GarageCars                                                 2580
GarageYrBlt                                             2728360
GrLivArea                                               2116337
HalfBath                                                    546
KitchenAbvGr                                               1432
LotArea                                                14749525
LotFrontage                                              103580
LowQualFinSF                                               5702
MSSubClass                                                77255
MasVnrArea                                               149435
MiscVal                                                   59144
MoSold                                                     8735
OpenPorchSF                                               65194
OverallCond                                                7692
OverallQual                                                8532
PoolArea                                                   4028
ScreenPorch                                               21989
TotRmsAbvGrd                                               9036
TotalBsmtSF                                             1481661
WoodDeckSF                                               134393
YearBuilt                                               2720710
YearRemodAdd                                            2737915
YrSold                                                  2768773
SalePrice                                             255776246
dtype: object
```

In [58]:
```
#step2[predict sale price without categorical features]
#drop
f.drop(['BldgType', 'CentralAir'],axis = 1)
```

Out[58]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	OverallQual	Poo
0	856	854	0	3	706	0	1	0	150	0	...	7	
1	1262	0	0	3	978	0	0	1	284	0	...	6	
2	920	866	0	3	486	0	1	0	434	0	...	7	
3	961	756	0	3	216	0	1	0	540	272	...	7	
4	1145	1053	0	4	655	0	1	0	490	0	...	8	
5	796	566	320	1	732	0	1	0	64	0	...	5	
6	1694	0	0	3	1369	0	1	0	317	0	...	8	
7	1107	983	0	3	859	32	1	0	216	228	...	7	
8	1022	752	0	2	0	0	0	0	952	205	...	7	
9	1077	0	0	2	851	0	1	0	140	0	...	5	
10	1040	0	0	3	906	0	1	0	134	0	...	5	
11	1182	1142	0	4	998	0	1	0	177	0	...	9	
12	912	0	0	2	737	0	1	0	175	0	...	5	
13	1494	0	0	3	0	0	0	0	1494	0	...	7	
14	1253	0	0	2	733	0	1	0	520	176	...	6	
15	854	0	0	2	0	0	0	0	832	0	...	7	
16	1004	0	0	2	578	0	1	0	426	0	...	6	
17	1296	0	0	2	0	0	0	0	0	0	...	4	
18	1114	0	0	3	646	0	1	0	468	0	...	5	
19	1339	0	0	3	504	0	0	0	525	0	...	5	
20	1158	1218	0	4	0	0	0	0	1158	0	...	8	
21	1108	0	0	3	0	0	0	0	637	205	...	7	
22	1795	0	0	3	0	0	0	0	1777	0	...	8	
23	1060	0	0	3	840	0	1	0	200	0	...	5	
24	1060	0	0	3	188	668	1	0	204	0	...	5	
25	1600	0	0	3	0	0	0	0	1566	0	...	8	
26	900	0	0	3	234	486	0	1	180	0	...	5	
27	1704	0	0	3	1218	0	1	0	486	0	...	8	
28	1600	0	0	2	1277	0	1	0	207	0	...	5	
29	520	0	0	1	0	0	0	0	520	87	...	4	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1349	1048	510	0	3	580	0	1	0	333	0	...	5	
1350	804	0	0	2	510	0	1	0	278	154	...	5	
1351	1440	0	0	3	678	0	0	0	762	99	...	6	
1352	734	1104	0	4	0	0	0	0	732	0	...	5	
1353	958	0	0	2	958	0	0	0	0	0	...	6	
1354	968	0	0	4	0	0	0	0	656	0	...	4	
1355	962	830	0	3	0	0	1	0	936	0	...	6	
1356	1126	0	0	3	936	0	1	0	190	0	...	5	
1357	1537	0	0	3	0	0	1	0	1319	0	...	6	
1358	864	0	0	3	616	0	0	0	248	0	...	4	
1359	1932	0	304	2	1336	0	1	0	596	0	...	8	
1360	1236	0	0	2	600	0	1	0	312	158	...	6	
1361	1040	685	0	3	315	110	0	0	114	216	...	7	
1362	1423	748	0	3	0	0	0	0	588	0	...	6	
1363	848	0	0	1	697	0	1	0	151	0	...	6	
1364	1026	981	0	3	765	0	1	0	252	0	...	10	
1365	952	0	0	2	0	0	0	0	952	0	...	6	
1366	1422	0	0	3	0	0	0	0	1422	0	...	7	
1367	913	0	0	3	187	627	1	0	0	252	...	6	
1368	1188	0	0	3	593	0	0	0	595	0	...	5	
1369	1220	870	0	3	1079	0	1	0	141	0	...	8	
1370	796	550	0	2	0	0	0	0	560	0	...	4	
1371	1578	0	0	3	0	0	0	0	1573	0	...	8	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	OverallQual	Poo
1372	1072	0	0	2	547	0	1	0	0	0	...	5	
1373	1221	0	0	2	410	0	1	0	811	0	...	7	
1374	953	694	0	3	0	0	0	0	953	0	...	6	
1375	2073	0	0	3	790	163	1	0	589	0	...	6	
1376	1188	1152	0	4	275	0	0	0	877	0	...	7	
1377	1078	0	0	2	49	1029	1	0	0	112	...	5	
1378	1256	0	0	3	830	290	1	0	136	0	...	5	

1379 rows × 37 columns

In [59]:
```
#pop
df2 = pd.concat([f.pop(x) for x in ['BldgType','CentralAir']], axis=1)
print(df2)
```

```
     BldgType CentralAir
0        1Fam          Y
1        1Fam          Y
2        1Fam          Y
3        1Fam          Y
4        1Fam          Y
5        1Fam          Y
6        1Fam          Y
7        1Fam          Y
8        1Fam          Y
9      2fmCon          Y
10       1Fam          Y
11       1Fam          Y
12       1Fam          Y
13       1Fam          Y
14       1Fam          Y
15       1Fam          Y
16       1Fam          Y
17     Duplex          Y
18       1Fam          Y
19       1Fam          Y
20       1Fam          Y
21       1Fam          Y
22       1Fam          Y
23     TwnhsE          Y
24       1Fam          Y
25       1Fam          Y
26       1Fam          Y
27       1Fam          Y
28       1Fam          Y
29       1Fam          N
...       ...        ...
1349     1Fam          Y
1350     1Fam          Y
1351     1Fam          Y
1352     1Fam          Y
1353   TwnhsE          Y
1354     1Fam          Y
1355     1Fam          Y
1356     1Fam          Y
1357     1Fam          Y
1358     1Fam          Y
1359     1Fam          Y
1360     1Fam          Y
1361     1Fam          Y
1362     1Fam          Y
1363   TwnhsE          Y
1364     1Fam          Y
1365     1Fam          N
1366     1Fam          Y
1367     1Fam          Y
1368     1Fam          Y
1369     1Fam          Y
1370     1Fam          N
1371     1Fam          Y
1372   TwnhsE          Y
1373     1Fam          Y
1374     1Fam          Y
1375     1Fam          Y
1376     1Fam          Y
1377     1Fam          Y
1378     1Fam          Y

[1379 rows x 2 columns]
```

In [61]:
```
#prepare X matrix (36 feature columns)and y vector
X=f.drop(["SalePrice"],axis=1)
print(X)
```

```
      1stFlrSF  2ndFlrSF  3SsnPorch  BedroomAbvGr  BsmtFinSF1  BsmtFinSF2  \
0          856       854          0             3         706           0
1         1262         0          0             3         978           0
2          920       866          0             3         486           0
3          961       756          0             3         216           0
4         1145      1053          0             4         655           0
5          796       566        320             1         732           0
6         1694         0          0             3        1369           0
7         1107       983          0             3         859          32
8         1022       752          0             2           0           0
9         1077         0          0             2         851           0
10        1040         0          0             3         906           0
11        1182      1142          0             4         998           0
12         912         0          0             2         737           0
13        1494         0          0             3           0           0
14        1253         0          0             2         733           0
15         854         0          0             2           0           0
16        1004         0          0             2         578           0
17        1296         0          0             2           0           0
```

In [62]:
```
y=f["SalePrice"].values
y
```

Out[62]:
```
array([208500, 181500, 223500, ..., 266500, 142125, 147500], dtype=int64)
```

In [63]:
```
#split dataset for training and testing as x_train,x_test,y_train,y_test:

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=.25,random_state=42)
```

In [65]:
```
#create LinearRegression Model:

from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train,y_train)
```

Out[65]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [66]:
```
y_pred=reg.predict(X_test)
y_pred
```

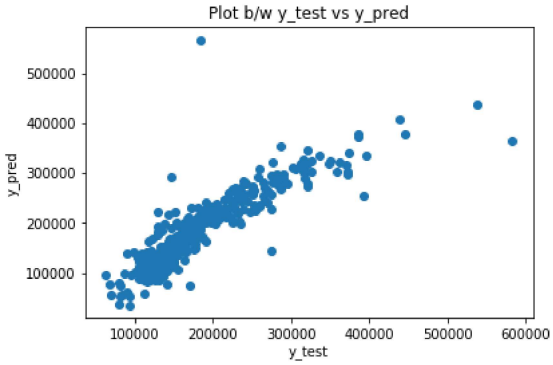
Out[66]:
```
array([257434.93050731, 111083.73474777, 100018.05832306, 204028.53821287,
       207319.25418308,  38036.30928908, 234153.38582861, 205076.12689755,
       187014.12655265, 235636.78799033, 100976.50943785, 304119.53647053,
       101769.64600697, 288758.46001583, 204767.89337818, 145002.47279615,
       248322.97436857, 151533.68038627, 209697.39458735, 278312.23574247,
        93329.85601416, 153784.1486835 , 163237.87451148, 241495.91212954,
       372429.94726955, 221672.07367176, 119658.75685685, 100039.7813706 ,
       319435.028922  , 172088.64665035, 258772.64470132, 199597.36129748,
       156349.69283189, 142311.22500082, 204807.33161343, 379608.17785511,
       124176.23377579, 141127.25006086, 273832.43247537, 212547.04205568,
       169474.57176061, 123305.65719342, 200341.24016626, 377957.32307172,
       146941.71712444, 283693.48760685, 106694.19786713, 221733.02172149,
        87733.47013999, 295005.6654633 , 125250.62019285,  53569.11256064,
       133831.05086051, 170901.42929767, 213273.17757313, 110818.14723236,
       103893.81850248, 200665.21921066, 137764.84464954, 311840.64900745,
        53035.52267558, 184866.47797863, 154675.79963863, 298477.53600015,
        78162.54074696, 323728.93535842, 169417.44138475, 112718.07937644,
       200547.87648943, 205739.30910725, 130292.88475688, 271581.37221154,
       119489.76459211, 126635.29232479,  77972.26780369, 138563.37305239,
        35555.51251473, 163787.98080926, 271929.13690193, 115260.62742948,
       319958.55004418, 218515.16541539, 291730.97232825, 211176.53228446,
        93761.81278803, 263031.10150395, 141886.3966697 , 118491.50416138,
       143580.34191078, 278395.85370536, 265558.84178051, 300839.51324272,
       182673.34437968, 168604.54926015, 148322.24627478, 213781.03244098,
       247750.58741002, 221693.90686826, 274474.96290588, 242912.73279155,
       111041.26737668, 100160.66573763, 193531.44933253, 206050.88712664,
       154584.47385998, 217440.52365194, 210904.23539294, 125864.27688686,
       171322.66919875, 224459.70210801, 155548.12334484, 111942.78717778,
       316624.83474091, 298116.31992271, 335573.77762729,  86738.8850683 ,
       146692.58771876, 228471.5403277 , 123374.0735452 , 204867.33148085,
       217343.21720894, 120569.43114743, 134354.12384283, 223860.01643792,
       138360.76267159, 210765.75439348, 145423.97229827, 291155.26974376,
       183089.17904067,  98716.55524055, 323896.4670688 , 170447.3065641 ,
       123643.15346339, 134345.2479979 , 309423.49789847, 136836.97808909,
       282083.02092595, 132621.2237255 ,  90821.15043377, 164193.14328582,
       147333.6002683 , 162030.92949986, 177621.20328208, 254054.45691385,
       119092.92298319, 140081.84030908, 236509.39317261, 318578.43961647,
       105538.81578077, 192023.58779349, 174531.98294673, 146314.75163405,
        95146.46236331, 250936.3282582 , 304056.57810783,  55008.42319177,
       281810.05395735,  93402.68236871, 138283.4460376 , 166415.31922825,
       209758.26998542, 205602.16975924, 177026.58819413, 248435.62725141,
       145190.55457334, 133283.94785869, 171593.43395049, 344925.23412434,
       181194.80288371,  98431.28117697, 131287.46561749, 110849.96802314,
       131283.81966809, 194221.20494183, 156966.39268734, 268340.5847198 ,
       211804.83660293, 151971.72881399, 115946.37551519, 243618.89320988,
       136292.76068264, 265335.41730238, 302175.59064225,  85054.45272429,
       146007.43532849, 150439.28821428, 159145.83079314, 232246.23120778,
       231571.1353202 , 139178.99313667, 355030.61932852, 127659.29864589,
       235339.23724218, 116707.21916853, 103103.7519102 , 159574.23418427,
       149485.12468629, 408898.40096836, 335368.65026124, 290500.91914893,
       281845.43486745, 275341.81718381, 321705.52591353, 307549.31796903,
       201661.1953694 , 142047.81494913, 157876.45367587, 258695.10810497,
       207592.92972053, 146121.6018252 , 107032.8224219 , 151245.32235257,
       100804.20363788, 296980.23153218, 341413.30991684, 255158.86137004,
       147006.34954483, 199721.99255106, 127291.4109391 , 264030.58756798,
       121760.19396662, 134138.79148651, 307661.79229143, 170331.68886934,
       173593.89056583, 567240.20119394, 182277.72265649, 144159.48526119,
       199515.17896284, 107691.20503758, 181383.00130468, 328401.72330922,
       139225.20214788, 197660.10401716, 116340.37479536,  54949.57465389,
       199375.67314476, 272915.38777785, 119902.62390801, 197573.81603521,
       231881.98496949, 119555.98822736, 128709.54702323, 289158.80585509,
       200733.61408617, 364228.3936614 , 119308.94568399, 124086.72254054,
       225161.62739582, 171005.22659863,  93631.91767257, 178506.48207476,
       215661.64584831, 226225.30038966, 132391.85608637,  73443.41948242,
       242913.00968392, 141042.39030875, 114633.38186086,  89093.07025277,
       208503.15479623, 172828.98633425, 270797.22136354, 253693.44924262,
       103518.85057136, 220521.27460705, 185034.32129264, 169518.90296965,
       154846.1745908 , 224605.38839973,  85972.23828416, 151323.12661404,
       180628.9255433 , 200106.98029798, 179499.81727916, 223755.69902659,
        73639.52734249,  86011.81491741, 110000.03909782, 231518.89807858,
       153369.3597679 , 320899.35432207, 194412.83170294,  57741.65743014,
       203877.63015243, 109803.05085773, 194633.36256902,  99160.35922058,
       230240.81594426, 211484.77477153, 198342.97693786, 109749.28794237,
        95564.23930754, 235396.44527297, 184775.90385464, 188806.96222909,
       265328.01705295, 245862.13668813, 107422.0040257 , 164786.63106788,
       268303.44421412, 116075.01592387, 222914.33666749, 102600.43292169,
       226400.498392  , 156093.55758686, 112003.82907434, 218089.37220929,
        81085.92660207,  85015.51341447,  98208.34263385, 304262.27895876,
       143757.7904366 , 195367.3512203 , 241835.21911118,  60944.84144952,
       143731.15831775, 110459.58351765, 438412.24654647, 123265.07239607,
       116119.87930804, 277961.71617766, 201687.8173972 , 237947.02580858,
       206117.84986919,  99453.60459334, 115124.57034995, 209795.31353843,
       167537.15563199, 162970.26517767, 151279.43057806, 189724.86274122,
        96455.47006125, 168652.82586796,  83051.3159692 , 171340.42959845,
       176868.33711507, 175644.689618  , 131414.67048292, 204543.74960397,
```

```
       233147.40044759, 125478.90203709, 175060.5016749 , 258877.30470759,
       229115.65129681])
```

In [67]:
```
import sklearn.metrics as metrics
mse=metrics.mean_squared_error(y_test,y_pred)
print("MSE without Categorical: ",mse)
```

```
MSE without Categorical:  1474827325.5952084
```

In [68]:
```
#step 3 [ Create Scatter Plot]
import matplotlib.pyplot as plt
%matplotlib inline
plt.scatter(y_test,y_pred)
plt.xlabel("y_test")
plt.ylabel("y_pred")
plt.title("Plot b/w y_test vs y_pred")
plt.show()
```



In [69]:
```
#step 4:[Encode Categorical columns]
encoding=pd.get_dummies(f)
print(encoding)
```

```
1352       734      1104          0           4           0           0
1353       958         0          0           2         958           0
1354       968         0          0           4           0           0
1355       962       830          0           3           0           0
1356      1126         0          0           3         936           0
1357      1537         0          0           3           0           0
1358       864         0          0           3         616           0
1359      1932         0        304           2        1336           0
1360      1236         0          0           2         600           0
1361      1040       685          0           3         315         110
1362      1423       748          0           3           0           0
1363       848         0          0           1         697           0
1364      1026       981          0           3         765           0
1365       952         0          0           2           0           0
1366      1422         0          0           3           0           0
1367       913         0          0           3         187         627
1368      1188         0          0           3         593           0
1369      1220       870          0           3        1079           0
1370       796       550          0           2           0           0
1371      1578         0          0           3           0           0
```

In [70]:
```
y=encoding["SalePrice"].values
y
```

Out[70]:
```
array([208500, 181500, 223500, ..., 266500, 142125, 147500], dtype=int64)
```

In [71]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=.25,random_state=42)
```

In [72]:
```
from sklearn.linear_model import LinearRegression
import sklearn.metrics as metrics
reg=LinearRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
print(y)
mse_cd=metrics.mean_squared_error(y_test,y_pred)
print("MSE with Categorical data: ",mse_cd)
```

```
[208500 181500 223500 ... 266500 142125 147500]
MSE with Categorical data:  1474827325.5952084
```

In [73]:
```
#step6:[Normalize using StandardScaler and Predict Sale Price]
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
ss_X_train=ss.fit_transform(X_train)
ss_X_train
```

Out[73]:
```
array([[ 0.39851037, -0.79290427, -0.11340519, ...,  0.84304574,
         0.65341548,  0.11447318],
       [ 1.57467708, -0.79290427, -0.11340519, ...,  1.14796951,
         1.04559751,  0.8683921 ],
       [ 0.37564751,  0.70143387, -0.11340519, ..., -1.52858358,
        -1.74869949,  0.8683921 ],
       ...,
       [ 1.22157303, -0.79290427, -0.11340519, ..., -0.61381227,
         0.50634721,  0.11447318],
       [-1.11297817,  0.90510323, -0.11340519, ...,  1.08020867,
         0.947552  ,  0.8683921 ],
       [ 0.11145456, -0.79290427, -0.11340519, ...,  0.87692616,
         0.65341548,  0.8683921 ]])
```

In [74]:
```
ss_X_test=ss.transform(X_test)
ss_X_test
```

Out[74]:
```
array([[ 0.85830772, -0.79290427, -0.11340519, ...,  1.01244784,
         0.89852925,  0.11447318],
       [-0.64810018, -0.79290427, -0.11340519, ..., -0.27500808,
        -1.01335818,  0.8683921 ],
       [-0.21624632,  0.76093278, -0.11340519, ..., -2.47723531,
        -1.74869949, -1.39336465],
       ...,
       [-0.04350477,  0.37647826, -0.11340519, ..., -1.86738777,
        -1.74869949,  0.11447318],
       [-0.64301955,  1.46805451, -0.11340519, ...,  0.63976323,
         0.31025619, -1.39336465],
       [-0.29499614,  0.81585486, -0.11340519, ...,  0.47036113,
         0.06514242, -1.39336465]])
```

In [77]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(ss_X_train,y_train)
ss_y_pred=lr.predict(ss_X_test)
ss_y_pred
```

Out[77]:
```
array([257434.93050744, 111083.73474785, 100018.05832285, 204028.53821287,
       207319.25418308,  38036.30928916, 234153.38582861, 205076.12689752,
       187014.12655254, 235636.7879903 , 100976.50943791, 304119.53647048,
       101769.64600699, 288758.4600158 , 204767.89337824, 145002.47279621,
       248322.97436854, 151533.68038629, 209697.39458739, 278312.23574242,
        93329.85601403, 153784.14868355, 163237.87451149, 241495.91212963,
       372429.94726962, 221672.07367176, 119658.75685683, 100039.78137061,
       319435.02892196, 172088.64665033, 258772.64470129, 199597.36129749,
       156349.69283166, 142311.22500091, 204807.33161344, 379608.17785506,
       124176.23377582, 141127.2500608 , 273832.43247534, 212547.04205579,
       169474.57176061, 123305.65719341, 200341.2401663 , 377957.32307188,
       146941.71712444, 283693.48760682, 106694.19786716, 221733.02172145,
        87733.47014007, 295005.66546328, 125250.6201929 ,  53569.11256069,
       133831.05086042, 170901.42929766, 213273.17757306, 110818.14723246,
       103893.81850247, 200665.21921075, 137764.8446496 , 311840.64900749,
        53035.52267542, 184866.47797863, 154675.79963857, 298477.53600009,
        78162.54074701, 323728.93535835, 169417.44138457, 112718.07937648,
       200547.87648951, 205739.30910726, 130292.8847569 , 271581.37221162,
       119489.76459216, 126635.29232465,  77972.26780375, 138563.37305251,
        35555.51251461, 163787.98080927, 271929.13690199, 115260.62742945,
       319958.55004413, 218515.16541528, 291730.97232829, 211176.53228434,
        93761.812788  , 263031.10150393, 141886.39666972, 118491.50416142,
       143580.34191084, 278395.85370537, 265558.84178047, 300839.51324266,
       182673.34437967, 168604.54925995, 148322.24627476, 213781.03244089,
       247750.58740998, 221693.90686818, 274474.96290598, 242912.73279156,
       111041.26737669, 100160.66573763, 193531.44933257, 206050.8871266 ,
       154584.47386006, 217440.52365197, 210904.23539298, 125864.27688693,
       171322.66919883, 224459.70210798, 155548.12334489, 111942.78717778,
       316624.83474096, 298116.31992267, 335573.77762724,  86738.88506836,
       146692.58771881, 228471.54032766, 123374.07354518, 204867.33148091,
       217343.217209  , 120569.43114725, 134354.12384288, 223860.01643784,
       138360.7626716 , 210765.75439347, 145423.97229829, 291155.26974384,
       183089.17904065,  98716.55524057, 323896.46706873, 170447.30656403,
       123643.1534634 , 134345.24799791, 309423.49789842, 136836.97808912,
       282083.02092594, 132621.2237255 ,  90821.15043383, 164193.14328591,
       147333.60026833, 162030.92949988, 177621.20328213, 254054.45691383,
       119092.92298318, 140081.84030908, 236509.39317262, 318578.43961648,
       105538.81578073, 192023.58779343, 174531.98294686, 146314.75163406,
        95146.46236334, 250936.32825823, 304056.57810778,  55008.42319182,
       281810.05395732,  93402.68236882, 138283.44603769, 166415.31922829,
       209758.26998546, 205602.16975917, 177026.5881941 , 248435.62725139,
       145190.55457331, 133283.9478586 , 171593.43395047, 344925.23412415,
       181194.80288374,  98431.28117698, 131287.46561753, 110849.96802318,
       131283.81966818, 194221.20494186, 156966.39268725, 268340.58472   ,
       211804.83660295, 151971.72881405, 115946.37551512, 243618.89320985,
       136292.76068289, 265335.41730235, 302175.59064225,  85054.45272437,
       146007.43532851, 150439.28821433, 159145.83079309, 232246.23120773,
       231571.1353202 , 139178.99313672, 355030.61932861, 127659.2986459 ,
       235339.23724214, 116707.2191686 , 103103.75191025, 159574.23418421,
       149485.12468629, 408898.40096834, 335368.65026124, 290500.91914891,
       281845.43486751, 275341.81718383, 321705.52591348, 307549.31796903,
       201661.19536935, 142047.81494919, 157876.45367591, 258695.10810474,
       207592.92972052, 146121.60182518, 107032.8224218 , 151245.32235255,
       100804.20363792, 296980.23153226, 341413.30991685, 255158.86137003,
       147006.3495449 , 199721.99255085, 127291.41093915, 264030.58756798,
       121760.19396667, 134138.7914865 , 307661.79229133, 170331.6888693 ,
       173593.89056582, 567240.20119402, 182277.72265651, 144159.48526112,
       199515.17896271, 107691.20503763, 181383.00130478, 328401.72330919,
       139225.20214793, 197660.10401723, 116340.37479543,  54949.57465397,
       199375.67314479, 272915.38777785, 119902.62390806, 197573.81603521,
       231881.98496948, 119555.98822744, 128709.5470232 , 289158.80585502,
       200733.61408608, 364228.39366135, 119308.94568406, 124086.72254061,
       225161.62739577, 171005.22659863,  93631.91767256, 178506.48207471,
       215661.64584826, 226225.30038961, 132391.85608634,  73443.41948249,
       242913.00968389, 141042.3903088 , 114633.38186092,  89093.07025268,
       208503.15479627, 172828.98633426, 270797.22136365, 253693.44924264,
       103518.85057142, 220521.2746072 , 185034.32129258, 169518.90296967,
       154846.17459075, 224605.38839972,  85972.23828417, 151323.12661412,
       180628.92554336, 200106.98029802, 179499.81727913, 223755.69902665,
        73639.52734244,  86011.8149172 , 110000.03909792, 231518.8980785 ,
       153369.35976795, 320899.35432208, 194412.83170292,  57741.65743015,
       203877.63015237, 109803.05085771, 194633.36256902,  99160.35922069,
       230240.8159443 , 211484.7747715 , 198342.97693784, 109749.28794236,
        95564.23930747, 235396.44527286, 184775.90385458, 188806.96222905,
       265328.01705292, 245862.13668812, 107422.0040257 , 164786.63106792,
       268303.44421403, 116075.01592387, 222914.33666749, 102600.43292165,
       226400.49839201, 156093.55758685, 112003.8290744 , 218089.37220934,
        81085.92660214,  85015.51341453,  98208.34263391, 304262.27895878,
       143757.79043647, 195367.35122033, 241835.21911105,  60944.84144938,
       143731.15831782, 110459.58351767, 438412.24654632, 123265.07239614,
       116119.87930806, 277961.71617761, 201687.81739722, 237947.02580858,
       206117.84986913,  99453.60459339, 115124.57034996, 209795.31353844,
       167537.15563177, 162970.26517777, 151279.43057802, 189724.86274136,
        96455.47006128, 168652.82586799,  83051.31596924, 171340.42959818,
       176868.33711508, 175644.68961804, 131414.67048289, 204543.74960401,
```

```
       233147.40044758, 125478.90203709, 175060.50167483, 258877.30470755,
       229115.65129686])
```

In [78]:
```
ss_mse=metrics.mean_squared_error(y_test,ss_y_pred)
print("SS_MSE: ",ss_mse)
```

```
SS_MSE:  1474827325.5955286
```

In [79]:
```
#step 7:[Normalize using MinMaxScaler and Predict Sale Price]
#Repeat with MinMaxScaler
from sklearn.preprocessing import MinMaxScaler
mm=MinMaxScaler()
mm_X_train=mm.fit_transform(X_train)
mm_X_test=mm.transform(X_test)
mm_lr=LinearRegression()
mm_lr.fit(mm_X_train,y_train)
mm_y_pred=mm_lr.predict(mm_X_test)
print("Predictions of scaled data using MinMaxScaler:",mm_y_pred)
```

```
Predictions of scaled data using MinMaxScaler: [257434.93050744 111083.73474785 100018.05832285 204028.53821287
 207319.25418308  38036.30928916 234153.38582861 205076.12689752
 187014.12655254 235636.7879903  100976.50943791 304119.53647048
 101769.64600699 288758.4600158  204767.89337824 145002.47279621
 248322.97436854 151533.68038629 209697.39458739 278312.23574242
  93329.85601403 153784.14868355 163237.87451149 241495.91212963
 372429.94726962 221672.07367176 119658.75685683 100039.78137061
 319435.02892196 172088.64665033 258772.64470129 199597.36129749
 156349.69283166 142311.22500091 204807.33161344 379608.17785506
 124176.23377582 141127.2500608  273832.43247534 212547.04205579
 169474.57176061 123305.65719341 200341.2401663  377957.32307188
 146941.71712444 283693.48760682 106694.19786716 221733.02172145
  87733.47014007 295005.66546328 125250.6201929   53569.11256069
 133831.05086042 170901.42929766 213273.17757306 110818.14723246
 103893.81850247 200665.21921075 137764.8446496  311840.64900749
  53035.52267542 184866.47797863 154675.79963857 298477.53600009
  78162.54074701 323728.93535835 169417.44138457 112718.07937648
 200547.87648951 205739.30910726 130292.8847569  271581.37221162
 119489.76459216 126635.29232465  77972.26780375 138563.37305251
  35555.51251461 163787.98080927 271929.13690199 115260.62742945
 319958.55004413 218515.16541528 291730.97232829 211176.53228434
  93761.812788   263031.10150393 141886.39666972 118491.50416142
 143580.34191084 278395.85370537 265558.84178047 300839.51324266
 182673.34437967 168604.54925995 148322.24627476 213781.03244089
 247750.58740998 221693.90686818 274474.96290598 242912.73279156
 111041.26737669 100160.66573763 193531.44933257 206050.8871266
 154584.47386006 217440.52365197 210904.23539298 125864.27688693
 171322.66919883 224459.70210798 155548.12334489 111942.78717778
 316624.83474095 298116.31992267 335573.77762724  86738.88506836
 146692.58771881 228471.54032766 123374.07354518 204867.33148091
 217343.217209   120569.43114725 134354.12384288 223860.01643784
 138360.7626716  210765.75439347 145423.97229829 291155.26974384
 183089.17904065  98716.55524057 323896.46706873 170447.30656403
 123643.1534634  134345.24799791 309423.49789842 136836.97808912
 282083.02092594 132621.2237255   90821.15043383 164193.14328591
 147333.60026833 162030.92949988 177621.20328213 254054.45691383
 119092.92298318 140081.84030908 236509.39317262 318578.43961648
 105538.81578073 192023.58779343 174531.98294686 146314.75163406
  95146.46236334 250936.32825823 304056.57810778  55008.42319182
 281810.05395732  93402.68236882 138283.44603769 166415.31922829
 209758.26998546 205602.16975917 177026.5881941  248435.62725139
 145190.55457331 133283.9478586  171593.43395047 344925.23412415
 181194.80288374  98431.28117698 131287.46561753 110849.96802318
 131283.81966818 194221.20494186 156966.39268725 268340.58472
 211804.83660295 151971.72881405 115946.37551512 243618.89320985
 136292.76068289 265335.41730235 302175.59064225  85054.45272437
 146007.43532851 150439.28821433 159145.83079309 232246.23120773
 231571.1353202  139178.99313672 355030.61932861 127659.2986459
 235339.23724214 116707.2191686  103103.75191025 159574.23418421
 149485.12468629 408898.40096834 335368.65026124 290500.91914891
 281845.43486751 275341.81718383 321705.52591348 307549.31796903
 201661.19536935 142047.81494919 157876.45367591 258695.10810474
 207592.92972052 146121.60182518 107032.8224218  151245.32235255
 100804.20363792 296980.23153226 341413.30991685 255158.86137003
 147006.3495449  199721.99255085 127291.41093915 264030.58756798
 121760.19396667 134138.7914865  307661.79229133 170331.6888693
 173593.89056582 567240.20119402 182277.72265651 144159.48526112
 199515.17896271 107691.20503763 181383.00130478 328401.72330919
 139225.20214793 197660.10401723 116340.37479543  54949.57465397
 199375.67314479 272915.38777785 119902.62390806 197573.81603521
 231881.98496948 119555.98822744 128709.5470232  289158.80585502
 200733.61408608 364228.39366135 119308.94568406 124086.72254061
 225161.62739577 171005.22659863  93631.91767256 178506.48207471
 215661.64584826 226225.30038961 132391.85608634  73443.41948249
 242913.00968389 141042.3903088  114633.38186092  89093.07025268
 208503.15479627 172828.98633426 270797.22136365 253693.44924264
 103518.85057142 220521.2746072  185034.32129258 169518.90296967
 154846.17459075 224605.38839972  85972.23828417 151323.12661412
 180628.92554336 200106.98029802 179499.81727913 223755.69902665
  73639.52734244  86011.8149172  110000.03909793 231518.8980785
 153369.35976795 320899.35432208 194412.83170292  57741.65743015
 203877.63015237 109803.05085771 194633.36256902  99160.35922069
 230240.8159443  211484.7747715  198342.97693784 109749.28794236
  95564.23930747 235396.44527286 184775.90385458 188806.96222905
 265328.01705292 245862.13668812 107422.0040257  164786.63106792
 268303.44421403 116075.01592387 222914.33666749 102600.43292165
 226400.49839201 156093.55758685 112003.8290744  218089.37220934
  81085.92660214  85015.51341453  98208.34263391 304262.27895878
 143757.79043647 195367.35122033 241835.21911105  60944.84144938
 143731.15831782 110459.58351767 438412.24654632 123265.07239614
 116119.87930806 277961.71617761 201687.81739722 237947.02580858
 206117.84986913  99453.60459339 115124.57034996 209795.31353844
 167537.15563177 162970.26517777 151279.43057802 189724.86274136
  96455.47006128 168652.82586799  83051.31596924 171340.42959818
 176868.33711508 175644.68961804 131414.67048289 204543.74960401
```

```
233147.40044758 125478.90203709 175060.50167483 258877.30470755
229115.65129686]
```

In [80]:
```
mm_mse=metrics.mean_squared_error(y_test,mm_y_pred)
print("MM_MSE: ",mm_mse)
```

```
MM_MSE:  1474827325.595527
```

In [81]:
```
#Step-8: [Predict using SGD Regressor]
from sklearn.linear_model import SGDRegressor
sgd=SGDRegressor()
sgd.fit(ss_X_train, y_train)
sgd_y_pred=sgd.predict(ss_X_test)
print("Predictions of scaled data using SGDRegressor:", sgd_y_pred)
```

```
Predictions of scaled data using SGDRegressor: [252414.86358195 103710.02736608 112342.57780954 202429.49638962
 209081.61091982  30704.83630667 235594.78850454 202315.98385543
 187759.0930742  229299.86997856  92641.87627655 301936.2669506
 104114.76618952 294814.33241353 211395.62718237 145235.86855375
 250412.74222141 144832.82232162 211780.63859769 278776.36682153
  94541.07602716 150314.27853466 157739.0652652  241795.02416722
 385447.42315857 221106.84906308 116755.71154634  94578.11958347
 326539.39339051 166134.42567998 254563.36005127 199930.50071645
 154776.43983968 137656.32135944 207745.8886153  378611.97936181
 115155.33210421 110743.20278961 271775.53086107 211515.60480356
 172614.87843389 118184.51139169 185588.14871336 364212.48781365
 148437.80517137 294964.51190469  99867.424484   219442.9418907
  81655.05581181 301350.09996908 119071.99952478  45855.05139632
 119036.7688937  157310.21958457 216870.44874318 110096.10527317
 101880.21411344 200500.97117288 138355.1763748  314247.80531574
  54008.87100446 183866.33950838 157947.97851659 300847.71068129
  79739.31881974 329952.97086082 168260.50143118 115481.56973185
 196498.99249914 210093.41754262 133435.04613045 280693.39494912
 117367.04434427 131183.46036866  70956.62017518 123475.96384488
  34730.72303563 165991.22124499 273675.25566654 107636.41500855
 327221.31344841 221460.1124827  294504.55566713 215716.47769799
  96063.04839216 267634.92356231 134493.10084958 109926.68623052
 134493.22071736 288986.63780909 272822.45856076 307441.73632755
 185361.46379161 169853.33285696 156730.61109138 218533.11412077
 249895.05702545 224591.94656065 279385.98573644 244985.90605383
  99744.07841429  92359.03271049 196657.40435541 200779.82970506
 150711.7341519  218239.64862754 212319.68088539 117927.72099936
 171459.77651365 228165.09163013 151967.1419365  104727.51223934
 321513.14276558 300914.5007683  341671.92200363  80879.16443073
 141970.53120189 227787.99674771 121457.1868824  205618.1768498
 219106.88768376 132610.65815331 126263.97221505 227424.48918074
 133045.19862963 213787.61928009 139683.37355783 297108.76444382
 186239.23385331  89713.19769976 329195.76262349 171107.68625437
 117889.74324646 130766.0410598  306662.14865906 142848.69070403
 286984.68608392 129135.06652864  81839.12920982 164321.63241767
 141585.25284486 155479.78687309 173644.67577747 254775.89030564
 113898.52709474 132874.1296687  232420.86499656 319365.02181522
 108583.71729427 196478.2934673  162753.10964379 145940.14423964
  88934.38010285 257313.43291337 311001.49590583  48958.14350044
 284181.55347511  89942.90535598 139884.79566452 168251.40706028
 211553.13906525 208823.9129607  181119.49134202 254388.12408347
 154951.6126285  127820.37914982 173718.61606619 346286.42419227
 170647.60451137  93131.35140715 123947.69546946 109379.25345868
 127904.15939339 190807.35741847 150255.18057763 260905.59990943
 212818.70742507 150194.52266264 110659.49011226 246998.91992912
 109674.47677681 273004.3975465  307006.39376392  82909.21090599
 136855.78148874 150023.85185306 162732.61499104 233088.33892305
 234524.87205977 132118.2471154  337369.86820794 118676.44572177
 236886.69457846 111168.66894817  94480.63284864 161989.68418474
 145391.59159735 420011.4951731  342056.69065815 300647.48733061
 280132.93707999 280965.59735313 327372.93499803 311663.68972081
 205762.49650496 137598.56946528 155771.5787019  202611.47518235
 214118.33371576 144871.02547614 108695.74877087 149749.70507176
  99410.27025703 309689.12785692 345041.43034091 263370.52749969
 143423.34763769 214925.14951953 124439.29471141 273940.24592854
 122669.10906945 133758.10559721 315484.60519153 162804.26732409
 174325.87130815 560143.17020052 181082.68167788 141322.87185572
 201353.21752514 103190.42594356 180068.1150621  330613.74257168
 134269.24608847 199667.80255761 115455.79187186  45736.99428612
 203795.24579142 279875.00981417 116303.07378196 200347.80641372
 232192.08442507 119170.58499507 124391.80937647 290829.8563093
 208551.51931869 380790.75505211 110943.34589895 119016.28085124
 225036.49824751 167993.3044678   81231.58577615 180648.11132079
 222031.04718466 232445.70522639 132824.31261157  78730.62589463
 253609.27979708 137376.83810402 109944.2572823   98765.42184157
 204252.28279548 168427.98108038 280851.64717125 252533.50647094
  97187.75335617 213571.91396261 191945.02123752 170878.00434004
 159146.88007757 229012.75486622  82371.7869235  143988.97667311
 173499.04027741 204828.90095995 183444.14258936 206886.047962
  -9528.33286076 100706.94712     98569.80384367 227675.07244954
 150311.68937109 323979.62623651 198468.9058216   57394.84379213
 207088.55841654 104693.99933562 194249.66404674  99112.95973974
 234041.25154372 214732.44512006 202185.27311015 105592.51402
  87744.81195713 242057.84919119 188102.36168592 191200.70555198
 273008.37795304 246469.67961109 104436.91474873 148688.46720523
 264801.51786141 108531.3164552  219260.16086342  99389.26532717
 230315.7258856  157348.28994965 113518.50868429 205123.6584691
  71728.96692793  87441.29971611  88721.15142168 305654.35772646
  37261.65477772 188671.16973729 227593.17636063  58123.97821558
 138735.75969021 108729.41485473 444643.20682429 115448.44107785
 108578.61781732 283704.40835976 204686.3609425  247035.30252065
 212433.8917326   92581.25218114 109190.50604295 212198.4502553
 167123.7354618  157501.77197878 149674.35399888 183174.15665458
  87998.30938585 168960.89961403  75571.05574281 171325.65884648
 175934.80484077 170937.22250001 124903.04347655 210155.92915927
```

```
 226572.7428022  124289.11964626 182266.37814004 259226.1477213
 223580.97290061]
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-packages\sklearn\linear_model\stochastic_gradient.p
y:128: FutureWarning: max_iter and tol parameters have been added in <class 'sklearn.linear_model.stochastic_gradient.SGDRegres
sor'> in 0.19. If both are left unset, they default to max_iter=5 and tol=None. If tol is not None, max_iter defaults to max_it
er=1000. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  "and default tol will be 1e-3." % type(self), FutureWarning)
```

In [84]:
```
sgd_mse=metrics.mean_squared_error(y_test, sgd_y_pred)
print("SGD_MSE:",sgd_mse)
```

```
SGD_MSE: 1670797465.899073
```

In [85]:
```
#Step-8:[Predict using Ridge Regressor]
from sklearn.linear_model import Ridge
ridge=Ridge()
ridge.fit(ss_X_train, y_train)
ridge_y_pred=ridge.predict(ss_X_test)
print("Predictions of scaled data using RIDGERegression:", ridge_y_pred)
```

```
Predictions of scaled data using RIDGERegression: [257421.40327463 111048.9390094  100148.22166498 204007.69016257
 207318.26438772  38130.16627249 234150.19679266 205083.79807365
 186990.35389951 235496.28665194 100966.52935343 303934.6316491
 101890.94872864 288779.9847382  204801.28874564 145108.44985328
 248228.38505616 151485.36209941 209733.22745174 278197.43944307
  93455.660932   153804.93005229 163194.49635281 241430.24636124
 372368.42646845 221632.71135354 119760.90284338 100047.5884376
 319413.64552304 172029.21920288 258742.75120254 199493.44526769
 156253.40634554 142260.62082567 204826.18067282 379453.76863328
 124049.03967134 140938.6802175  273742.3805046  212502.29963135
 169496.97954218 123279.27640318 200131.43501214 377624.34223346
 147067.67507844 283730.24445759 106654.11168627 221721.98334682
  87859.60159348 294987.18055782 125202.71399397  53519.12559164
 133851.65120493 170814.1145784  213257.89409126 110890.21052625
 103962.51734198 200705.10120899 137825.92554881 311801.15478353
  53110.79651664 184799.60969549 154734.86643989 298380.40493951
  78246.19078424 323702.02510823 169466.06274078 112823.93682163
 200532.47961718 205732.74348111 130430.90213612 271576.54141389
 119517.69643867 126813.06030995  77948.0791305  138459.94428048
  35594.69872609 163851.24303446 271911.16049013 115170.8409187
 319926.92800432 218539.4849081  291693.99915377 211226.11308601
  93921.33416755 262946.95574993 141809.88774304 118408.82479371
 143529.08575461 278395.45721678 265527.26500266 300767.84214434
 182712.18425798 168624.19471717 148468.01120129 213828.16525828
 247699.43190121 221655.98604164 274448.97443046 242853.53739015
 111008.88869602 100159.22278113 193584.21304234 205986.02391762
 154621.79158788 217316.85068612 210896.31853727 125867.74499658
 171334.06377111 224552.40663287 155561.48910925 112029.96523957
 316553.30396301 297970.38898708 335452.566429    86702.91494711
 146732.96845766 228454.19721056 123467.16116299 204859.18689769
 217316.45502934 120778.23382013 134362.92386476 223895.62311331
 138354.30270106 210759.85184412 145371.25618934 291104.98829885
 183135.45289803  98714.38796372 323759.01921353 170426.52509233
 123724.57076041 134394.63803651 309274.63604244 136992.66409203
 282021.23633093 132696.19997056  90813.11962895 164237.83443566
 147281.26846701 162003.41170017 177647.14470976 253988.23631587
 119205.33260247 140118.15986413 236442.68491147 318486.93596213
 105671.08096912 192080.32594001 174466.08136767 146377.97610558
  95151.72242241 250913.00451269 304117.3963575   55131.83869796
 281794.68069314  93471.49702752 138431.7751145  166584.56732144
 209797.30479004 205691.94925081 177054.65626531 248576.0299266
 145449.7351938  133220.86441605 171632.89844548 344767.21487919
 181092.55310354  98456.02843201 131282.04325421 110934.36779141
 131317.3323564  194129.22012653 156885.58294293 268253.70749413
 211776.05309371 151997.19209851 115885.87158074 243633.12876296
 136046.12092616 265371.76336651 302122.63871329  85249.62268466
 145934.87875567 150476.83298301 159184.29159015 232268.61259152
 231568.20715332 139168.63211654 354705.36600677 127677.96529031
 235350.87769955 116731.80224165 103066.91017805 159648.98348195
 149516.96209971 408710.91797554 335360.95624805 290491.2487189
 281759.86238461 275280.43869726 321581.4846304  307495.11977824
 201708.68669459 142026.52881729 157911.49475883 258946.36167615
 207616.5467689  146189.84392725 107127.69428065 151193.37312045
 100806.45474245 297016.85500481 341332.96001979 255156.65509967
 147042.32133389 199835.16351096 127396.28236175 264028.87507232
 121842.35752276 134224.48204848 307690.96067671 170299.19516278
 173634.61888311 566895.37162062 182267.31525211 144230.712434
 199542.41967237 107725.24929291 181340.30552993 328346.05767985
 139169.93510602 197676.26414776 116515.40028887  54920.9736789
 199359.79732592 272909.48207611 119964.1524353  197635.14097922
 231846.26227293 119652.57513809 128795.15181621 289062.25535623
 200794.72224418 364283.23697094 119300.12544332 124055.74846523
 225088.06390043 170924.12061508  93546.44521802 178549.05783355
 215651.32553709 226256.01720286 132397.67484782  73733.38519025
 242947.50811012 141056.2714406  114625.16811529  89328.50332655
 208394.97589475 172744.77142249 270816.4453816  253664.42153466
 103542.72018396 220477.69002155 185184.89845876 169549.94526116
 154899.03211362 224612.69076295  86017.9320044  151268.6967863
 180569.55851465 200116.32845302 179525.24701346 223517.00898661
  73881.921606    86307.09832204 109982.15609072 231440.35424803
 153338.62588431 320803.87482757 194414.65718472  57804.18747409
 203909.81175019 109852.78795055 194673.94964339  99307.14089041
 230248.65801326 211480.25262312 198415.83220879 109779.16815378
  95603.4369218  235431.54757497 184887.1784599  188858.81438931
 265348.10109783 245750.11638293 107449.72804791 164526.92609276
 268206.85567296 116000.3723075  222832.75187172 102708.5394977
 226446.39169701 156039.73352196 112118.26213524 218009.48577393
  81138.79394621  85183.53240152  98120.58664145 304221.85064056
 143825.10439909 195254.90131561 241807.65208575  60898.2165972
 143757.78555659 110569.79788577 438249.6239445  123258.6686974
 116123.99585895 277966.01915233 201690.38568328 237978.53565053
 206247.64338758  99449.19364684 115124.23894791 209778.46658592
 167434.01104627 162852.00375486 151360.56894067 189760.65511031
  96349.80598179 168767.97321607  83076.25780277 171281.01213295
 176931.55688458 175568.05302625 131382.65018549 204547.73197298
```

```
 233033.33528843 125501.91800532 175116.80170639 258825.79733591
 228967.00891861]
```

In [86]:
```
ridge_mse=metrics.mean_squared_error(y_test, ridge_y_pred)
ridge_r2=metrics.r2_score(y_test, sgd_y_pred)
print("RIDGE_MSE:",ridge_mse)
```

```
RIDGE_MSE: 1473019452.6323287
```

In [87]:
```
#Step-8:[Predict using Lasso Regression]
from sklearn.linear_model import Lasso
lasso=Lasso()
lasso.fit(ss_X_train, y_train)
lasso_y_pred=lasso.predict(ss_X_test)
print("Predictions of scaled data using LASSORegression:", lasso_y_pred)
```

```
Predictions of scaled data using LASSORegression: [257421.66197131 111077.7187949  100035.61918286 204024.71698841
 207324.54758549  38037.76044535 234149.0762938  205067.06537914
 186995.76236854 235644.98929291 100980.81814118 304124.32600947
 101772.1791833  288771.22252722 204759.49619302 145026.21553491
 248321.68441805 151528.08346282 209694.08183583 278309.14448759
  93335.15311371 153783.56779992 163228.65012326 241483.60508935
 372429.51050619 221679.58600971 119671.76785971 100045.46561751
 319437.72582219 172082.33265964 258777.40797267 199570.17239239
 156363.52609093 142320.71703113 204809.98100187 379595.39611493
 124166.56112372 141099.38504751 273833.8572836  212527.67447478
 169476.63411453 123302.15760259 200345.3164127  377933.40951119
 146945.90334538 283683.39848174 106687.15479997 221733.415641
  87730.76613375 295006.8035834  125244.58525479  53575.54835447
 133831.34643864 170889.49976306 213275.46196991 110829.42033243
 103891.63327972 200669.15236797 137760.96717691 311838.89428255
  53043.94004267 184860.11037273 154678.25277191 298472.0419572
  78160.84398492 323727.20193706 169432.06791652 112719.49362597
 200537.03858629 205730.2382202  130319.0338602  271572.18419675
 119472.76012381 126676.81795613  77973.93013375 138543.96745565
  35566.47418893 163776.26481063 271932.10184361 115245.10184021
 319947.80376823 218524.65113737 291728.64199646 211184.25132826
  93766.2441843  263013.07200833 141884.14887872 118479.94316044
 143579.25976533 278387.0901624  265542.40107943 300840.57586437
 182676.22636997 168614.55293958 148325.13384923 213772.65343831
 247744.36515991 221694.07398906 274473.40032138 242910.07733685
 111024.32538024 100163.6312411  193529.83383847 206051.9984012
 154580.62661225 217424.37479612 210897.8675806  125862.78260233
 171328.72206544 224472.85497151 155547.8708405  111948.33451201
 316614.91783714 298110.80068015 335577.6410617   86728.45381605
 146691.87441872 228462.03993328 123369.93579043 204857.54373323
 217328.94220476 120601.69683226 134345.84299463 223863.77784628
 138369.30181302 210754.70687603 145421.12522714 291149.08503416
 183093.71939825  98717.31648489 323879.66307012 170451.65427567
 123658.89486774 134346.04127054 309415.83992274 136856.15012239
 282080.39764246 132642.24782293  90805.08020319 164197.20983367
 147307.78749689 162024.48429864 177631.27200197 254051.12619446
 119093.87836389 140079.94586773 236494.44620788 318591.37268903
 105548.04487966 192017.47219484 174529.12056039 146312.31922153
  95155.25228458 250937.191386   304068.78612063  55008.97537976
 281813.60681016  93421.72719678 138282.43504323 166444.38175673
 209754.51753515 205585.33877292 177023.01206168 248437.29278084
 145210.40127495 133272.48067321 171592.92483782 344930.02937232
 181190.92645162  98431.12271089 131285.74545294 110846.83896733
 131266.92367967 194210.32162302 156959.82444772 268332.77077341
 211799.4931586  151961.57337179 115947.01195855 243618.37820513
 136267.49311255 265336.5249903  302183.29729314  85066.61512975
 145993.91535665 150453.58032624 159131.11702019 232247.0411628
 231568.64435723 139159.5315474  355010.16172568 127649.28854378
 235347.25115031 116723.15594943 103097.8087313  159579.48633323
 149485.86637454 408877.8084223  335364.79787411 290491.82929321
 281827.17369033 275344.01943145 321689.95267998 307550.06260339
 201657.13893698 142043.29571018 157877.79211313 258686.36229443
 207597.504559   146126.19537024 107060.58360934 151209.63196986
 100803.88138409 296986.55366535 341413.68242693 255159.09225165
 147004.67858324 199741.60256609 127286.60854165 264039.47903932
 121757.7601844  134168.47750092 307664.59299139 170333.21476984
 173582.30534568 567237.57815422 182257.0770013  144169.57742384
 199532.70579553 107682.4433424  181381.68902914 328403.51993649
 139213.92957116 197651.36277804 116359.69944382  54926.04103468
 199373.73822288 272930.85953307 119891.94030638 197577.32950166
 231874.90352676 119554.80329458 128727.34148946 289156.98065867
 200732.19876914 364217.55430575 119318.00170467 124076.51847101
 225155.15159558 170995.8491121   93628.56726597 178508.96243851
 215656.81266809 226226.64669323 132395.31450612  73477.28786246
 242922.6410713  141045.58981073 114624.5400826   89110.56212213
 208500.76357759 172822.22260698 270785.2583861  253704.15606843
 103518.18195129 220525.08538564 185034.26794748 169525.90813544
 154842.46818304 224609.79313209  85978.1340752  151315.30593602
 180620.69731953 200112.1648663  179501.34198602 223731.81293344
  73647.74981723  86044.36289155 109989.31336521 231531.36663061
 153351.49297773 320891.11855489 194414.35048753  57747.24284662
 203879.31819901 109783.81220302 194641.16474264  99165.08860279
 230240.64044691 211487.44994296 198349.64311187 109741.69039227
  95564.46870923 235405.42730463 184786.80800329 188804.34018413
 265336.71871287 245859.81354429 107410.41353438 164771.90203734
 268301.56371186 116072.68714631 222929.57591009 102598.54783259
 226401.1860395  156092.60471478 112036.64840946 218083.15785633
  81100.88899393  85030.47954977  98197.57559793 304263.8783473
 143748.45509156 195361.4466563  241819.00146461  60942.15059456
 143722.83401196 110480.74297359 438384.98921717 123261.18298112
 116126.99598136 277962.95998619 201695.2940015  237947.3539218
 206133.53792871  99446.01705387 115122.09776624 209784.1140348
 167541.5967022  162969.03673952 151277.96604482 189720.09910655
  96447.55044815 168669.90592379  83070.12662456 171326.97842039
 176873.70242598 175633.95029981 131412.81394597 204533.41322737
```

```
 233136.3863491  125468.90236905 175066.41285199 258869.95615192
 229111.23932392]

C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the number of iterations. Fitting data with v
ery small alpha may cause precision problems.
  ConvergenceWarning)
```

In [88]:
```
lasso_mse=metrics.mean_squared_error(y_test, lasso_y_pred)
print("LASSO_MSE:",lasso_mse)
```

```
LASSO_MSE: 1474731226.7746537
```

In [89]:
```
#Step-9:[RMSE]
import numpy as np

#RMSE without CD
print("RMSE without CD: ",np.sqrt(mse))

#RMSE with CD
print("RMSE with CD: ",np.sqrt(mse_cd))

#RMSE with CD and Standard Scaling
print("RMSE with CD and SS: ",np.sqrt(ss_mse))

#RMSE with CD and MinMaxScaling
print("RMSE with CD and MnMaxScaling: ",np.sqrt(mm_mse))

#RMSE of SGDRegressor with CD and StandardScaler
print("RMSE of SGDRegressor with CD and StandardScaler: ",np.sqrt(sgd_mse))

#RMSE of Ridgecv with CD and Standard Scaler
print("RMSE of Ridgecv with CD and Standard Scaler: ",np.sqrt(ridge_mse))

#RMSE of LassoCV with CD and StandardScaler
print("RMSE of LassoCV with CD and StandardScaler",np.sqrt(lasso_mse))
```

```
RMSE without CD:  38403.48064427505
RMSE with CD:  38403.48064427505
RMSE with CD and SS:  38403.48064427922
RMSE with CD and MnMaxScaling:  38403.480644279196
RMSE of SGDRegressor with CD and StandardScaler:  40875.389489264475
RMSE of Ridgecv with CD and Standard Scaler:  38379.93554752703
RMSE of LassoCV with CD and StandardScaler 38402.22945057557
```

In [ ]:
```

```