LAB 4 : House Price Prediction using LR with Regularization

DINESH KUMAR K

225229108

Step 1 : Import Dataset

In [3]:
```
import pandas as pd
```

In [4]:
```
data=pd.read_csv(r'Ames_House_Sales_Cropped.csv')
data
```

Out[4]:

	BldgType	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinS
0	1Fam	Y	856.0	854.0	0.0	3	706.0	
1	1Fam	Y	1262.0	0.0	0.0	3	978.0	
2	1Fam	Y	920.0	866.0	0.0	3	486.0	
3	1Fam	Y	961.0	756.0	0.0	3	216.0	
4	1Fam	Y	1145.0	1053.0	0.0	4	655.0	
5	1Fam	Y	796.0	566.0	320.0	1	732.0	
6	1Fam	Y	1694.0	0.0	0.0	3	1369.0	
7	1Fam	Y	1107.0	983.0	0.0	3	859.0	3
8	1Fam	Y	1022.0	752.0	0.0	2	0.0	
9	2fmCon	Y	1077.0	0.0	0.0	2	851.0	
10	1Fam	Y	1040.0	0.0	0.0	3	906.0	
11	1Fam	Y	1182.0	1142.0	0.0	4	998.0	
12	1Fam	Y	912.0	0.0	0.0	2	737.0	
13	1Fam	Y	1494.0	0.0	0.0	3	0.0	
14	1Fam	Y	1253.0	0.0	0.0	2	733.0	
15	1Fam	Y	854.0	0.0	0.0	2	0.0	
16	1Fam	Y	1004.0	0.0	0.0	2	578.0	
17	Duplex	Y	1296.0	0.0	0.0	2	0.0	
18	1Fam	Y	1114.0	0.0	0.0	3	646.0	
19	1Fam	Y	1339.0	0.0	0.0	3	504.0	
20	1Fam	Y	1158.0	1218.0	0.0	4	0.0	
21	1Fam	Y	1108.0	0.0	0.0	3	0.0	
22	1Fam	Y	1795.0	0.0	0.0	3	0.0	
23	TwnhsE	Y	1060.0	0.0	0.0	3	840.0	
24	1Fam	Y	1060.0	0.0	0.0	3	188.0	66
25	1Fam	Y	1600.0	0.0	0.0	3	0.0	
26	1Fam	Y	900.0	0.0	0.0	3	234.0	48
27	1Fam	Y	1704.0	0.0	0.0	3	1218.0	
28	1Fam	Y	1600.0	0.0	0.0	2	1277.0	
29	1Fam	N	520.0	0.0	0.0	1	0.0	
...	...	...	...	...	...	...	...	
1349	1Fam	Y	1048.0	510.0	0.0	3	580.0	
1350	1Fam	Y	804.0	0.0	0.0	2	510.0	

	BldgType	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinS
1351	1Fam	Y	1440.0	0.0	0.0	3	678.0	
1352	1Fam	Y	734.0	1104.0	0.0	4	0.0	
1353	TwnhsE	Y	958.0	0.0	0.0	2	958.0	
1354	1Fam	Y	968.0	0.0	0.0	4	0.0	
1355	1Fam	Y	962.0	830.0	0.0	3	0.0	
1356	1Fam	Y	1126.0	0.0	0.0	3	936.0	
1357	1Fam	Y	1537.0	0.0	0.0	3	0.0	
1358	1Fam	Y	864.0	0.0	0.0	3	616.0	
1359	1Fam	Y	1932.0	0.0	304.0	2	1336.0	
1360	1Fam	Y	1236.0	0.0	0.0	2	600.0	
1361	1Fam	Y	1040.0	685.0	0.0	3	315.0	11
1362	1Fam	Y	1423.0	748.0	0.0	3	0.0	
1363	TwnhsE	Y	848.0	0.0	0.0	1	697.0	
1364	1Fam	Y	1026.0	981.0	0.0	3	765.0	
1365	1Fam	N	952.0	0.0	0.0	2	0.0	
1366	1Fam	Y	1422.0	0.0	0.0	3	0.0	
1367	1Fam	Y	913.0	0.0	0.0	3	187.0	62
1368	1Fam	Y	1188.0	0.0	0.0	3	593.0	
1369	1Fam	Y	1220.0	870.0	0.0	3	1079.0	
1370	1Fam	N	796.0	550.0	0.0	2	0.0	
1371	1Fam	Y	1578.0	0.0	0.0	3	0.0	
1372	TwnhsE	Y	1072.0	0.0	0.0	2	547.0	
1373	1Fam	Y	1221.0	0.0	0.0	2	410.0	
1374	1Fam	Y	953.0	694.0	0.0	3	0.0	
1375	1Fam	Y	2073.0	0.0	0.0	3	790.0	16
1376	1Fam	Y	1188.0	1152.0	0.0	4	275.0	
1377	1Fam	Y	1078.0	0.0	0.0	2	49.0	102
1378	1Fam	Y	1256.0	0.0	0.0	3	830.0	29

1379 rows × 39 columns

In [5]:
```
data.head()
```

Out[5]:

	BldgType	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2
0	1Fam	Y	856.0	854.0	0.0	3	706.0	0.0
1	1Fam	Y	1262.0	0.0	0.0	3	978.0	0.0
2	1Fam	Y	920.0	866.0	0.0	3	486.0	0.0
3	1Fam	Y	961.0	756.0	0.0	3	216.0	0.0
4	1Fam	Y	1145.0	1053.0	0.0	4	655.0	0.0

5 rows × 39 columns

In [6]:
```
data.shape
```

Out[6]:
```
(1379, 39)
```

In [7]:
```
data.columns
```

Out[7]:
```
Index(['BldgType', 'CentralAir', '1stFlrSF', '2ndFlrSF', '3SsnPorch',
       'BedroomAbvGr', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtFullBath',
       'BsmtHalfBath', 'BsmtUnfSF', 'EnclosedPorch', 'Fireplaces', 'FullBath',
       'GarageArea', 'GarageCars', 'GarageYrBlt', 'GrLivArea', 'HalfBath',
       'KitchenAbvGr', 'LotArea', 'LotFrontage', 'LowQualFinSF', 'MSSubClass',
       'MasVnrArea', 'MiscVal', 'MoSold', 'OpenPorchSF', 'OverallCond',
       'OverallQual', 'PoolArea', 'ScreenPorch', 'TotRmsAbvGrd', 'TotalBsmtSF',
       'WoodDeckSF', 'YearBuilt', 'YearRemodAdd', 'YrSold', 'SalePrice'],
      dtype='object')
```

In [8]:
```
data.dtypes
```

Out[8]:
```
BldgType          object
CentralAir        object
1stFlrSF         float64
2ndFlrSF         float64
3SsnPorch        float64
BedroomAbvGr       int64
BsmtFinSF1       float64
BsmtFinSF2       float64
BsmtFullBath       int64
BsmtHalfBath       int64
BsmtUnfSF        float64
EnclosedPorch    float64
Fireplaces         int64
FullBath           int64
GarageArea       float64
GarageCars         int64
GarageYrBlt      float64
GrLivArea        float64
HalfBath           int64
KitchenAbvGr       int64
LotArea          float64
LotFrontage      float64
LowQualFinSF     float64
MSSubClass         int64
MasVnrArea       float64
MiscVal          float64
MoSold             int64
OpenPorchSF      float64
OverallCond        int64
OverallQual        int64
PoolArea         float64
ScreenPorch      float64
TotRmsAbvGrd       int64
TotalBsmtSF      float64
WoodDeckSF       float64
YearBuilt          int64
YearRemodAdd       int64
YrSold             int64
SalePrice        float64
dtype: object
```

In [9]:
```
data.info
```

Out[9]:
```
<bound method DataFrame.info of      BldgType CentralAir  1stFlrSF  2ndFlrSF  3SsnPorch  BedroomAbvGr  \
0        1Fam          Y     856.0     854.0        0.0             3
1        1Fam          Y    1262.0       0.0        0.0             3
2        1Fam          Y     920.0     866.0        0.0             3
3        1Fam          Y     961.0     756.0        0.0             3
4        1Fam          Y    1145.0    1053.0        0.0             4
5        1Fam          Y     796.0     566.0      320.0             1
6        1Fam          Y    1694.0       0.0        0.0             3
7        1Fam          Y    1107.0     983.0        0.0             3
8        1Fam          Y    1022.0     752.0        0.0             2
9      2fmCon          Y    1077.0       0.0        0.0             2
10       1Fam          Y    1040.0       0.0        0.0             3
11       1Fam          Y    1182.0    1142.0        0.0             4
12       1Fam          Y     912.0       0.0        0.0             2
13       1Fam          Y    1494.0       0.0        0.0             3
14       1Fam          Y    1253.0       0.0        0.0             2
15       1Fam          Y     854.0       0.0        0.0             2
16       1Fam          Y    1004.0       0.0        0.0             2
```

In [10]:
```
data.CentralAir.value_counts()
```

Out[10]:
```
Y    1310
N      69
Name: CentralAir, dtype: int64
```

In [11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1379 entries, 0 to 1378
Data columns (total 39 columns):
BldgType         1379 non-null object
CentralAir       1379 non-null object
1stFlrSF         1379 non-null float64
2ndFlrSF         1379 non-null float64
3SsnPorch        1379 non-null float64
BedroomAbvGr     1379 non-null int64
BsmtFinSF1       1379 non-null float64
BsmtFinSF2       1379 non-null float64
BsmtFullBath     1379 non-null int64
BsmtHalfBath     1379 non-null int64
BsmtUnfSF        1379 non-null float64
EnclosedPorch    1379 non-null float64
Fireplaces       1379 non-null int64
FullBath         1379 non-null int64
GarageArea       1379 non-null float64
GarageCars       1379 non-null int64
GarageYrBlt      1379 non-null float64
GrLivArea        1379 non-null float64
HalfBath         1379 non-null int64
KitchenAbvGr     1379 non-null int64
LotArea          1379 non-null float64
LotFrontage      1379 non-null float64
LowQualFinSF     1379 non-null float64
MSSubClass       1379 non-null int64
MasVnrArea       1379 non-null float64
MiscVal          1379 non-null float64
MoSold           1379 non-null int64
OpenPorchSF      1379 non-null float64
OverallCond      1379 non-null int64
OverallQual      1379 non-null int64
PoolArea         1379 non-null float64
ScreenPorch      1379 non-null float64
TotRmsAbvGrd     1379 non-null int64
TotalBsmtSF      1379 non-null float64
WoodDeckSF       1379 non-null float64
YearBuilt        1379 non-null int64
YearRemodAdd     1379 non-null int64
YrSold           1379 non-null int64
SalePrice        1379 non-null float64
dtypes: float64(21), int64(16), object(2)
memory usage: 420.2+ KB
```

Step 2 : Predict Sale Price without Categorical features

In [14]:
```
df=data.drop("BldgType",axis=1)
df
```

Out[14]:

	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtF
0	Y	856.0	854.0	0.0	3	706.0	0.0	
1	Y	1262.0	0.0	0.0	3	978.0	0.0	
2	Y	920.0	866.0	0.0	3	486.0	0.0	
3	Y	961.0	756.0	0.0	3	216.0	0.0	
4	Y	1145.0	1053.0	0.0	4	655.0	0.0	
5	Y	796.0	566.0	320.0	1	732.0	0.0	
6	Y	1694.0	0.0	0.0	3	1369.0	0.0	
7	Y	1107.0	983.0	0.0	3	859.0	32.0	
8	Y	1022.0	752.0	0.0	2	0.0	0.0	
9	Y	1077.0	0.0	0.0	2	851.0	0.0	
10	Y	1040.0	0.0	0.0	3	906.0	0.0	
11	Y	1182.0	1142.0	0.0	4	998.0	0.0	
12	Y	912.0	0.0	0.0	2	737.0	0.0	
13	Y	1494.0	0.0	0.0	3	0.0	0.0	
14	Y	1253.0	0.0	0.0	2	733.0	0.0	
15	Y	854.0	0.0	0.0	2	0.0	0.0	
16	Y	1004.0	0.0	0.0	2	578.0	0.0	
17	Y	1296.0	0.0	0.0	2	0.0	0.0	
18	Y	1114.0	0.0	0.0	3	646.0	0.0	
19	Y	1339.0	0.0	0.0	3	504.0	0.0	
20	Y	1158.0	1218.0	0.0	4	0.0	0.0	
21	Y	1108.0	0.0	0.0	3	0.0	0.0	
22	Y	1795.0	0.0	0.0	3	0.0	0.0	
23	Y	1060.0	0.0	0.0	3	840.0	0.0	
24	Y	1060.0	0.0	0.0	3	188.0	668.0	
25	Y	1600.0	0.0	0.0	3	0.0	0.0	
26	Y	900.0	0.0	0.0	3	234.0	486.0	
27	Y	1704.0	0.0	0.0	3	1218.0	0.0	
28	Y	1600.0	0.0	0.0	2	1277.0	0.0	
29	N	520.0	0.0	0.0	1	0.0	0.0	
...	...	...	...	...	...	...	...	...
1349	Y	1048.0	510.0	0.0	3	580.0	0.0	
1350	Y	804.0	0.0	0.0	2	510.0	0.0	

	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtF
1351	Y	1440.0	0.0	0.0	3	678.0	0.0	
1352	Y	734.0	1104.0	0.0	4	0.0	0.0	
1353	Y	958.0	0.0	0.0	2	958.0	0.0	
1354	Y	968.0	0.0	0.0	4	0.0	0.0	
1355	Y	962.0	830.0	0.0	3	0.0	0.0	
1356	Y	1126.0	0.0	0.0	3	936.0	0.0	
1357	Y	1537.0	0.0	0.0	3	0.0	0.0	
1358	Y	864.0	0.0	0.0	3	616.0	0.0	
1359	Y	1932.0	0.0	304.0	2	1336.0	0.0	
1360	Y	1236.0	0.0	0.0	2	600.0	0.0	
1361	Y	1040.0	685.0	0.0	3	315.0	110.0	
1362	Y	1423.0	748.0	0.0	3	0.0	0.0	
1363	Y	848.0	0.0	0.0	1	697.0	0.0	
1364	Y	1026.0	981.0	0.0	3	765.0	0.0	
1365	N	952.0	0.0	0.0	2	0.0	0.0	
1366	Y	1422.0	0.0	0.0	3	0.0	0.0	
1367	Y	913.0	0.0	0.0	3	187.0	627.0	
1368	Y	1188.0	0.0	0.0	3	593.0	0.0	
1369	Y	1220.0	870.0	0.0	3	1079.0	0.0	
1370	N	796.0	550.0	0.0	2	0.0	0.0	
1371	Y	1578.0	0.0	0.0	3	0.0	0.0	
1372	Y	1072.0	0.0	0.0	2	547.0	0.0	
1373	Y	1221.0	0.0	0.0	2	410.0	0.0	
1374	Y	953.0	694.0	0.0	3	0.0	0.0	
1375	Y	2073.0	0.0	0.0	3	790.0	163.0	
1376	Y	1188.0	1152.0	0.0	4	275.0	0.0	
1377	Y	1078.0	0.0	0.0	2	49.0	1029.0	
1378	Y	1256.0	0.0	0.0	3	830.0	290.0	

1379 rows × 38 columns

In [16]:
```
df2=df.drop("CentralAir",axis=1)
df2
```

Out[16]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
0	856.0	854.0	0.0	3	706.0	0.0	1	
1	1262.0	0.0	0.0	3	978.0	0.0	0	
2	920.0	866.0	0.0	3	486.0	0.0	1	
3	961.0	756.0	0.0	3	216.0	0.0	1	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	
5	796.0	566.0	320.0	1	732.0	0.0	1	
6	1694.0	0.0	0.0	3	1369.0	0.0	1	
7	1107.0	983.0	0.0	3	859.0	32.0	1	
8	1022.0	752.0	0.0	2	0.0	0.0	0	
9	1077.0	0.0	0.0	2	851.0	0.0	1	
10	1040.0	0.0	0.0	3	906.0	0.0	1	
11	1182.0	1142.0	0.0	4	998.0	0.0	1	
12	912.0	0.0	0.0	2	737.0	0.0	1	
13	1494.0	0.0	0.0	3	0.0	0.0	0	
14	1253.0	0.0	0.0	2	733.0	0.0	1	
15	854.0	0.0	0.0	2	0.0	0.0	0	
16	1004.0	0.0	0.0	2	578.0	0.0	1	
17	1296.0	0.0	0.0	2	0.0	0.0	0	
18	1114.0	0.0	0.0	3	646.0	0.0	1	
19	1339.0	0.0	0.0	3	504.0	0.0	0	
20	1158.0	1218.0	0.0	4	0.0	0.0	0	
21	1108.0	0.0	0.0	3	0.0	0.0	0	
22	1795.0	0.0	0.0	3	0.0	0.0	0	
23	1060.0	0.0	0.0	3	840.0	0.0	1	
24	1060.0	0.0	0.0	3	188.0	668.0	1	
25	1600.0	0.0	0.0	3	0.0	0.0	0	
26	900.0	0.0	0.0	3	234.0	486.0	0	
27	1704.0	0.0	0.0	3	1218.0	0.0	1	
28	1600.0	0.0	0.0	2	1277.0	0.0	1	
29	520.0	0.0	0.0	1	0.0	0.0	0	
...	...	...	...	...	...	...	...	
1349	1048.0	510.0	0.0	3	580.0	0.0	1	
1350	804.0	0.0	0.0	2	510.0	0.0	1	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
1351	1440.0	0.0	0.0	3	678.0	0.0	0	
1352	734.0	1104.0	0.0	4	0.0	0.0	0	
1353	958.0	0.0	0.0	2	958.0	0.0	0	
1354	968.0	0.0	0.0	4	0.0	0.0	0	
1355	962.0	830.0	0.0	3	0.0	0.0	1	
1356	1126.0	0.0	0.0	3	936.0	0.0	1	
1357	1537.0	0.0	0.0	3	0.0	0.0	1	
1358	864.0	0.0	0.0	3	616.0	0.0	0	
1359	1932.0	0.0	304.0	2	1336.0	0.0	1	
1360	1236.0	0.0	0.0	2	600.0	0.0	1	
1361	1040.0	685.0	0.0	3	315.0	110.0	0	
1362	1423.0	748.0	0.0	3	0.0	0.0	0	
1363	848.0	0.0	0.0	1	697.0	0.0	1	
1364	1026.0	981.0	0.0	3	765.0	0.0	1	
1365	952.0	0.0	0.0	2	0.0	0.0	0	
1366	1422.0	0.0	0.0	3	0.0	0.0	0	
1367	913.0	0.0	0.0	3	187.0	627.0	1	
1368	1188.0	0.0	0.0	3	593.0	0.0	0	
1369	1220.0	870.0	0.0	3	1079.0	0.0	1	
1370	796.0	550.0	0.0	2	0.0	0.0	0	
1371	1578.0	0.0	0.0	3	0.0	0.0	0	
1372	1072.0	0.0	0.0	2	547.0	0.0	1	
1373	1221.0	0.0	0.0	2	410.0	0.0	1	
1374	953.0	694.0	0.0	3	0.0	0.0	0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	

1379 rows × 37 columns

In [18]:
```
p=data.pop("BldgType")
p
```

Out[18]:
```
0         1Fam
1         1Fam
2         1Fam
3         1Fam
4         1Fam
5         1Fam
6         1Fam
7         1Fam
8         1Fam
9       2fmCon
10        1Fam
11        1Fam
12        1Fam
13        1Fam
14        1Fam
15        1Fam
16        1Fam
17      Duplex
18        1Fam
19        1Fam
20        1Fam
21        1Fam
22        1Fam
23      TwnhsE
24        1Fam
25        1Fam
26        1Fam
27        1Fam
28        1Fam
29        1Fam
          ...
1349      1Fam
1350      1Fam
1351      1Fam
1352      1Fam
1353    TwnhsE
1354      1Fam
1355      1Fam
1356      1Fam
1357      1Fam
1358      1Fam
1359      1Fam
1360      1Fam
1361      1Fam
1362      1Fam
1363    TwnhsE
1364      1Fam
1365      1Fam
1366      1Fam
1367      1Fam
1368      1Fam
1369      1Fam
1370      1Fam
1371      1Fam
```

```
1372    TwnhsE
1373      1Fam
1374      1Fam
1375      1Fam
1376      1Fam
1377      1Fam
1378      1Fam
Name: BldgType, Length: 1379, dtype: object
```

In [19]:
```
p1=data.pop("CentralAir")
p1
```

Out[19]:
```
0       Y
1       Y
2       Y
3       Y
4       Y
5       Y
6       Y
7       Y
8       Y
9       Y
10      Y
11      Y
12      Y
13      Y
14      Y
15      Y
16      Y
17      Y
18      Y
19      Y
20      Y
21      Y
22      Y
23      Y
24      Y
25      Y
26      Y
27      Y
28      Y
29      N
       ..
1349    Y
1350    Y
1351    Y
1352    Y
1353    Y
1354    Y
1355    Y
1356    Y
1357    Y
1358    Y
1359    Y
1360    Y
1361    Y
1362    Y
1363    Y
1364    Y
1365    N
1366    Y
1367    Y
1368    Y
1369    Y
1370    N
1371    Y
```

```
1372    Y
1373    Y
1374    Y
1375    Y
1376    Y
1377    Y
1378    Y
Name: CentralAir, Length: 1379, dtype: object
```

In [20]:
```
data
```

Out[20]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
0	856.0	854.0	0.0	3	706.0	0.0	1	
1	1262.0	0.0	0.0	3	978.0	0.0	0	
2	920.0	866.0	0.0	3	486.0	0.0	1	
3	961.0	756.0	0.0	3	216.0	0.0	1	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	
5	796.0	566.0	320.0	1	732.0	0.0	1	
6	1694.0	0.0	0.0	3	1369.0	0.0	1	
7	1107.0	983.0	0.0	3	859.0	32.0	1	
8	1022.0	752.0	0.0	2	0.0	0.0	0	
9	1077.0	0.0	0.0	2	851.0	0.0	1	
10	1040.0	0.0	0.0	3	906.0	0.0	1	
11	1182.0	1142.0	0.0	4	998.0	0.0	1	
12	912.0	0.0	0.0	2	737.0	0.0	1	
13	1494.0	0.0	0.0	3	0.0	0.0	0	
14	1253.0	0.0	0.0	2	733.0	0.0	1	
15	854.0	0.0	0.0	2	0.0	0.0	0	
16	1004.0	0.0	0.0	2	578.0	0.0	1	
17	1296.0	0.0	0.0	2	0.0	0.0	0	
18	1114.0	0.0	0.0	3	646.0	0.0	1	
19	1339.0	0.0	0.0	3	504.0	0.0	0	
20	1158.0	1218.0	0.0	4	0.0	0.0	0	
21	1108.0	0.0	0.0	3	0.0	0.0	0	
22	1795.0	0.0	0.0	3	0.0	0.0	0	
23	1060.0	0.0	0.0	3	840.0	0.0	1	
24	1060.0	0.0	0.0	3	188.0	668.0	1	
25	1600.0	0.0	0.0	3	0.0	0.0	0	
26	900.0	0.0	0.0	3	234.0	486.0	0	
27	1704.0	0.0	0.0	3	1218.0	0.0	1	
28	1600.0	0.0	0.0	2	1277.0	0.0	1	
29	520.0	0.0	0.0	1	0.0	0.0	0	
...	...	...	...	...	...	...	...	
1349	1048.0	510.0	0.0	3	580.0	0.0	1	
1350	804.0	0.0	0.0	2	510.0	0.0	1	
1351	1440.0	0.0	0.0	3	678.0	0.0	0	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
1352	734.0	1104.0	0.0	4	0.0	0.0	0	
1353	958.0	0.0	0.0	2	958.0	0.0	0	
1354	968.0	0.0	0.0	4	0.0	0.0	0	
1355	962.0	830.0	0.0	3	0.0	0.0	1	
1356	1126.0	0.0	0.0	3	936.0	0.0	1	
1357	1537.0	0.0	0.0	3	0.0	0.0	1	
1358	864.0	0.0	0.0	3	616.0	0.0	0	
1359	1932.0	0.0	304.0	2	1336.0	0.0	1	
1360	1236.0	0.0	0.0	2	600.0	0.0	1	
1361	1040.0	685.0	0.0	3	315.0	110.0	0	
1362	1423.0	748.0	0.0	3	0.0	0.0	0	
1363	848.0	0.0	0.0	1	697.0	0.0	1	
1364	1026.0	981.0	0.0	3	765.0	0.0	1	
1365	952.0	0.0	0.0	2	0.0	0.0	0	
1366	1422.0	0.0	0.0	3	0.0	0.0	0	
1367	913.0	0.0	0.0	3	187.0	627.0	1	
1368	1188.0	0.0	0.0	3	593.0	0.0	0	
1369	1220.0	870.0	0.0	3	1079.0	0.0	1	
1370	796.0	550.0	0.0	2	0.0	0.0	0	
1371	1578.0	0.0	0.0	3	0.0	0.0	0	
1372	1072.0	0.0	0.0	2	547.0	0.0	1	
1373	1221.0	0.0	0.0	2	410.0	0.0	1	
1374	953.0	694.0	0.0	3	0.0	0.0	0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	

1379 rows × 37 columns

In [21]:
```
y=data[["SalePrice"]]
y
```

Out[21]:

	SalePrice
0	208500.0
1	181500.0
2	223500.0
3	140000.0
4	250000.0
5	143000.0
6	307000.0
7	200000.0
8	129900.0
9	118000.0
10	129500.0
11	345000.0
12	144000.0
13	279500.0
14	157000.0
15	132000.0
16	149000.0
17	90000.0
18	159000.0
19	139000.0
20	325300.0
21	139400.0
22	230000.0
23	129900.0
24	154000.0
25	256300.0
26	134800.0
27	306000.0
28	207500.0
29	68500.0
...	...
1349	140000.0
1350	119000.0

	SalePrice
1351	182900.0
1352	192140.0
1353	143750.0
1354	64500.0
1355	186500.0
1356	160000.0
1357	174000.0
1358	120500.0
1359	394617.0
1360	149700.0
1361	197000.0
1362	191000.0
1363	149300.0
1364	310000.0
1365	121000.0
1366	179600.0
1367	129000.0
1368	157900.0
1369	240000.0
1370	112000.0
1371	287090.0
1372	145000.0
1373	185000.0
1374	175000.0
1375	210000.0
1376	266500.0
1377	142125.0
1378	147500.0

1379 rows × 1 columns

In [22]:
```
X=df2.drop("SalePrice",axis=1)
X
```

Out[22]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
0	856.0	854.0	0.0	3	706.0	0.0	1	
1	1262.0	0.0	0.0	3	978.0	0.0	0	
2	920.0	866.0	0.0	3	486.0	0.0	1	
3	961.0	756.0	0.0	3	216.0	0.0	1	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	
5	796.0	566.0	320.0	1	732.0	0.0	1	
6	1694.0	0.0	0.0	3	1369.0	0.0	1	
7	1107.0	983.0	0.0	3	859.0	32.0	1	
8	1022.0	752.0	0.0	2	0.0	0.0	0	
9	1077.0	0.0	0.0	2	851.0	0.0	1	
10	1040.0	0.0	0.0	3	906.0	0.0	1	
11	1182.0	1142.0	0.0	4	998.0	0.0	1	
12	912.0	0.0	0.0	2	737.0	0.0	1	
13	1494.0	0.0	0.0	3	0.0	0.0	0	
14	1253.0	0.0	0.0	2	733.0	0.0	1	
15	854.0	0.0	0.0	2	0.0	0.0	0	
16	1004.0	0.0	0.0	2	578.0	0.0	1	
17	1296.0	0.0	0.0	2	0.0	0.0	0	
18	1114.0	0.0	0.0	3	646.0	0.0	1	
19	1339.0	0.0	0.0	3	504.0	0.0	0	
20	1158.0	1218.0	0.0	4	0.0	0.0	0	
21	1108.0	0.0	0.0	3	0.0	0.0	0	
22	1795.0	0.0	0.0	3	0.0	0.0	0	
23	1060.0	0.0	0.0	3	840.0	0.0	1	
24	1060.0	0.0	0.0	3	188.0	668.0	1	
25	1600.0	0.0	0.0	3	0.0	0.0	0	
26	900.0	0.0	0.0	3	234.0	486.0	0	
27	1704.0	0.0	0.0	3	1218.0	0.0	1	
28	1600.0	0.0	0.0	2	1277.0	0.0	1	
29	520.0	0.0	0.0	1	0.0	0.0	0	
...	...	...	...	...	...	...	...	
1349	1048.0	510.0	0.0	3	580.0	0.0	1	
1350	804.0	0.0	0.0	2	510.0	0.0	1	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
1351	1440.0	0.0	0.0	3	678.0	0.0	0	
1352	734.0	1104.0	0.0	4	0.0	0.0	0	
1353	958.0	0.0	0.0	2	958.0	0.0	0	
1354	968.0	0.0	0.0	4	0.0	0.0	0	
1355	962.0	830.0	0.0	3	0.0	0.0	1	
1356	1126.0	0.0	0.0	3	936.0	0.0	1	
1357	1537.0	0.0	0.0	3	0.0	0.0	1	
1358	864.0	0.0	0.0	3	616.0	0.0	0	
1359	1932.0	0.0	304.0	2	1336.0	0.0	1	
1360	1236.0	0.0	0.0	2	600.0	0.0	1	
1361	1040.0	685.0	0.0	3	315.0	110.0	0	
1362	1423.0	748.0	0.0	3	0.0	0.0	0	
1363	848.0	0.0	0.0	1	697.0	0.0	1	
1364	1026.0	981.0	0.0	3	765.0	0.0	1	
1365	952.0	0.0	0.0	2	0.0	0.0	0	
1366	1422.0	0.0	0.0	3	0.0	0.0	0	
1367	913.0	0.0	0.0	3	187.0	627.0	1	
1368	1188.0	0.0	0.0	3	593.0	0.0	0	
1369	1220.0	870.0	0.0	3	1079.0	0.0	1	
1370	796.0	550.0	0.0	2	0.0	0.0	0	
1371	1578.0	0.0	0.0	3	0.0	0.0	0	
1372	1072.0	0.0	0.0	2	547.0	0.0	1	
1373	1221.0	0.0	0.0	2	410.0	0.0	1	
1374	953.0	694.0	0.0	3	0.0	0.0	0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	

1379 rows × 36 columns

In [23]:
```
from sklearn.model_selection import train_test_split
```

In [24]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=42
```

In [25]:
```
X_train
```

Out[25]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
1193	1337.0	0.0	0.0	2	266.0	0.0	1	
910	1800.0	0.0	0.0	2	0.0	0.0	0	
1068	1328.0	653.0	0.0	4	622.0	0.0	1	
1196	2018.0	0.0	0.0	3	0.0	0.0	0	
1102	959.0	712.0	0.0	3	786.0	0.0	1	
447	970.0	0.0	0.0	2	630.0	0.0	1	
796	1701.0	0.0	0.0	3	1390.0	0.0	1	
543	1320.0	0.0	0.0	3	328.0	551.0	1	
901	768.0	0.0	0.0	2	660.0	0.0	0	
968	1264.0	0.0	0.0	3	697.0	0.0	1	
411	904.0	0.0	0.0	2	0.0	0.0	0	
96	1535.0	0.0	0.0	4	0.0	0.0	0	
429	624.0	720.0	0.0	4	0.0	0.0	0	
361	784.0	0.0	0.0	2	0.0	0.0	0	
933	778.0	798.0	0.0	3	0.0	0.0	0	
588	1422.0	0.0	0.0	3	0.0	0.0	0	
156	854.0	0.0	0.0	2	360.0	0.0	0	
528	1389.0	0.0	0.0	2	1071.0	123.0	1	
654	616.0	0.0	0.0	2	616.0	0.0	0	
857	1636.0	0.0	0.0	3	63.0	0.0	1	
1237	1294.0	0.0	0.0	3	1200.0	0.0	1	
534	1496.0	636.0	0.0	1	1441.0	0.0	1	
1078	1466.0	1362.0	0.0	4	1150.0	0.0	1	
1190	1050.0	0.0	0.0	2	504.0	0.0	0	
1312	869.0	349.0	0.0	3	375.0	0.0	0	
371	1144.0	0.0	0.0	3	739.0	0.0	1	
677	848.0	0.0	0.0	1	662.0	0.0	1	
308	596.0	596.0	0.0	3	0.0	0.0	0	
1371	1578.0	0.0	0.0	3	0.0	0.0	0	
710	866.0	902.0	0.0	3	20.0	0.0	0	
...	...	...	...	...	...	...	...	
474	1801.0	0.0	0.0	1	1247.0	0.0	1	
856	1063.0	0.0	0.0	3	354.0	290.0	1	
747	1086.0	809.0	0.0	3	0.0	0.0	0	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
252	1095.0	844.0	0.0	3	0.0	0.0	0	
21	1108.0	0.0	0.0	3	0.0	0.0	0	
1337	1569.0	0.0	0.0	1	988.0	0.0	0	
459	1484.0	0.0	0.0	3	998.0	0.0	0	
1184	760.0	896.0	0.0	3	0.0	0.0	0	
276	910.0	648.0	0.0	4	420.0	0.0	0	
955	1022.0	0.0	0.0	2	247.0	465.0	1	
1215	1582.0	0.0	0.0	4	812.0	0.0	0	
385	1050.0	1028.0	0.0	3	789.0	0.0	1	
805	1779.0	0.0	0.0	3	306.0	1085.0	1	
343	790.0	784.0	0.0	3	712.0	0.0	1	
769	1008.0	0.0	0.0	2	486.0	0.0	0	
1332	944.0	896.0	0.0	3	666.0	0.0	1	
130	928.0	836.0	0.0	3	821.0	0.0	1	
871	936.0	785.0	0.0	3	814.0	0.0	0	
1123	1622.0	0.0	0.0	3	1159.0	0.0	1	
87	964.0	0.0	0.0	2	713.0	0.0	1	
330	1453.0	0.0	0.0	2	1082.0	0.0	1	
1238	1902.0	0.0	0.0	3	1406.0	0.0	1	
466	886.0	0.0	0.0	2	0.0	0.0	0	
121	1216.0	941.0	0.0	4	445.0	0.0	0	
1044	1500.0	1122.0	0.0	3	1032.0	0.0	1	
1095	855.0	601.0	0.0	3	311.0	0.0	0	
1130	815.0	875.0	0.0	3	0.0	0.0	0	
1294	1661.0	0.0	0.0	3	831.0	0.0	1	
860	742.0	742.0	0.0	3	0.0	0.0	0	
1126	1224.0	0.0	0.0	2	883.0	0.0	1	

1034 rows × 36 columns

In [26]:
```
X_test
```

Out[26]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
599	1518.0	0.0	0.0	1	1218.0	0.0	0	
881	925.0	0.0	0.0	2	338.0	466.0	0	
634	1095.0	679.0	0.0	4	0.0	0.0	1	
425	888.0	868.0	0.0	3	742.0	0.0	1	
906	1337.0	0.0	0.0	3	699.0	0.0	1	
1079	672.0	252.0	0.0	2	348.0	0.0	1	
65	1479.0	0.0	0.0	3	1013.0	0.0	1	
1351	1440.0	0.0	0.0	3	678.0	0.0	0	
479	689.0	689.0	0.0	3	141.0	0.0	0	
67	1304.0	983.0	0.0	3	603.0	0.0	0	
939	774.0	456.0	0.0	3	384.0	0.0	1	
573	1940.0	1254.0	0.0	4	428.0	0.0	0	
917	918.0	0.0	0.0	2	0.0	0.0	0	
1054	1734.0	0.0	0.0	3	1004.0	0.0	1	
941	1442.0	0.0	0.0	2	0.0	0.0	0	
1116	1130.0	0.0	0.0	2	821.0	0.0	1	
237	1005.0	1286.0	0.0	4	0.0	0.0	0	
578	1054.0	0.0	0.0	3	763.0	0.0	1	
772	1358.0	0.0	0.0	2	733.0	0.0	1	
303	1898.0	1080.0	0.0	5	0.0	0.0	0	
953	720.0	551.0	0.0	4	0.0	0.0	0	
783	520.0	600.0	0.0	2	0.0	0.0	0	
339	912.0	0.0	0.0	2	773.0	0.0	1	
718	1494.0	0.0	0.0	2	437.0	1057.0	1	
208	2392.0	0.0	0.0	3	56.0	0.0	0	
624	1465.0	915.0	0.0	3	187.0	723.0	0	
1075	1167.0	0.0	0.0	3	645.0	0.0	0	
1040	950.0	0.0	0.0	3	412.0	287.0	0	
575	1476.0	677.0	0.0	3	904.0	0.0	1	
244	1212.0	0.0	0.0	3	506.0	0.0	1	
...	...	...	...	...	...	...	...	
1334	1040.0	0.0	0.0	2	0.0	0.0	0	
199	1236.0	0.0	0.0	2	360.0	0.0	0	
367	961.0	406.0	0.0	4	241.0	391.0	1	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
723	1690.0	1589.0	0.0	4	1416.0	0.0	1	
354	914.0	0.0	0.0	2	298.0	0.0	0	
10	1040.0	0.0	0.0	3	906.0	0.0	1	
147	1392.0	1070.0	168.0	4	57.0	0.0	1	
538	846.0	846.0	0.0	3	0.0	0.0	0	
282	1541.0	0.0	0.0	3	0.0	0.0	0	
298	1472.0	0.0	0.0	3	1036.0	0.0	1	
522	1048.0	0.0	0.0	2	0.0	0.0	0	
291	793.0	325.0	0.0	3	507.0	0.0	1	
503	880.0	844.0	0.0	3	0.0	0.0	0	
903	979.0	979.0	0.0	4	484.0	0.0	0	
930	1001.0	634.0	0.0	2	0.0	0.0	0	
439	888.0	756.0	0.0	3	386.0	0.0	0	
1033	1200.0	0.0	0.0	1	661.0	0.0	1	
331	616.0	495.0	0.0	3	236.0	380.0	0	
527	1392.0	0.0	0.0	3	1302.0	0.0	1	
462	616.0	688.0	0.0	3	0.0	0.0	0	
861	1105.0	1169.0	0.0	5	443.0	0.0	0	
630	1208.0	0.0	0.0	3	767.0	0.0	1	
135	970.0	739.0	0.0	3	0.0	0.0	0	
358	1026.0	665.0	0.0	3	218.0	0.0	0	
363	1269.0	0.0	0.0	2	24.0	0.0	0	
618	1142.0	793.0	0.0	3	0.0	0.0	0	
561	684.0	684.0	0.0	3	0.0	0.0	0	
529	1163.0	511.0	0.0	4	0.0	0.0	0	
567	927.0	988.0	0.0	3	789.0	0.0	1	
158	1064.0	703.0	0.0	2	495.0	215.0	1	

345 rows × 36 columns

In [27]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train)
```

Out[27]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [28]:
```
y_pred=lr.predict(X_test)
y_pred
```

```
       [123305.65719311],
       [200341.24016791],
       [377957.32307114],
       [146941.7171239 ],
       [283693.48760668],
       [106694.19786718],
       [221733.02172508],
       [ 87733.47014004],
       [295005.66546305],
       [125250.62019285],
       [ 53569.11256022],
       [133831.0508612 ],

       [170901.4292971 ],
       [213273.17757301],
       [110818.14723232],
       [103893.81849706],
       [200665.21921033],
       [137764.84464902],
       [311840.64900748],
```

In [29]:
```
from sklearn.metrics import mean_squared_error
```

In [30]:
```
mse_In=mean_squared_error(y_test,y_pred)
mse_In
```

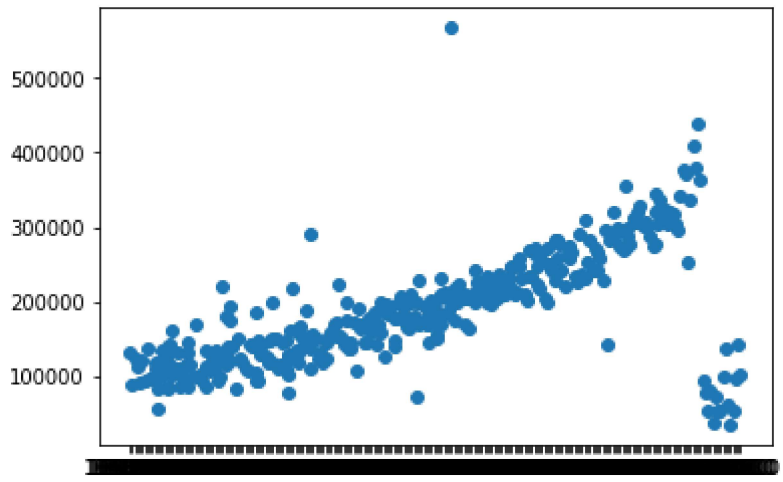
Out[30]:
```
1474827325.5975406
```

Step 3 : Create Scatter Plot

In [31]:
```
import matplotlib.pyplot as plt
```

In [32]:
```
plt.scatter(y_test,y_pred)
```

Out[32]:
```
<matplotlib.collections.PathCollection at 0x1e92734c358>
```



Step 4 : Encode Categorical Columns

In [33]:
```
gd=pd.get_dummies(df)
gd
```

Out[33]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath
0	856.0	854.0	0.0	3	706.0	0.0	1
1	1262.0	0.0	0.0	3	978.0	0.0	0
2	920.0	866.0	0.0	3	486.0	0.0	1
3	961.0	756.0	0.0	3	216.0	0.0	1
4	1145.0	1053.0	0.0	4	655.0	0.0	1
5	796.0	566.0	320.0	1	732.0	0.0	1
6	1694.0	0.0	0.0	3	1369.0	0.0	1
7	1107.0	983.0	0.0	3	859.0	32.0	1
8	1022.0	752.0	0.0	2	0.0	0.0	0
9	1077.0	0.0	0.0	2	851.0	0.0	1
10	1040.0	0.0	0.0	3	906.0	0.0	1
11	1182.0	1142.0	0.0	4	998.0	0.0	1
12	912.0	0.0	0.0	2	737.0	0.0	1
13	1494.0	0.0	0.0	3	0.0	0.0	0
14	1253.0	0.0	0.0	2	733.0	0.0	1
15	854.0	0.0	0.0	2	0.0	0.0	0
16	1004.0	0.0	0.0	2	578.0	0.0	1
17	1296.0	0.0	0.0	2	0.0	0.0	0
18	1114.0	0.0	0.0	3	646.0	0.0	1
19	1339.0	0.0	0.0	3	504.0	0.0	0
20	1158.0	1218.0	0.0	4	0.0	0.0	0
21	1108.0	0.0	0.0	3	0.0	0.0	0
22	1795.0	0.0	0.0	3	0.0	0.0	0
23	1060.0	0.0	0.0	3	840.0	0.0	1
24	1060.0	0.0	0.0	3	188.0	668.0	1
25	1600.0	0.0	0.0	3	0.0	0.0	0
26	900.0	0.0	0.0	3	234.0	486.0	0
27	1704.0	0.0	0.0	3	1218.0	0.0	1
28	1600.0	0.0	0.0	2	1277.0	0.0	1
29	520.0	0.0	0.0	1	0.0	0.0	0
...	...	...	...	...	...	...	...
1349	1048.0	510.0	0.0	3	580.0	0.0	1
1350	804.0	0.0	0.0	2	510.0	0.0	1

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath
1351	1440.0	0.0	0.0	3	678.0	0.0	0
1352	734.0	1104.0	0.0	4	0.0	0.0	0
1353	958.0	0.0	0.0	2	958.0	0.0	0
1354	968.0	0.0	0.0	4	0.0	0.0	0
1355	962.0	830.0	0.0	3	0.0	0.0	1
1356	1126.0	0.0	0.0	3	936.0	0.0	1
1357	1537.0	0.0	0.0	3	0.0	0.0	1
1358	864.0	0.0	0.0	3	616.0	0.0	0
1359	1932.0	0.0	304.0	2	1336.0	0.0	1
1360	1236.0	0.0	0.0	2	600.0	0.0	1
1361	1040.0	685.0	0.0	3	315.0	110.0	0
1362	1423.0	748.0	0.0	3	0.0	0.0	0
1363	848.0	0.0	0.0	1	697.0	0.0	1
1364	1026.0	981.0	0.0	3	765.0	0.0	1
1365	952.0	0.0	0.0	2	0.0	0.0	0
1366	1422.0	0.0	0.0	3	0.0	0.0	0
1367	913.0	0.0	0.0	3	187.0	627.0	1
1368	1188.0	0.0	0.0	3	593.0	0.0	0
1369	1220.0	870.0	0.0	3	1079.0	0.0	1
1370	796.0	550.0	0.0	2	0.0	0.0	0
1371	1578.0	0.0	0.0	3	0.0	0.0	0
1372	1072.0	0.0	0.0	2	547.0	0.0	1
1373	1221.0	0.0	0.0	2	410.0	0.0	1
1374	953.0	694.0	0.0	3	0.0	0.0	0
1375	2073.0	0.0	0.0	3	790.0	163.0	1
1376	1188.0	1152.0	0.0	4	275.0	0.0	0
1377	1078.0	0.0	0.0	2	49.0	1029.0	1
1378	1256.0	0.0	0.0	3	830.0	290.0	1

1379 rows × 39 columns

Step 5 : Predict Sale Price with Categorical features

In [34]:
```
x=gd.drop("SalePrice",axis=1)
x
```

Out[34]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
0	856.0	854.0	0.0	3	706.0	0.0	1	
1	1262.0	0.0	0.0	3	978.0	0.0	0	
2	920.0	866.0	0.0	3	486.0	0.0	1	
3	961.0	756.0	0.0	3	216.0	0.0	1	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	
5	796.0	566.0	320.0	1	732.0	0.0	1	
6	1694.0	0.0	0.0	3	1369.0	0.0	1	
7	1107.0	983.0	0.0	3	859.0	32.0	1	
8	1022.0	752.0	0.0	2	0.0	0.0	0	
9	1077.0	0.0	0.0	2	851.0	0.0	1	
10	1040.0	0.0	0.0	3	906.0	0.0	1	
11	1182.0	1142.0	0.0	4	998.0	0.0	1	
12	912.0	0.0	0.0	2	737.0	0.0	1	
13	1494.0	0.0	0.0	3	0.0	0.0	0	
14	1253.0	0.0	0.0	2	733.0	0.0	1	
15	854.0	0.0	0.0	2	0.0	0.0	0	
16	1004.0	0.0	0.0	2	578.0	0.0	1	
17	1296.0	0.0	0.0	2	0.0	0.0	0	
18	1114.0	0.0	0.0	3	646.0	0.0	1	
19	1339.0	0.0	0.0	3	504.0	0.0	0	
20	1158.0	1218.0	0.0	4	0.0	0.0	0	
21	1108.0	0.0	0.0	3	0.0	0.0	0	
22	1795.0	0.0	0.0	3	0.0	0.0	0	
23	1060.0	0.0	0.0	3	840.0	0.0	1	
24	1060.0	0.0	0.0	3	188.0	668.0	1	
25	1600.0	0.0	0.0	3	0.0	0.0	0	
26	900.0	0.0	0.0	3	234.0	486.0	0	
27	1704.0	0.0	0.0	3	1218.0	0.0	1	
28	1600.0	0.0	0.0	2	1277.0	0.0	1	
29	520.0	0.0	0.0	1	0.0	0.0	0	
...	...	...	...	...	...	...	...	
1349	1048.0	510.0	0.0	3	580.0	0.0	1	
1350	804.0	0.0	0.0	2	510.0	0.0	1	

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	Bs
1351	1440.0	0.0	0.0	3	678.0	0.0	0	
1352	734.0	1104.0	0.0	4	0.0	0.0	0	
1353	958.0	0.0	0.0	2	958.0	0.0	0	
1354	968.0	0.0	0.0	4	0.0	0.0	0	
1355	962.0	830.0	0.0	3	0.0	0.0	1	
1356	1126.0	0.0	0.0	3	936.0	0.0	1	
1357	1537.0	0.0	0.0	3	0.0	0.0	1	
1358	864.0	0.0	0.0	3	616.0	0.0	0	
1359	1932.0	0.0	304.0	2	1336.0	0.0	1	
1360	1236.0	0.0	0.0	2	600.0	0.0	1	
1361	1040.0	685.0	0.0	3	315.0	110.0	0	
1362	1423.0	748.0	0.0	3	0.0	0.0	0	
1363	848.0	0.0	0.0	1	697.0	0.0	1	
1364	1026.0	981.0	0.0	3	765.0	0.0	1	
1365	952.0	0.0	0.0	2	0.0	0.0	0	
1366	1422.0	0.0	0.0	3	0.0	0.0	0	
1367	913.0	0.0	0.0	3	187.0	627.0	1	
1368	1188.0	0.0	0.0	3	593.0	0.0	0	
1369	1220.0	870.0	0.0	3	1079.0	0.0	1	
1370	796.0	550.0	0.0	2	0.0	0.0	0	
1371	1578.0	0.0	0.0	3	0.0	0.0	0	
1372	1072.0	0.0	0.0	2	547.0	0.0	1	
1373	1221.0	0.0	0.0	2	410.0	0.0	1	
1374	953.0	694.0	0.0	3	0.0	0.0	0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	

1379 rows × 38 columns

In [35]:
```
Y=gd.pop('SalePrice')
```

In [36]:
```
Y
```

Out[36]:
```
0       208500.0
1       181500.0
2       223500.0
3       140000.0
4       250000.0
5       143000.0
6       307000.0
7       200000.0
8       129900.0
9       118000.0
10      129500.0
11      345000.0
12      144000.0
13      279500.0
14      157000.0
15      132000.0
16      149000.0
17       90000.0
18      159000.0
19      139000.0
20      325300.0
21      139400.0
22      230000.0
23      129900.0
24      154000.0
25      256300.0
26      134800.0
27      306000.0
28      207500.0
29       68500.0
          ...
1349    140000.0
1350    119000.0
1351    182900.0
1352    192140.0
1353    143750.0
1354     64500.0
1355    186500.0
1356    160000.0
1357    174000.0
1358    120500.0
1359    394617.0
1360    149700.0
1361    197000.0
1362    191000.0
1363    149300.0
1364    310000.0
1365    121000.0
1366    179600.0
1367    129000.0
1368    157900.0
1369    240000.0
1370    112000.0
1371    287090.0
1372    145000.0
```

```
1373    185000.0
1374    175000.0
1375    210000.0
1376    266500.0
1377    142125.0
1378    147500.0
Name: SalePrice, Length: 1379, dtype: float64
```

In [78]:
```
from sklearn.model_selection import train_test_split
```

In [79]:
```
X_train,X_test,y_train,y_test=train_test_split(x,Y,test_size=0.25,random_state=42
```

In [80]:
```
X_train
```

Out[80]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath
1193	1337.0	0.0	0.0	2	266.0	0.0	1
910	1800.0	0.0	0.0	2	0.0	0.0	0
1068	1328.0	653.0	0.0	4	622.0	0.0	1
1196	2018.0	0.0	0.0	3	0.0	0.0	0
1102	959.0	712.0	0.0	3	786.0	0.0	1
447	970.0	0.0	0.0	2	630.0	0.0	1
796	1701.0	0.0	0.0	3	1390.0	0.0	1
543	1320.0	0.0	0.0	3	328.0	551.0	1
901	768.0	0.0	0.0	2	660.0	0.0	0
968	1264.0	0.0	0.0	3	697.0	0.0	1
411	904.0	0.0	0.0	2	0.0	0.0	0
96	1535.0	0.0	0.0	4	0.0	0.0	0
429	624.0	720.0	0.0	4	0.0	0.0	0
361	784.0	0.0	0.0	2	0.0	0.0	0
933	778.0	798.0	0.0	3	0.0	0.0	0
588	1422.0	0.0	0.0	3	0.0	0.0	0
156	854.0	0.0	0.0	2	360.0	0.0	0
528	1389.0	0.0	0.0	2	1071.0	123.0	1
654	616.0	0.0	0.0	2	616.0	0.0	0
857	1636.0	0.0	0.0	3	63.0	0.0	1
1237	1294.0	0.0	0.0	3	1200.0	0.0	1
534	1496.0	636.0	0.0	1	1441.0	0.0	1
1078	1466.0	1362.0	0.0	4	1150.0	0.0	1
1190	1050.0	0.0	0.0	2	504.0	0.0	0
1312	869.0	349.0	0.0	3	375.0	0.0	0
371	1144.0	0.0	0.0	3	739.0	0.0	1
677	848.0	0.0	0.0	1	662.0	0.0	1
308	596.0	596.0	0.0	3	0.0	0.0	0
1371	1578.0	0.0	0.0	3	0.0	0.0	0
710	866.0	902.0	0.0	3	20.0	0.0	0
...	...	...	...	...	...	...	...
474	1801.0	0.0	0.0	1	1247.0	0.0	1
856	1063.0	0.0	0.0	3	354.0	290.0	1
747	1086.0	809.0	0.0	3	0.0	0.0	0

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath
252	1095.0	844.0	0.0	3	0.0	0.0	0
21	1108.0	0.0	0.0	3	0.0	0.0	0
1337	1569.0	0.0	0.0	1	988.0	0.0	0
459	1484.0	0.0	0.0	3	998.0	0.0	0
1184	760.0	896.0	0.0	3	0.0	0.0	0
276	910.0	648.0	0.0	4	420.0	0.0	0
955	1022.0	0.0	0.0	2	247.0	465.0	1
1215	1582.0	0.0	0.0	4	812.0	0.0	0
385	1050.0	1028.0	0.0	3	789.0	0.0	1
805	1779.0	0.0	0.0	3	306.0	1085.0	1
343	790.0	784.0	0.0	3	712.0	0.0	1
769	1008.0	0.0	0.0	2	486.0	0.0	0
1332	944.0	896.0	0.0	3	666.0	0.0	1
130	928.0	836.0	0.0	3	821.0	0.0	1
871	936.0	785.0	0.0	3	814.0	0.0	0
1123	1622.0	0.0	0.0	3	1159.0	0.0	1
87	964.0	0.0	0.0	2	713.0	0.0	1
330	1453.0	0.0	0.0	2	1082.0	0.0	1
1238	1902.0	0.0	0.0	3	1406.0	0.0	1
466	886.0	0.0	0.0	2	0.0	0.0	0
121	1216.0	941.0	0.0	4	445.0	0.0	0
1044	1500.0	1122.0	0.0	3	1032.0	0.0	1
1095	855.0	601.0	0.0	3	311.0	0.0	0
1130	815.0	875.0	0.0	3	0.0	0.0	0
1294	1661.0	0.0	0.0	3	831.0	0.0	1
860	742.0	742.0	0.0	3	0.0	0.0	0
1126	1224.0	0.0	0.0	2	883.0	0.0	1

1034 rows × 38 columns

In [81]:
```
from sklearn.linear_model import LinearRegression
lrr=LinearRegression()
lrr.fit(X_train,y_train)
```

Out[81]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [82]:
```
y_pred=lrr.predict(X_test)
y_pred
```

Out[82]:
```
array([257335.7000608 , 110874.86923391, 102030.23199612, 203810.90312835,
       207483.2480659 ,  42212.7574326 , 234280.44650339, 205672.50051956,
       186278.85766228, 234095.76559427, 100749.55432836, 303151.19469182,
       101416.85324779, 289134.34874561, 204622.14962899, 144125.59534156,
       248672.44116553, 151327.54761829, 209923.69905589, 278332.41533913,
        92437.05947657, 153368.70559331, 163347.7261726 , 241763.42050308,
       372545.10155691, 220606.10539607, 118571.34158749,  99792.7072199 ,
       319574.68090212, 172462.69803728, 258594.10016255, 199314.67396265,
       160816.17421074, 142048.73733347, 204308.92322938, 379621.05611974,
       122596.82381481, 140080.77925227, 274173.50644988, 212534.56183001,
       169526.18279983, 123222.23772216, 199622.34002294, 377498.88115512,
       147008.64960397, 284099.6780763 , 106018.68433928, 222067.43202873,
        87265.63294038, 295295.75316694, 125047.0784354 ,  57910.53539282,
       133650.09128778, 170939.04118439, 213586.29140634, 109341.23724215,
       103574.73601321, 200417.51505359, 137888.98047553, 311955.8103948 ,
        50931.25239792, 184831.59705593, 153942.66047453, 299123.12761307,
        75076.47529428, 323944.61412586, 168632.67343763, 111726.63702113,
       200007.22568537, 205885.41544422, 129884.14537041, 271618.65324128,
       118475.14015599, 131418.42150466,  76868.17741699, 137365.5323129 ,
        40068.05247601, 162673.92013052, 272441.29797779, 114664.49901778,
       320087.30946353, 218755.44030883, 292125.72890611, 211792.16962337,
        92967.77080722, 262583.93655816, 141983.16409308, 117600.86362534,
       143986.33531252, 278655.5474147 , 265530.94077979, 300716.04326903,
       182888.73508398, 167585.307986  , 146971.48817186, 214038.71837231,
       247638.35320219, 222020.78721895, 274589.80661869, 243256.83404347,
       110257.15897263,  99138.74840333, 193243.36415994, 205971.65829986,
       154014.47166577, 216779.85757434, 210878.7555899 , 125908.52099319,
       171006.95358214, 223485.47077546, 155128.87061852, 110624.75682335,
       316947.8201294 , 298288.41865442, 335834.25633813,  85600.23905958,
       146636.73217727, 228761.74907502, 123944.08245288, 204942.54760062,
       217379.50043091, 123762.90277275, 133672.09036587, 222836.20302007,
       138297.84732162, 210669.13922565, 145050.50648765, 291274.12940229,
       183335.25499099,  99131.25787205, 323949.3516407 , 169815.46232986,
       122859.66680039, 133635.25866724, 309288.0746345 , 136427.39180403,
       282054.59591017, 132357.92824253,  95782.40952431, 163543.36969658,
       147333.58567345, 161930.11711144, 178056.2451805 , 254021.77883811,
       118317.1863203 , 140147.24404783, 236480.59822887, 318539.83302864,
       104929.12688967, 192416.54111092, 173337.92698551, 146226.00369995,
        95590.19036005, 250935.38499394, 304466.56126348,  54155.91750029,
       281934.30483635,  92487.50385048, 137721.08954024, 166802.82729637,
       209882.85943555, 204879.81358078, 176968.7488887 , 248831.37001694,
       144271.26919536, 133014.99640096, 171729.75486892, 344889.13422791,
       185807.61187329,  98612.6640654 , 130562.05373372, 110800.35081728,
       131463.68500163, 193902.26395394, 156925.36923765, 268362.43407674,
       211633.97348416, 151833.75936011, 115207.8500842 , 243561.57215852,
       136980.78209122, 265194.67864609, 302373.36601291,  83731.86747192,
       145987.11111235, 150210.36085536, 157955.98265923, 232738.29150477,
       231619.37907519, 138762.51038792, 354958.59991748, 126835.48148651,
       235624.48563074, 116911.46008256, 102267.54537677, 159295.9267011 ,
       148763.87849564, 409048.91224914, 335794.92992339, 290627.28539879,
       281669.39020495, 275809.63128087, 321782.968471  , 307506.50021844,
       201780.16496724, 140396.39503433, 157398.91025594, 258932.34870213,
       207947.58817439, 146052.76342727, 106164.11806731, 148850.03768317,
       105470.13175352, 297401.51893045, 341465.92561059, 255183.45571356,
```

```
146693.79439062, 198929.65387489, 127613.99875597, 264183.77661559,
120875.34401829, 133746.45035929, 308189.2903134 , 169768.80931141,
173759.95936348, 566893.63974757, 182391.6408597 , 144318.52637991,
198879.21357919, 113102.45671667, 181317.70570622, 328601.18681405,
138928.93055974, 197725.53743694, 116408.33631823,  59496.61434674,
199370.52983974, 273178.0107013 , 119160.4920924 , 196622.81109287,
231850.60506868, 119330.11057542, 128673.89814385, 288886.86634337,
200492.85292132, 364165.59124924, 119394.41730747, 123962.17746738,
225584.31291989, 171262.27478797,  92436.09527977, 178476.74397583,
215908.76086063, 225752.02723759, 137034.31840065,  71728.53466883,
242892.98566822, 140597.66673376, 114299.42184527,  86823.13212311,
208248.80398969, 173073.58876992, 271073.81878313, 254165.32576792,
103156.72088931, 220718.44240527, 184489.21945713, 169603.94267526,
155042.76100916, 224852.88526431,  85446.00693613, 150018.35377773,
180589.26981649, 200150.19648482, 180047.76937309, 223497.17824985,
 74167.89083354,  85514.20006451, 108864.57795317, 233170.15935959,
152568.03837914, 320811.34056259, 194942.81676947,  56377.50662304,
204309.1305674 , 108545.58911279, 194861.13833177, 104106.84474521,
230056.61174817, 210941.90990009, 198341.21165699, 109515.09806044,
 95171.55007366, 235616.03943939, 185015.36886542, 189118.16026642,
265289.8748012 , 245979.66404826, 106185.14323225, 163736.51226748,
268132.69533488, 115413.08083202, 222462.07696328, 101986.48913422,
226503.80621331, 155293.88748451, 111481.20892496, 219053.45748964,
 80666.99094144,  84534.23575514,  97067.58074313, 304358.36736192,
143618.6767515 , 194838.68012385, 241402.30035471,  65336.49507497,
143451.99793023, 109929.52816359, 438435.09082674, 123040.05819342,
115632.92703816, 278523.10699513, 201672.73661732, 237921.46254273,
206051.72233254,  98153.64445104, 114462.18412199, 209364.03050815,
172231.0782842 , 161996.76684104, 151092.09934419, 189895.84501914,
101797.1978762 , 168524.98406397,  83142.15433374, 169792.18839573,
177211.98667652, 175314.84698001, 130024.25037615, 204638.5520429 ,
233310.77718692, 129809.59532018, 174325.73822936, 258642.31033449,
229318.17141995])
```

In [83]:
```
from sklearn.metrics import mean_squared_error
```

In [84]:
```
MSE=mean_squared_error(y_test,y_pred)
MSE
```

Out[84]:
```
1472346697.5193377
```

Step 6 : Normalize using StandardScaler and Predict Sale Price

In [85]:
```
from sklearn.preprocessing import StandardScaler
```

In [86]:
```
scaler=StandardScaler()
```

In [87]:
```
scaled_X_train=scaler.fit_transform(X_train)
scaled_X_train
```

Out[87]:
```
array([[ 0.39851037, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [ 1.57467708, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [ 0.37564751,  0.70143387, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       ...,
       [ 1.22157303, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-1.11297817,  0.90510323, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [ 0.11145456, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619]])
```

In [88]:
```
scaled_X_train.shape
```

Out[88]:
```
(1034, 38)
```

In [89]:
```
scaled_X_test=scaler.transform(X_test)
scaled_X_test
```

Out[89]:
```
array([[ 0.85830772, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-0.64810018, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [-0.21624632,  0.76093278, -0.11340519, ..., -1.39336465,
         4.39027446, -4.39027446],
       ...,
       [-0.04350477,  0.37647826, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-0.64301955,  1.46805451, -0.11340519, ..., -1.39336465,
        -0.22777619,  0.22777619],
       [-0.29499614,  0.81585486, -0.11340519, ..., -1.39336465,
        -0.22777619,  0.22777619]])
```

In [90]:
```
from sklearn.metrics import mean_squared_error
```

In [91]:
```
Scaler=StandardScaler()
scaled_X_train=Scaler.fit_transform(X_train)
scaled_X_train
```

Out[91]:
```
array([[ 0.39851037, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [ 1.57467708, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [ 0.37564751,  0.70143387, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       ...,
       [ 1.22157303, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-1.11297817,  0.90510323, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [ 0.11145456, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619]])
```

In [92]:
```
scaled_X_test=Scaler.transform(X_test)
scaled_X_test
```

Out[92]:
```
array([[ 0.85830772, -0.79290427, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-0.64810018, -0.79290427, -0.11340519, ...,  0.8683921 ,
        -0.22777619,  0.22777619],
       [-0.21624632,  0.76093278, -0.11340519, ..., -1.39336465,
         4.39027446, -4.39027446],
       ...,
       [-0.04350477,  0.37647826, -0.11340519, ...,  0.11447318,
        -0.22777619,  0.22777619],
       [-0.64301955,  1.46805451, -0.11340519, ..., -1.39336465,
        -0.22777619,  0.22777619],
       [-0.29499614,  0.81585486, -0.11340519, ..., -1.39336465,
        -0.22777619,  0.22777619]])
```

In [93]:
```
model1=LinearRegression()
model1.fit(scaled_X_train,y_train)
sy_pred=model1.predict(scaled_X_test)
sy_pred
```

Out[93]:
```
array([257335.70006079, 110874.86923392, 102030.23199608, 203810.90312837,
       207483.24806589,  42212.75743268, 234280.44650336, 205672.50051949,
       186278.85766227, 234095.76559429, 100749.55432845, 303151.19469193,
       101416.8532478 , 289134.34874559, 204622.14962902, 144125.5953416 ,
       248672.44116556, 151327.54761831, 209923.69905592, 278332.41533923,
        92437.05947649, 153368.70559335, 163347.7261726 , 241763.42050309,
       372545.10155694, 220606.1053961 , 118571.34158743,  99792.70721984,
       319574.68090213, 172462.69803731, 258594.10016254, 199314.67396263,
       160816.17421064, 142048.7373335 , 204308.92322941, 379621.05611975,
       122596.82381485, 140080.77925232, 274173.50644996, 212534.56183008,
       169526.18279981, 123222.23772213, 199622.34002308, 377498.88115508,
       147008.64960396, 284099.6780763 , 106018.68433925, 222067.43202863,
        87265.63294041, 295295.75316692, 125047.07843544,  57910.53539287,
       133650.09128776, 170939.04118448, 213586.29140632, 109341.2372422 ,
       103574.7360132 , 200417.51505365, 137888.98047556, 311955.81039482,
        50931.25239784, 184831.59705591, 153942.66047449, 299123.12761303,
        75076.47529434, 323944.61412584, 168632.67343754, 111726.6370212 ,
       200007.22568541, 205885.41544417, 129884.14537043, 271618.65324131,
       118475.14015603, 131418.4215046 ,  76868.17741696, 137365.53231295,
        40068.05247593, 162673.92013056, 272441.29797782, 114664.49901777,
       320087.30946347, 218755.44030861, 292125.72890613, 211792.16962336,
        92967.77080716, 262583.93655815, 141983.16409304, 117600.86362539,
       143986.33531252, 278655.54741473, 265530.94077975, 300716.04326905,
       182888.73508399, 167585.3079858 , 146971.48817191, 214038.71837227,
       247638.35320219, 222020.78721891, 274589.80661878, 243256.83404344,
       110257.1589726 ,  99138.7484034 , 193243.36415995, 205971.65829986,
       154014.47166577, 216779.85757428, 210878.75558992, 125908.52099322,
       171006.95358224, 223485.47077536, 155128.87061857, 110624.75682333,
       316947.8201294 , 298288.41865442, 335834.25633816,  85600.2390596 ,
       146636.73217728, 228761.74907496, 123944.08245283, 204942.54760066,
       217379.50043094, 123762.90277267, 133672.0903659 , 222836.20302013,
       138297.84732161, 210669.13922564, 145050.50648768, 291274.12940233,
       183335.25499099,  99131.25787201, 323949.35164069, 169815.46232989,
       122859.66680041, 133635.25866721, 309288.07463452, 136427.391804  ,
       282054.5959102 , 132357.92824255,  95782.40952433, 163543.36969663,
       147333.58567343, 161930.11711144, 178056.24518048, 254021.77883814,
       118317.18632029, 140147.24404785, 236480.59822898, 318539.83302871,
       104929.12688968, 192416.54111088, 173337.92698564, 146226.00369988,
        95590.19036001, 250935.38499392, 304466.56126343,  54155.91750032,
       281934.30483638,  92487.50385053, 137721.08954028, 166802.82729636,
       209882.85943558, 204879.81358066, 176968.7488887 , 248831.37001693,
       144271.26919537, 133014.99640085, 171729.75486891, 344889.13422787,
       185807.61187335,  98612.66406539, 130562.05373374, 110800.35081734,
       131463.6850017 , 193902.26395398, 156925.36923761, 268362.43407685,
       211633.9734842 , 151833.75936011, 115207.85008425, 243561.57215856,
       136980.78209138, 265194.6786461 , 302373.3660129 ,  83731.86747195,
       145987.11111225, 150210.36085541, 157955.98265927, 232738.29150469,
       231619.37907517, 138762.51038796, 354958.5999176 , 126835.4814865 ,
       235624.48563076, 116911.46008251, 102267.54537676, 159295.92670103,
       148763.87849563, 409048.91224919, 335794.92992335, 290627.2853988 ,
       281669.39020499, 275809.6312809 , 321782.96847099, 307506.50021851,
       201780.16496717, 140396.39503433, 157398.910256  , 258932.34870208,
```

```
207947.58817436, 146052.76342721, 106164.1180673 , 148850.03768324,
105470.13175355, 297401.51893047, 341465.92561058, 255183.45571352,
146693.7943907 , 198929.65387474, 127613.998756  , 264183.77661557,
120875.34401832, 133746.45035928, 308189.29031338, 169768.80931128,
173759.95936344, 566893.63974758, 182391.64085972, 144318.5263799 ,
198879.21357916, 113102.45671674, 181317.70570623, 328601.18681399,
138928.93055976, 197725.53743697, 116408.3363182 ,  59496.61434679,
199370.52983972, 273178.01070135, 119160.49209244, 196622.8110928 ,
231850.60506867, 119330.11057544, 128673.89814385, 288886.86634336,
200492.85292127, 364165.59124924, 119394.4173075 , 123962.17746744,
225584.31291968, 171262.27478796,  92436.09527978, 178476.7439758 ,
215908.76086059, 225752.02723758, 137034.31840071,  71728.53466881,
242892.98566818, 140597.66673379, 114299.42184529,  86823.13212314,
208248.80398971, 173073.58876988, 271073.81878316, 254165.32576783,
103156.72088932, 220718.44240536, 184489.2194571 , 169603.94267526,
155042.76100906, 224852.8852643 ,  85446.0069361 , 150018.35377784,
180589.26981653, 200150.19648481, 180047.76937302, 223497.17824987,
 74167.89083358,  85514.2000643 , 108864.57795327, 233170.15935943,
152568.03837919, 320811.34056264, 194942.81676943,  56377.50662303,
204309.13056739, 108545.58911283, 194861.13833172, 104106.84474528,
230056.61174825, 210941.90990015, 198341.21165698, 109515.09806039,
 95171.55007358, 235616.03943937, 185015.36886539, 189118.16026643,
265289.87480122, 245979.66404827, 106185.14323229, 163736.5122675 ,
268132.6953349 , 115413.08083206, 222462.07696335, 101986.48913419,
226503.80621333, 155293.88748457, 111481.20892499, 219053.45748969,
 80666.99094146,  84534.2357552 ,  97067.58074317, 304358.36736191,
143618.67675154, 194838.68012388, 241402.30035477,  65336.49507492,
143451.99793027, 109929.52816361, 438435.09082657, 123040.0581934 ,
115632.92703816, 278523.10699502, 201672.73661731, 237921.46254269,
206051.72233251,  98153.6444511 , 114462.18412202, 209364.03050816,
172231.07828409, 161996.76684115, 151092.09934421, 189895.84501926,
101797.1978762 , 168524.98406397,  83142.15433382, 169792.18839557,
177211.98667652, 175314.84698005, 130024.25037621, 204638.5520429 ,
233310.77718695, 129809.59532021, 174325.73822928, 258642.31033451,
229318.17142004])
```

In [94]:
```
mse_In1=mean_squared_error(y_test,sy_pred)
mse_In1
print("mean square error using standard scalar:",mse_In1)
```

```
mean square error using standard scalar: 1472346697.5197492
```

Step 7 : Normalize using MinMaxscaler and Predict Sale Price

In [95]:
```
from sklearn.preprocessing import MinMaxScaler
mm_scaler=MinMaxScaler()
```

In [96]:
```
mmX_train=mm_scaler.fit_transform(X_train)
mmX_train
```

Out[96]:
```
array([[0.21133051, 0.        , 0.        , ..., 0.5       , 0.        ,
        1.        ],
       [0.32016925, 0.        , 0.        , ..., 0.75      , 0.        ,
        1.        ],
       [0.20921486, 0.31622276, 0.        , ..., 0.75      , 0.        ,
        1.        ],
       ...,
       [0.28749412, 0.        , 0.        , ..., 0.5       , 0.        ,
        1.        ],
       [0.07146215, 0.35932203, 0.        , ..., 0.75      , 0.        ,
        1.        ],
       [0.18476728, 0.        , 0.        , ..., 0.75      , 0.        ,
        1.        ]])
```

In [97]:
```
mmX_test=mm_scaler.transform(X_test)
mmX_test
```

Out[97]:
```
array([[0.2538787 , 0.        , 0.        , ..., 0.5       , 0.        ,
        1.        ],
       [0.11448049, 0.        , 0.        , ..., 0.75      , 0.        ,
        1.        ],
       [0.15444288, 0.32881356, 0.        , ..., 0.        , 1.        ,
        0.        ],
       ...,
       [0.17042783, 0.24745763, 0.        , ..., 0.5       , 0.        ,
        1.        ],
       [0.11495063, 0.47845036, 0.        , ..., 0.        , 0.        ,
        1.        ],
       [0.14715562, 0.34043584, 0.        , ..., 0.        , 0.        ,
        1.        ]])
```

In [98]:
```
model3=LinearRegression()
model3.fit(mmX_train,y_train)
```

Out[98]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [99]:
```
mms_y_pred=model3.predict(mmX_test)
mms_y_pred
```

Out[99]:
```
array([257335.70006079, 110874.86923392, 102030.23199608, 203810.90312837,
       207483.24806589,  42212.75743267, 234280.44650336, 205672.50051949,
       186278.85766227, 234095.76559429, 100749.55432845, 303151.19469193,
       101416.8532478 , 289134.34874559, 204622.14962902, 144125.5953416 ,
       248672.44116556, 151327.54761831, 209923.69905592, 278332.41533923,
        92437.05947649, 153368.70559335, 163347.7261726 , 241763.42050309,
       372545.10155694, 220606.1053961 , 118571.34158743,  99792.70721984,
       319574.68090213, 172462.69803731, 258594.10016254, 199314.67396263,
       160816.17421064, 142048.7373335 , 204308.92322941, 379621.05611975,
       122596.82381485, 140080.77925232, 274173.50644997, 212534.56183008,
       169526.18279981, 123222.23772213, 199622.34002308, 377498.88115508,
       147008.64960396, 284099.6780763 , 106018.68433925, 222067.43202863,
        87265.63294041, 295295.75316692, 125047.07843544,  57910.53539287,
       133650.09128776, 170939.04118448, 213586.29140632, 109341.2372422 ,
       103574.7360132 , 200417.51505365, 137888.98047556, 311955.81039482,
        50931.25239784, 184831.59705591, 153942.66047449, 299123.12761303,
        75076.47529434, 323944.61412584, 168632.67343754, 111726.6370212 ,
       200007.22568541, 205885.41544417, 129884.14537043, 271618.65324131,
       118475.14015603, 131418.4215046 ,  76868.17741695, 137365.53231295,
        40068.05247593, 162673.92013056, 272441.29797782, 114664.49901777,
       320087.30946347, 218755.44030861, 292125.72890613, 211792.16962336,
        92967.77080716, 262583.93655815, 141983.16409304, 117600.86362539,
       143986.33531252, 278655.54741473, 265530.94077975, 300716.04326905,
       182888.73508399, 167585.3079858 , 146971.48817191, 214038.71837227,
       247638.35320219, 222020.78721891, 274589.80661878, 243256.83404344,
       110257.1589726 ,  99138.7484034 , 193243.36415995, 205971.65829986,
       154014.47166577, 216779.85757428, 210878.75558992, 125908.52099322,
       171006.95358224, 223485.47077536, 155128.87061857, 110624.75682333,
       316947.8201294 , 298288.41865443, 335834.25633817,  85600.2390596 ,
       146636.73217728, 228761.74907496, 123944.08245283, 204942.54760066,
       217379.50043094, 123762.90277267, 133672.0903659 , 222836.20302013,
       138297.84732161, 210669.13922564, 145050.50648768, 291274.12940233,
       183335.25499099,  99131.25787201, 323949.35164069, 169815.46232989,
       122859.66680041, 133635.25866721, 309288.07463452, 136427.391804  ,
       282054.5959102 , 132357.92824255,  95782.40952433, 163543.36969663,
       147333.58567343, 161930.11711144, 178056.24518048, 254021.77883814,
       118317.18632029, 140147.24404785, 236480.59822898, 318539.83302871,
       104929.12688968, 192416.54111088, 173337.92698564, 146226.00369988,
        95590.19036001, 250935.38499392, 304466.56126343,  54155.91750032,
       281934.30483638,  92487.50385053, 137721.08954028, 166802.82729636,
       209882.85943558, 204879.81358066, 176968.7488887 , 248831.37001693,
       144271.26919537, 133014.99640085, 171729.75486891, 344889.13422787,
       185807.61187335,  98612.66406539, 130562.05373374, 110800.35081734,
       131463.6850017 , 193902.26395398, 156925.36923761, 268362.43407685,
       211633.9734842 , 151833.75936011, 115207.85008425, 243561.57215856,
       136980.78209138, 265194.6786461 , 302373.3660129 ,  83731.86747195,
       145987.11111225, 150210.36085541, 157955.98265927, 232738.29150469,
       231619.37907517, 138762.51038796, 354958.5999176 , 126835.4814865 ,
       235624.48563076, 116911.46008251, 102267.54537676, 159295.92670103,
       148763.87849563, 409048.91224919, 335794.92992335, 290627.2853988 ,
       281669.39020499, 275809.6312809 , 321782.96847099, 307506.50021851,
       201780.16496717, 140396.39503433, 157398.910256  , 258932.34870208,
       207947.58817436, 146052.76342721, 106164.1180673 , 148850.03768324,
       105470.13175355, 297401.51893047, 341465.92561058, 255183.45571352,
```

```
146693.7943907 , 198929.65387474, 127613.998756  , 264183.77661557,
120875.34401832, 133746.45035928, 308189.29031338, 169768.80931128,
173759.95936344, 566893.63974758, 182391.64085972, 144318.5263799 ,
198879.21357916, 113102.45671674, 181317.70570623, 328601.18681399,
138928.93055976, 197725.53743697, 116408.3363182 ,  59496.61434679,
199370.52983972, 273178.01070135, 119160.49209244, 196622.8110928 ,
231850.60506867, 119330.11057544, 128673.89814385, 288886.86634336,
200492.85292127, 364165.59124924, 119394.4173075 , 123962.17746744,
225584.31291968, 171262.27478796,  92436.09527978, 178476.7439758 ,
215908.76086059, 225752.02723758, 137034.31840071,  71728.53466881,
242892.98566818, 140597.66673379, 114299.42184529,  86823.13212314,
208248.80398971, 173073.58876988, 271073.81878316, 254165.32576783,
103156.72088932, 220718.44240536, 184489.2194571 , 169603.94267526,
155042.76100906, 224852.8852643 ,  85446.0069361 , 150018.35377784,
180589.26981653, 200150.19648481, 180047.76937302, 223497.17824987,
 74167.89083358,  85514.2000643 , 108864.57795327, 233170.15935943,
152568.03837919, 320811.34056264, 194942.81676943,  56377.50662303,
204309.13056739, 108545.58911283, 194861.13833172, 104106.84474528,
230056.61174825, 210941.90990015, 198341.21165698, 109515.09806039,
 95171.55007358, 235616.03943937, 185015.36886539, 189118.16026643,
265289.87480122, 245979.66404827, 106185.14323229, 163736.5122675 ,
268132.6953349 , 115413.08083206, 222462.07696335, 101986.48913419,
226503.80621333, 155293.88748457, 111481.20892499, 219053.45748969,
 80666.99094146,  84534.2357552 ,  97067.58074317, 304358.36736191,
143618.67675154, 194838.68012388, 241402.30035477,  65336.49507492,
143451.99793027, 109929.52816361, 438435.09082657, 123040.0581934 ,
115632.92703816, 278523.10699502, 201672.73661731, 237921.4625427 ,
206051.72233251,  98153.6444511 , 114462.18412202, 209364.03050816,
172231.07828409, 161996.76684115, 151092.09934421, 189895.84501926,
101797.1978762 , 168524.98406397,  83142.15433382, 169792.18839557,
177211.98667652, 175314.84698005, 130024.25037621, 204638.5520429 ,
233310.77718695, 129809.59532021, 174325.73822928, 258642.31033451,
229318.17142004])
```

In [100]:
```
mmMSE=mean_squared_error(y_test,mms_y_pred)
```

In [101]:
```
print("mean sqaure error using min max scaler:",mmMSE)
```

```
mean sqaure error using min max scaler: 1472346697.519749
```

Step 8 : Predict using Lasso Regression

In [104]:
```
from sklearn.linear_model import SGDRegressor
sgd=SGDRegressor()
sgd.fit(scaled_X_train, y_train)
sgd_y_pred=sgd.predict(scaled_X_test)
print("Predictions of scaled data using SGDRegressor:", sgd_y_pred)
```

```
Predictions of scaled data using SGDRegressor: [263365.35676475 110472.78294028
95876.27171828 203974.65299836
 209358.69059155  41634.42228152 238261.20641505 205937.54296951
 180822.0017188  224895.32039446  98133.14795187 296984.63986687
 105853.15009882 293019.1978707  206157.33647709 155207.65976748
 238889.24508223 152462.90403327 214242.68660139 268670.63202172
  92755.55172444 145706.55301437 166627.67454333 249797.87528517
 369914.40894417 221883.03016119 123811.6175156  100960.01478062
 324964.72166266 171752.20700511 256071.58459344 199144.17542032
 144576.82004303 143483.97914726 209283.27076375 377618.33804858
 117559.10207849 114656.28446828 270844.24852999 210450.31496603
 167577.71271999 125009.57140052 192394.58843547 385094.80545207
 154515.29579896 283250.5846553  106555.25356223 227894.31655648
  90039.2400267  301294.06246645 125337.25200119  50473.71254554
 122278.91855161 161781.14847996 207369.51573045 111693.67281796
 109559.61651955 204609.96200753 145679.63270328 318522.39407964
  47158.97783845 187805.93233284 153664.10179105 293894.12217371
  76908.29885904 326895.43951541 174136.68721484 107578.84752459
 202477.73714702 206728.42188626 135406.93463428 270821.75197058
 118844.8840881  132373.92252386  77570.88353148 143417.51630154
  39816.23957305 165332.73588845 277462.09636184 104522.4359979
 327518.6721803  218780.10684834 297166.2845403  213046.22019565
  99234.64254944 256886.11085484 141547.26690375 106999.82272623
 141650.23872747 278763.52866878 273265.77724649 289888.45744354
 179671.80285118 169047.1264908  152440.65731301 212509.96835274
 246000.19220075 214565.05880265 277710.04068949 243269.27518905
 100570.34706098  91727.6258901  196724.96847428 197557.07945053
 157924.97870373 215356.95968727 214682.66451779 127522.36904422
 170556.94463777 234673.68198354 146738.07981893 107375.90831942
 325000.43675712 287991.02942765 330671.41818765  84855.73760608
 150324.0166086  228318.37159794 125601.22249353 201991.01898236
 216044.4632567  129710.82520195 129242.30965652 218714.1890243
 139876.45556606 206587.04897509 143508.36364154 288262.57489623
 180426.6181588   98466.16111487 314813.79508064 169121.00399634
 127560.15815203 137528.68626506 299882.39447055 147033.29445658
 274881.25558554 134327.36616261  91135.77767165 168847.2270202
 145911.79912183 161057.95179305 181653.53734784 251300.17255759
 119408.46090719 140232.78817498 237503.1866845  316126.38614168
 103068.01755082 192663.84643148 162636.4082364  150700.09204013
  93670.32308787 248157.23886911 316869.69557048  52351.40856769
 284557.52081116  92650.82545753 143341.60780999 180815.80287786
 214471.29523992 201697.01054072 177591.64582561 260781.67419722
 154100.36274292 130273.93171889 170898.6607347  340890.88031595
 174076.50503693  95324.81241141 128391.4488511  108074.6062479
 127545.26590551 183304.840784   156249.99989016 274018.97144689
 209187.64019423 158066.23544483 110909.39026919 246145.66476953
 128658.64703682 262097.83295063 307529.53860652  92614.72194789
 146084.09949825 145389.19195184 155762.65132217 242749.84405123
 233952.38137835 141505.68199243 352485.80899357 132100.1080409
 236759.94696976 124977.26667349 102052.49031427 166587.67191347
 155772.547369   397882.24670046 347024.16272331 285562.64781991
```

```
 284419.59851524 275506.95511746 311064.63819102 304255.26759923
 205517.79296243 139163.59530646 158255.98229372 333943.81809029
 206319.87277425 150388.80473815 107979.15672894 141874.95467978
 105774.8182564  296946.69062161 346707.65093528 253661.45508168
 146976.89123804 197211.82657386 134424.5344684  260507.76868596
 130718.24268947 138936.2116488  311254.44011331 176269.48793053
 172788.13337855 565837.41179017 189817.48252579 152901.63066961
 199402.4394753  112219.96122721 183160.21974868 336152.23125497
 139027.56800796 195949.30720769 127440.10109962  52128.28634247
 197134.62592758 267787.03553815 116590.52660967 205211.21856129
 237701.81168474 119806.57252668 133103.21461361 280486.22329507
 202674.49165017 364544.1258099  121047.70092397 123868.3518243
 222724.20019683 171010.0780351   86660.79239584 178578.13242843
 215234.90465329 225108.75824135 135921.69860408  81287.58334035
 241006.12226854 140117.41386855 118249.74613009  91302.90697504
 211986.35943271 166741.39135662 276375.86738397 258939.24345547
 102959.54269804 224712.99048329 188721.35313423 164411.94330467
 158190.68786489 229405.04055244  86685.09565727 142610.86664441
 179631.10849211 198165.69862069 178203.1203026  211318.89354143
 150430.89340981  92730.50536484 111688.53811247 236689.91699689
 155580.7830597  310978.01013572 193700.04206936  57048.01282953
 207260.75149131 109902.334381   199621.82052163 106399.31923674
 228825.84913528 212535.55200842 209386.95062757 108811.82201435
  90657.70708014 240560.76111263 183051.1030502  191503.39351271
 259611.43486605 239489.71455523 102806.631124   160997.41514391
 258504.90449118 108050.52088174 218007.58167506 107237.28492153
 231812.46256942 152134.39671113 117639.6256501  227643.12966613
  80100.52762495  85369.18351343  92079.02539459 311613.4326524
 227559.47334305 186619.10975758 228905.2384136   60248.42811371
 142886.30016602 117713.54250814 441774.93204206 118668.82829705
 119026.97260075 277057.75199955 197311.76844619 237398.01269459
 217842.53166957  93499.93518836 115104.943539   204740.1119514
 159980.64729063 158268.41297521 151722.3974905  193282.51194474
  93731.49921701 179140.97306036  76665.19574513 161389.3370699
 183723.88483149 163644.1548467  119332.38988495 205856.43937098
 219116.33122782 124377.18778504 180388.09759045 260873.01021934
 223455.51568232]


C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\linear_model\stochastic_gradient.py:128: FutureWarning: max_iter
and tol parameters have been added in <class 'sklearn.linear_model.stochastic_g
radient.SGDRegressor'> in 0.19. If both are left unset, they default to max_ite
r=5 and tol=None. If tol is not None, max_iter defaults to max_iter=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 [105]:
```
sgd_mse=mean_squared_error(y_test, sgd_y_pred)
print("SGD_MSE:",sgd_mse)
```

```
SGD_MSE: 1405689651.7813113
```

In [106]:
```
from sklearn.linear_model import Ridge
ridge=Ridge()
ridge.fit(scaled_X_train, y_train)
ridge_y_pred=ridge.predict(scaled_X_test)
print("Predictions of scaled data using RIDGERegression:", ridge_y_pred)
```

```
Predictions of scaled data using RIDGERegression: [257323.77267829 110840.58185
498 102155.66930557 203791.60276732
 207481.15953914  42284.4900783  234276.89999096 205674.13038532
 186260.97593155 233964.9162914  100738.55980766 302973.48735115
 101540.64446997 289153.92149887 204656.39043347 144238.21637309
 248574.1142511  151279.33720417 209958.04407444 278216.14584553
  92568.25328136 153392.28220542 163302.27037943 241695.88223351
 372483.18459847 220574.1767691  118680.4208957   99801.31026195
 319553.39497003 172399.13663266 258564.3421647  199211.35006526
 160692.59599362 141997.96839949 204332.0161651  379466.80851643
 122480.27009099 139897.20219397 274078.53686686 212489.29761581
 169548.59269522 123195.111445   199413.56880273 377169.5771548
 147135.48021148 284134.00225094 105982.81551348 222053.84895016
  87394.90217785 295275.94312014 124998.84034769  57836.50618635
 133670.13060888 170846.99075969 213568.09375127 109424.98671997
 103645.04294905 200458.8241374  137949.84466075 311916.1108146
  51019.79886591 184763.32083922 154007.5372499  299020.63855923
  75185.12973932 323917.54774981 168685.22322342 111839.5775294
 199995.60620669 205878.43413142 130023.84897587 271614.01357478
 118511.96246591 131568.18511618  76851.1105597  137269.17926824
  40080.30860524 162747.03669939 272420.18513097 114577.56000151
 320056.66230936 218777.94221065 292086.24021462 211836.89664984
  93133.6554028  262503.12118525 141903.82832695 117522.97407172
 143930.22989791 278653.69909114 265500.07722137 300644.69209269
 182925.43302621 167610.93812603 147129.9323156  214084.12145134
 247588.24822937 221979.55824904 274563.20829393 243195.11230737
 110227.72936546  99141.873292   193299.01598946 205904.77789463
 154055.92387921 216660.25439282 210871.69835458 125909.91451486
 171020.9849321  223585.25856341 155144.95094252 110720.87702199
 316875.46048853 298140.47544089 335711.10461261  85571.92273582
 146676.92743401 228741.87601761 124031.35735309 204933.07955419
 217352.31262648 123957.13801279 133684.67618064 222879.19557009
 138290.6676174  210664.03372361 144998.94250597 291223.02784802
 183379.20514398  99124.08155879 323811.43314455 169797.87673626
 122946.50958169 133689.96172537 309138.57664401 136587.08197363
 281993.11795458 132434.06466665  95745.77349337 163593.24516797
 147278.64543553 161902.43001749 178078.99347023 253955.76021186
 118434.04901004 140180.9966762  236413.95113177 318448.64990148
 105064.70328193 192470.83542124 173277.92739205 146290.35517831
  95589.47644158 250912.48817894 304526.29505592  54284.60570199
 281918.50733021  92562.01176665 137874.13805753 166969.88339967
 209921.19120929 204972.94350999 176997.58640453 248970.56820364
 144538.68959925 132953.26136022 171768.02898653 344730.30272587
 185678.17473398  98633.82366157 130560.93668174 110883.67995496
 131495.59633134 193811.19427853 156842.26041482 268274.99623483
 211606.07315426 151860.01101354 115149.73109365 243576.57559463
 136725.50426085 265232.85528904 302320.27264266  83937.76220554
 145912.3955047  150249.08717602 158004.10795697 232757.8413444
 231616.34731427 138755.75867933 354630.16612608 126859.02791216
 235634.12893341 116933.18534985 102235.91237775 159373.05502429
 148800.49877887 408860.29422474 335786.34679371 290617.23961556
```

```
281585.83608695 275744.54542145 321657.72637839 307452.5240736
201826.76304537 140387.64252338 157437.68202175 259182.11632274
207968.29567739 146121.01111302 106263.75820218 148817.50165207
105447.39563861 297435.59904016 341386.68024592 255181.37091215
146731.59582038 199047.61729653 127717.04569049 264181.03117738
120965.76141859 133833.68434958 308215.27035477 169739.61650079
173799.65092426 566549.88478386 182380.31530683 144386.82297672
198911.27309274 113105.17681954 181274.51066155 328545.37187515
138875.29859587 197740.8350548  116583.82312451  59442.73524024
199354.44546315 273169.56659975 119227.15827753 196692.94261598
231815.33840579 119428.80635188 128759.09417294 288791.68164933
200555.9628117  364221.77887793 119383.37291562 123931.23199097
225507.08692444 171178.85421318  92357.47714862 178519.93076411
215895.87651683 225786.499088   137015.38573291  72031.90285688
242928.07644377 140613.63860065 114293.3695774   87078.81472833
208141.82365383 172986.29951687 271091.88279273 254133.30021855
103183.58207872 220673.57480131 184644.16945381 169634.0119175
155094.27992532 224858.41991327  85495.07700873 149972.51784779
180529.26566444 200159.14726669 180068.51973279 223257.24293544
 74405.1954927   85811.8471605  108853.93217252 233077.07669542
152544.19590163 320715.56000836 194940.39495025  56449.62081146
204338.62761706 108605.73030514 194900.32416814 104228.137455
230066.34257392 210940.52584996 198414.25108027 109545.27521156
 95211.2817839  235649.54567795 185124.63097701 189167.87859731
265310.61016441 245865.1585356  106221.13485808 163481.45812301
268035.45006991 115340.70676809 222382.55934778 102098.49266624
226549.65497099 155245.29183201 111599.88664975 218964.42575605
 80721.62342876  84705.11015039  96986.39969326 304318.42747467
143683.45994018 194729.20470215 241376.94560427  65263.84133136
143480.94925133 110041.56059057 438272.20486575 123034.17538604
115639.84357146 278522.50016955 201675.278766   237953.78333025
206182.95457152  98157.2592199  114464.42438128 209350.27792438
172099.58983002 161885.89226037 151174.92361755 189929.90736369
101660.04563579 168642.55757554  83163.8269337  169741.36306165
177272.58816486 175238.85177353 129999.7575984  204641.93254342
233191.37393452 129811.14022437 174386.67716924 258593.01156231
229165.73496778]
```

In [107]:
```
ridge_mse=mean_squared_error(y_test, ridge_y_pred)
print("RIDGE_MSE:",ridge_mse)
```

```
RIDGE_MSE: 1470531869.0331736
```

In [108]:
```
from sklearn.linear_model import Lasso
lasso=Lasso()
lasso.fit(scaled_X_train, y_train)
lasso_y_pred=lasso.predict(scaled_X_test)
print("Predictions of scaled data using LASSORegression:", lasso_y_pred)
```

```
Predictions of scaled data using LASSORegression: [257322.53122417 110869.06313
38  102045.77117079 203807.30056776
 207488.37669922  42210.01239849 234276.00930725 205662.83980841
 186261.2322228  234105.51520377 100754.09105965 303156.95715008
 101419.74088453 289146.73359144 204613.89887649 144150.21910279
 248670.80009633 151322.15780201 209920.15893014 278329.30380941
  92443.25359299 153368.54211663 163338.39141343 241750.84468992
 372544.54909489 220614.6887412  118585.44514583  99798.63970882
 319577.23749025 172456.00822708 258599.04282245 199287.76908114
 160825.5198753  142058.49309218 204312.07338222 379608.26144047
 122588.73803864 140053.96565655 274174.58857198 212515.2067883
 169528.19329873 123218.8219451  199627.13856789 377475.42820338
 147012.76857603 284089.18084067 106012.31997877 222067.4899571
  87263.39898277 295296.59982809 125041.24800124  57912.60924175
 133650.56868096 170927.0738602  213588.26120924 109353.99423358
 103572.87138125 200421.69708594 137884.97828026 311953.93996412
  50941.78398234 184825.26449581 153945.85021257 299116.9849268
  75077.87918565 323942.66400627 168648.08844706 111729.04739963
 199996.93098867 205876.19776012 129910.70514524 271609.42776919
 118459.15510829 131455.14139686  76870.94905641 137347.33021907
  40074.48027934 162663.32345881 272443.74833735 114649.57237486
 320076.4338195  218764.68462001 292123.0019517  211799.27011979
  92972.99999891 262566.35634126 141980.81907718 117590.19747504
 143984.84525446 278646.52295048 265514.52811159 300717.22994438
 182891.40066542 167596.33572469 146975.73288957 214030.08046566
 247632.24371686 222020.62591425 274588.12864759 243253.83286089
 110241.00479106  99142.74065484 193242.03811281 205972.8491778
 154011.19711457 216764.37250751 210872.41337779 125906.98225537
 171013.32365687 223499.60247546 155129.03934872 110631.62841829
 316937.57871343 298282.72649966 335837.85806262  85590.95183573
 146636.07499676 228751.9571     123939.37199472 204932.68428131
 217365.18897197 123791.95989038 133664.49477513 222840.9930816
 138306.44967579 210658.18878006 145048.03464772 291267.82527114
 183339.54810943  99131.60245338 323932.49450752 169820.44487252
 122876.19539499 133636.76528532 309280.55272226 136446.97535968
 282052.0011861  132379.21688007  95761.35458253 163548.08908949
 147307.77291665 161923.77319914 178065.87680196 254018.48095132
 118318.9211046  140145.28389359 236465.6801953  318552.80489033
 104938.96855952 192410.03070146 173336.26429947 146223.6604548
  95598.53445574 250936.24906941 304478.35735512  54157.32622378
 281937.732851    92507.46818309 137720.64356059 166831.50048497
 209878.98180688 204863.70836519 176965.23086899 248832.63793291
 144292.03952788 133003.79943826 171729.10879095 344893.96574658
 185799.10082732  98612.32335901 130561.06241008 110797.27161324
 131446.60829762 193891.70108377 156918.84221545 268354.59817762
 211628.80171065 151823.74253949 115209.22854416 243561.11474585
 136954.82324733 265195.92773799 302380.87395353  83745.35871449
 145973.61156067 150224.88297682 157942.46435896 232738.60707347
 231616.83964039 138743.46725011 354938.21467452 126826.29909575
 235632.21294216 116927.19165697 102262.44235712 159301.4584731
 148765.34483942 409028.16848039 335790.64924151 290618.0685795
```

```
 281651.30590465 275811.36350304 321767.31742886 307507.28787301
 201775.98900283 140393.53502156 157400.72849401 258923.36452973
 207951.8066779  146057.42613606 106192.75206559 148816.75390782
 105465.12151559 297407.41778472 341466.24525625 255183.66188455
 146692.43746166 198950.05997407 127608.8722458  264192.51417373
 120873.79926871 133776.53056943 308191.56102191 169770.90075213
 173748.20728949 566891.36490744 182370.8807478  144328.45874971
 198897.3793836  113088.25819064 181316.45903484 328602.78303491
 138917.95565507 197716.73045505 116427.59268381  59468.51219378
 199368.60008543 273193.21859223 119150.55413065 196627.28005964
 231843.55515419 119329.15258797 128691.7284275  288885.31437199
 200491.67950358 364154.81499292 119403.38745245 123952.09853164
 225577.41243592 171252.63903816  92433.94634828 178479.25421827
 215903.67939729 225753.84905152 137033.11241307  71764.12603991
 242902.63717426 140601.31306152 114290.91560562  86842.9046608
 208246.66832416 173066.57928392 271061.5779007  254175.55848652
 103156.41611068 220722.05508398 184489.71379056 169610.86239935
 155038.85708566 224857.04132989  85452.43144599 150011.84404029
 180581.08143588 200155.33763262 180048.74353762 223473.55189948
  74175.58244728  85547.24800574 108854.99304907 233180.96884468
 152550.97669823 320803.19322525 194943.80306371  56384.46263869
 204310.3850739  108527.61386396 194868.71165288 104106.60425963
 230056.6213258  210945.13050263 198347.87960463 109507.73580721
  95172.17402083 235624.80083896 185026.03241713 189115.22555243
 265298.61478372 245977.22282152 106174.79544845 163722.83831967
 268130.98638874 115411.41711858 222477.7706035  101985.22089106
 226504.39006466 155293.73806267 111514.55335064 219046.27449413
  80682.37424923  84549.68544312  97057.9598615  304359.87020782
 143609.4811779  194833.30673    241386.5176735   65329.39180697
 143443.95410439 109951.22018048 438407.81054515 123036.39485624
 115640.53296556 278523.78675907 201680.22837366 237921.81634002
 206067.47683221  98147.36301669 114460.37705565 209353.26432624
 172230.80324351 161996.516503   151090.82302776 189890.90959393
 101783.9112847  168542.19256589  83160.87372137 169780.29277758
 177217.00671341 175304.43906364 130023.79083196 204628.12041568
 233299.59893945 129795.24448753 174332.38764315 258635.19788408
 229313.55596966]
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\linear_model\coordinate_descent.py:491: ConvergenceWarning: Objec
tive did not converge. You might want to increase the number of iterations. Fit
ting data with very small alpha may cause precision problems.
  ConvergenceWarning)
```

In [109]:
```
lasso_mse=mean_squared_error(y_test, lasso_y_pred)
print("LASSO_MSE:",lasso_mse)
```

```
LASSO_MSE: 1472253146.559661
```

Step 9 : RMSE

In [110]:
```
import numpy as np
#RMSE without CD
print("RMSE without CD: ",np.sqrt(mse_In))
#RMSE with CD
print("RMSE with CD: ",np.sqrt(MSE))
#RMSE with CD and Standard Scaling
print("RMSE with CD and SS: ",np.sqrt(mse_In1))
#RMSE with CD and MinMaxScaling
print("RMSE with CD and MnMaxScaling: ",np.sqrt(mmMSE))
#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:  38403.48064430541
RMSE with CD:  38371.170134872584
RMSE with CD and SS:  38371.170134877946
RMSE with CD and MnMaxScaling:  38371.17013487794
RMSE of SGDRegressor with CD and StandardScaler:  37492.52794599627
RMSE of Ridgecv with CD and Standard Scaler:  38347.51450919829
RMSE of LassoCV with CD and StandardScaler 38369.95108883592
```

In [ ]: