PML LAB 4

RETHINAGIRI

225229130

Step-1

In [1]:
```
import pandas as pd
```

In [3]:
```
f=pd.read_csv("Ames_House_Sales_Cropped.csv")
f.head()
```

Out[3]:

	BldgType	CentralAir	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	...	OverallQual	PoolArea	ScreenPorch	TotRmsAbvGr
0	1Fam	Y	856.0	854.0	0.0	3	706.0	0.0	1	0	...	7	0.0	0.0	
1	1Fam	Y	1262.0	0.0	0.0	3	978.0	0.0	0	1	...	6	0.0	0.0	
2	1Fam	Y	920.0	866.0	0.0	3	486.0	0.0	1	0	...	7	0.0	0.0	
3	1Fam	Y	961.0	756.0	0.0	3	216.0	0.0	1	0	...	7	0.0	0.0	
4	1Fam	Y	1145.0	1053.0	0.0	4	655.0	0.0	1	0	...	8	0.0	0.0	

5 rows × 39 columns

In [4]:
```
f.shape
```

Out[4]:
```
(1379, 39)
```

In [5]:
```
f.info
```

Out[5]:
```
<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
...       ...        ...       ...       ...        ...           ...
1374     1Fam          Y     953.0     694.0        0.0             3
1375     1Fam          Y    2073.0       0.0        0.0             3
1376     1Fam          Y    1188.0    1152.0        0.0             4
1377     1Fam          Y    1078.0       0.0        0.0             2
1378     1Fam          Y    1256.0       0.0        0.0             3

      BsmtFinSF1  BsmtFinSF2  BsmtFullBath  BsmtHalfBath  ...  OverallQual  \
0          706.0         0.0             1             0  ...            7
1          978.0         0.0             0             1  ...            6
2          486.0         0.0             1             0  ...            7
3          216.0         0.0             1             0  ...            7
4          655.0         0.0             1             0  ...            8
...          ...         ...           ...           ...  ...          ...
1374         0.0         0.0             0             0  ...            6
1375       790.0       163.0             1             0  ...            6
1376       275.0         0.0             0             0  ...            7
1377        49.0      1029.0             1             0  ...            5
1378       830.0       290.0             1             0  ...            5

      PoolArea  ScreenPorch  TotRmsAbvGrd  TotalBsmtSF  WoodDeckSF  YearBuilt  \
0          0.0          0.0             8        856.0         0.0       2003
1          0.0          0.0             6       1262.0       298.0       1976
2          0.0          0.0             6        920.0         0.0       2001
3          0.0          0.0             7        756.0         0.0       1915
4          0.0          0.0             9       1145.0       192.0       2000
...        ...          ...           ...          ...         ...        ...
1374       0.0          0.0             7        953.0         0.0       1999
1375       0.0          0.0             7       1542.0       349.0       1978
1376       0.0          0.0             9       1152.0         0.0       1941
1377       0.0          0.0             5       1078.0       366.0       1950
1378       0.0          0.0             6       1256.0       736.0       1965

      YearRemodAdd  YrSold  SalePrice
0             2003    2008   208500.0
1             1976    2007   181500.0
2             2002    2008   223500.0
3             1970    2006   140000.0
4             2000    2008   250000.0
...            ...     ...        ...
1374          2000    2007   175000.0
1375          1988    2010   210000.0
1376          2006    2010   266500.0
1377          1996    2010   142125.0
1378          1965    2008   147500.0

[1379 rows x 39 columns]>
```

In [6]:
```
f.columns
```

Out[6]:
```
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 [7]:
```
type(f)
```

Out[7]:
```
pandas.core.frame.DataFrame
```

In [8]:
```
f["SalePrice"].value_counts
```

Out[8]:
```
<bound method IndexOpsMixin.value_counts of 0       208500.0
1       181500.0
2       223500.0
3       140000.0
4       250000.0
          ...
1374    175000.0
1375    210000.0
1376    266500.0
1377    142125.0
1378    147500.0
Name: SalePrice, Length: 1379, dtype: float64>
```

Step-2

In [9]:
```
dff=f.drop(["BldgType","CentralAir"],axis=1)
dff
```

Out[9]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	OverallQual	PoolArea	ScreenPorch	Tot
0	856.0	854.0	0.0	3	706.0	0.0	1	0	150.0	0.0	...	7	0.0	0.0	
1	1262.0	0.0	0.0	3	978.0	0.0	0	1	284.0	0.0	...	6	0.0	0.0	
2	920.0	866.0	0.0	3	486.0	0.0	1	0	434.0	0.0	...	7	0.0	0.0	
3	961.0	756.0	0.0	3	216.0	0.0	1	0	540.0	272.0	...	7	0.0	0.0	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	0	490.0	0.0	...	8	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1374	953.0	694.0	0.0	3	0.0	0.0	0	0	953.0	0.0	...	6	0.0	0.0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	0	589.0	0.0	...	6	0.0	0.0	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	0	877.0	0.0	...	7	0.0	0.0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	0	0.0	112.0	...	5	0.0	0.0	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	0	136.0	0.0	...	5	0.0	0.0	

1379 rows × 37 columns

In [10]:
```
X=dff.drop(["SalePrice"],axis=1)
```

In [11]:
```
X
```

Out[11]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	OverallCond	OverallQual	PoolArea	Scre
0	856.0	854.0	0.0	3	706.0	0.0	1	0	150.0	0.0	...	5	7	0.0	
1	1262.0	0.0	0.0	3	978.0	0.0	0	1	284.0	0.0	...	8	6	0.0	
2	920.0	866.0	0.0	3	486.0	0.0	1	0	434.0	0.0	...	5	7	0.0	
3	961.0	756.0	0.0	3	216.0	0.0	1	0	540.0	272.0	...	5	7	0.0	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	0	490.0	0.0	...	5	8	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1374	953.0	694.0	0.0	3	0.0	0.0	0	0	953.0	0.0	...	5	6	0.0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	0	589.0	0.0	...	6	6	0.0	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	0	877.0	0.0	...	9	7	0.0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	0	0.0	112.0	...	6	5	0.0	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	0	136.0	0.0	...	6	5	0.0	

1379 rows × 36 columns

In [12]:
```
y=dff["SalePrice"].values
```

In [13]:
```
y
```

Out[13]:
```
array([208500., 181500., 223500., ..., 266500., 142125., 147500.])
```

In [14]:
```
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 [15]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train,y_train)
```

Out[15]:
```
LinearRegression()
```

In [16]:
```
y_pred=reg.predict(X_test)
y_pred
```

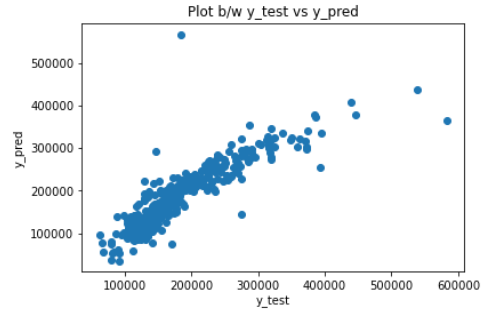
Out[16]:
```
array([257434.93050738, 111083.73474759, 100018.05832296, 204028.53821318,
       207319.25418309,  38036.30928931, 234153.38582866, 205076.12689602,
       187014.12655235, 235636.78799033, 100976.50943778, 304119.53647002,
       101769.64600707, 288758.46001588, 204767.89338323, 145002.47279612,
       248322.97436864, 151533.68038635, 209697.39458707, 278312.23574175,
        93329.85601468, 153784.14868376, 163237.8745113 , 241495.91212894,
       372429.94726948, 221672.07367619, 119658.75685735, 100039.78137073,
       319435.0289208 , 172088.64665034, 258772.64470128, 199597.36129644,
       156349.69283207, 142311.22500104, 204807.33161316, 379608.17785445,
       124176.23377551, 141127.25006137, 273832.43247659, 212547.04205549,
       169474.57176144, 123305.6571931 , 200341.24016803, 377957.3230709 ,
       146941.71712386, 283693.48760671, 106694.19786716, 221733.02172501,
        87733.47014007, 295005.66546301, 125250.6201929 ,  53569.11256026,
       133831.05086112, 170901.42929716, 213273.17757308, 110818.14723236,
       103893.81849705, 200665.21921036, 137764.84464912, 311840.64900742,
        53035.52267584, 184866.47797854, 154675.79963886, 298477.53600012,
        78162.54074678, 323728.93535808, 169417.44138503, 112718.07937644,
       200547.87648954, 205739.30910761, 130292.88475651, 271581.37221151,
       119489.76459229, 126635.29232502,  77972.26780477, 138563.37305257,
        35555.51251476, 163787.98080897, 271929.13690182, 115260.62742898,
       319958.55004397, 218515.16541522, 291730.97232818, 211176.53228427,
        93761.81278835, 263031.10150326, 141886.39666989, 118491.504161  ,
       143580.34191294, 278395.85370475, 265558.84177945, 300839.51324234,
       182673.34437972, 168604.54926029, 148322.24627429, 213781.03244617,
       247750.58741003, 221693.9068679 , 274474.96290575, 242912.73279152,
       111041.26737677, 100160.6657377 , 193531.44933258, 206050.88712636,
       154584.47386045, 217440.52365098, 210904.2353926 , 125864.27688692,
       171322.66919863, 224459.70210703, 155548.12334501, 111942.78717752,
       316624.83474103, 298116.31992286, 335573.77762729,  86738.88506815,
       146692.58771903, 228471.54032916, 123374.07354539, 204867.33148102,
       217343.21720915, 120569.43114728, 134354.12384259, 223860.01643724,
       138360.7626746 , 210765.75439353, 145423.97229835, 291155.26974374,
       183089.1790414 ,  98716.55524165, 323896.46706854, 170447.3065711 ,
       123643.153463  , 134345.24799847, 309423.49789828, 136836.97808941,
       282083.02092589, 132621.22372549,  90821.15043375, 164193.14328597,
       147333.60026799, 162030.92949962, 177621.20328218, 254054.45691359,
       119092.92298356, 140081.84030885, 236509.39317224, 318578.43961653,
       105538.81578048, 192023.58779381, 174531.98294666, 146314.75164219,
        95146.46236358, 250936.32825845, 304056.57810759,  55008.42319128,
       281810.05395734,  93402.68236909, 138283.44603752, 166415.31922931,
       209758.2699855 , 205602.16976218, 177026.58819399, 248435.62725105,
       145190.55457315, 133283.94785766, 171593.4339452 , 344925.23412408,
       181194.80288342,  98431.28117762, 131287.46561778, 110849.96802294,
       131283.81966774, 194221.20494203, 156966.39268753, 268340.58471938,
       211804.83660277, 151971.72881378, 115946.37551394, 243618.89320957,
       136292.76068258, 265335.41730194, 302175.59064203,  85054.45272479,
       146007.43532802, 150439.28821422, 159145.83079286, 232246.23120845,
       231571.13532006, 139178.99313673, 355030.61932411, 127659.29864105,
       235339.2372418 , 116707.21916897, 103103.75191046, 159574.23418447,
       149485.12469045, 408898.40096815, 335368.6502612 , 290500.91914901,
       281845.43486747, 275341.81718408, 321705.52591347, 307549.31796905,
       201661.19536928, 142047.81494957, 157876.45367578, 258695.10810337,
       207592.92972053, 146121.60182469, 107032.82242148, 151245.32235214,
       100804.20363826, 296980.23153198, 341413.30991668, 255158.86136982,
       147006.34954483, 199721.99255167, 127291.41093947, 264030.58756768,
       121760.19396638, 134138.79148627, 307661.7922912 , 170331.68886889,
       173593.89056668, 567240.20119429, 182277.72265592, 144159.48526089,
       199515.17896327, 107691.2050376 , 181383.00130506, 328401.72330992,
       139225.20214763, 197660.10401728, 116340.37479626,  54949.57465381,
       199375.67314461, 272915.38777829, 119902.62390782, 197573.81604284,
       231881.98496926, 119555.98822719, 128709.5470231 , 289158.80585503,
       200733.61408624, 364228.3936612 , 119308.94568433, 124086.72254029,
       225161.62739515, 171005.22659818,  93631.91767232, 178506.48207587,
       215661.64584808, 226225.30038969, 132391.85608606,  73443.41948302,
       242913.00968345, 141042.39030916, 114633.38186068,  89093.07024783,
       208503.15479636, 172828.98634138, 270797.22136171, 253693.44924268,
       103518.85057195, 220521.27460705, 185034.32129267, 169518.90296989,
       154846.17459055, 224605.38839963,  85972.23828464, 151323.12661368,
       180628.92554334, 200106.98029834, 179499.81727934, 223755.69902629,
        73639.52734131,  86011.81491639, 110000.03909843, 231518.89807254,
       153369.35976787, 320899.35432169, 194412.83170319,  57741.65743007,
       203877.6301523 , 109803.05085705, 194633.3625692 ,  99160.3592207 ,
       230240.81594384, 211484.7747706 , 198342.97693777, 109749.28794509,
        95564.23930699, 235396.44527251, 184775.90385438, 188806.96222904,
       265328.01705252, 245862.13668727, 107422.00402522, 164786.63106767,
       268303.44421418, 116075.01592358, 222914.33666733, 102600.43292175,
       226400.49839169, 156093.55758654, 112003.82907508, 218089.37220871,
        81085.92660259,  85015.51341445,  98208.34263396, 304262.27895911,
       143757.79043413, 195367.35122022, 241835.21911125,  60944.84144927,
       143731.15831775, 110459.58351747, 438412.24654506, 123265.07239631,
       116119.8793081 , 277961.71617824, 201687.81739718, 237947.02580913,
       206117.84987326,  99453.60459329, 115124.57035038, 209795.31353815,
       167537.1556324 , 162970.2651776 , 151279.43057799, 189724.86274086,
        96455.47006123, 168652.82586917,  83051.31596965, 171340.4295985 ,
       176868.33711876, 175644.68961811, 131414.67048235, 204543.74960373,
       233147.40044688, 125478.90203716, 175060.50167492, 258877.30470765,
       229115.6512967 ])
```

In [17]:
```
import sklearn.metrics as metrics
mse=metrics.mean_squared_error(y_test,y_pred)
print("MSE without Categorical: ",mse)
```

```
MSE without Categorical:  1474827325.5979307
```

Step-3

In [18]:
```
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()
```



Step-4

In [19]:
```
encoding=pd.get_dummies(f)
```

In [20]:
```
encoding
```

Out[20]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	YearRemodAdd	YrSold	SalePrice	BldgT
0	856.0	854.0	0.0	3	706.0	0.0	1	0	150.0	0.0	...	2003	2008	208500.0	
1	1262.0	0.0	0.0	3	978.0	0.0	0	1	284.0	0.0	...	1976	2007	181500.0	
2	920.0	866.0	0.0	3	486.0	0.0	1	0	434.0	0.0	...	2002	2008	223500.0	
3	961.0	756.0	0.0	3	216.0	0.0	1	0	540.0	272.0	...	1970	2006	140000.0	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	0	490.0	0.0	...	2000	2008	250000.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1374	953.0	694.0	0.0	3	0.0	0.0	0	0	953.0	0.0	...	2000	2007	175000.0	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	0	589.0	0.0	...	1988	2010	210000.0	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	0	877.0	0.0	...	2006	2010	266500.0	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	0	0.0	112.0	...	1996	2010	142125.0	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	0	136.0	0.0	...	1965	2008	147500.0	

1379 rows × 44 columns

Step-5

In [21]:
```
X=encoding.drop(["SalePrice"],axis=1)
```

In [22]:
```
X
```

Out[22]:

	1stFlrSF	2ndFlrSF	3SsnPorch	BedroomAbvGr	BsmtFinSF1	BsmtFinSF2	BsmtFullBath	BsmtHalfBath	BsmtUnfSF	EnclosedPorch	...	YearBuilt	YearRemodAdd	YrSold	BldgTy
0	856.0	854.0	0.0	3	706.0	0.0	1	0	150.0	0.0	...	2003	2003	2008	
1	1262.0	0.0	0.0	3	978.0	0.0	0	1	284.0	0.0	...	1976	1976	2007	
2	920.0	866.0	0.0	3	486.0	0.0	1	0	434.0	0.0	...	2001	2002	2008	
3	961.0	756.0	0.0	3	216.0	0.0	1	0	540.0	272.0	...	1915	1970	2006	
4	1145.0	1053.0	0.0	4	655.0	0.0	1	0	490.0	0.0	...	2000	2000	2008	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1374	953.0	694.0	0.0	3	0.0	0.0	0	0	953.0	0.0	...	1999	2000	2007	
1375	2073.0	0.0	0.0	3	790.0	163.0	1	0	589.0	0.0	...	1978	1988	2010	
1376	1188.0	1152.0	0.0	4	275.0	0.0	0	0	877.0	0.0	...	1941	2006	2010	
1377	1078.0	0.0	0.0	2	49.0	1029.0	1	0	0.0	112.0	...	1950	1996	2010	
1378	1256.0	0.0	0.0	3	830.0	290.0	1	0	136.0	0.0	...	1965	1965	2008	

1379 rows × 43 columns

In [23]:
```
y=encoding["SalePrice"].values
```

In [24]:
```
y
```

Out[24]:
```
array([208500., 181500., 223500., ..., 266500., 142125., 147500.])
```

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

In [26]:
```
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:  1461036570.1434536
```

Step-6

In [27]:
```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
ss_X_train=ss.fit_transform(X_train)
ss_X_train
```

Out[27]:
```
array([[ 0.39851037, -0.79290427, -0.11340519, ...,  3.45325933,
        -0.22777619,  0.22777619],
       [ 1.57467708, -0.79290427, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [ 0.37564751,  0.70143387, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       ...,
       [ 1.22157303, -0.79290427, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [-1.11297817,  0.90510323, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [ 0.11145456, -0.79290427, -0.11340519, ...,  3.45325933,
        -0.22777619,  0.22777619]])
```

In [28]:
```
ss_X_test=ss.transform(X_test)
ss_X_test
```

Out[28]:
```
array([[ 0.85830772, -0.79290427, -0.11340519, ...,  3.45325933,
        -0.22777619,  0.22777619],
       [-0.64810018, -0.79290427, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [-0.21624632,  0.76093278, -0.11340519, ..., -0.2895815 ,
         4.39027446, -4.39027446],
       ...,
       [-0.04350477,  0.37647826, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [-0.64301955,  1.46805451, -0.11340519, ..., -0.2895815 ,
        -0.22777619,  0.22777619],
       [-0.29499614,  0.81585486, -0.11340519, ...,  3.45325933,
        -0.22777619,  0.22777619]])
```

In [29]:
```
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[29]:
```
array([257746.67634751, 113257.14215097,  83086.83449793, 204109.44696503,
       209062.51895279,  69091.78945845, 235181.17333175, 205722.26326666,
       187627.82768487, 234276.07879635, 101443.5208025 , 304189.5355825 ,
       102111.08803612, 289185.34424259, 204499.21833429, 142096.29751997,
       249636.03306251, 147881.8828702 , 210574.13858086, 277457.62922997,
        91416.02948336, 153937.03347396, 160626.96732406, 243147.94924865,
       372541.07042985, 219861.59417986, 116849.37194182, 100880.38852652,
       318604.3072889 , 168091.93524469, 258085.96044038, 201316.99711925,
       154599.59088249, 142746.1064703 , 205138.6701111 , 382254.53304498,
       124930.2956851 , 141597.13070899, 273664.48982728, 214149.35804835,
       170457.21544846, 124570.77597839, 197619.98987752, 376155.99238854,
       148003.84323244, 288043.80788135, 108672.5169885 , 218565.03839169,
        88455.07127747, 295191.90839717, 125798.92162385,  54900.46063499,
       133431.33010679, 173734.6432617 , 214868.38940441, 110305.00323981,
       101777.6571833 , 198417.39484009, 139859.94217493, 313011.39577057,
        71373.41173384, 185200.29760901, 153952.30442378, 299603.66711185,
        76797.97905956, 324386.66707621, 165286.38031963, 112128.08996347,
       200841.50826584, 206567.94099559, 130662.03800731, 271533.51448715,
       119169.71075355, 144780.32426223,  75275.27013335, 136814.73927026,
        35548.77310788, 163999.4816694 , 273975.35104906, 116033.21218901,
       319960.29175167, 219051.51124482, 293875.7312285 , 211801.06700944,
        91269.29114983, 265733.72817127, 142679.76528577, 118577.18340652,
       143674.08252179, 279159.65322884, 269371.00847357, 301749.6475942 ,
       184234.99923566, 165120.97530255, 149749.83444921, 213942.08827205,
       249106.5262464 , 224105.75907123, 274402.65899366, 245923.74180536,
       111650.50965645,  99272.63102469, 195172.55269882, 208067.81804138,
       155001.86752871, 218149.55729734, 212687.26428796, 122414.02996443,
       162732.20657544, 221859.13237471, 146948.24111576, 109948.53523129,
       319916.30228215, 298406.39751083, 337078.1346426 ,  88053.83690322,
       146957.67256334, 227538.58670511, 122778.70033749, 205374.95008286,
       217590.01872639, 107383.84101482, 133741.2940345 , 220233.83033811,
       134541.44321812, 212453.09527909, 145667.19230495, 291995.67805595,
       183987.92894577,  97709.62989828, 325909.11123175, 168153.0537991 ,
       122835.11047077, 134056.79102667, 309609.27342891, 134547.87576178,
       281848.2892548 , 130902.414594  ,  92709.96908775, 160928.91702672,
       148231.56791932, 158731.16111448, 178201.93181948, 257204.46954565,
       114890.4575372 , 138122.68344433, 238125.08207166, 318625.89235561,
       104214.83201718, 193580.28738241, 172127.95271212, 144833.00763331,
        95559.79845962, 250974.08034067, 304496.84969754,  53072.22208749,
       284199.95101748,  93322.70762404, 137832.3476692 , 164298.42069277,
       209757.60013178, 203537.90106371, 176604.23461749, 250435.04474144,
       142419.38528636, 134385.83846204, 171734.49004071, 344239.73144709,
       182540.85994093,  99608.62865269, 131558.48391034, 102401.37979263,
       123453.73975651, 193361.22440516, 156609.55466075, 269877.79864932,
       214063.10991695, 151945.88694257, 116602.06046004, 244961.96222125,
       134853.80033622, 264897.14158982, 302718.95386478, 112752.48763543,
       146428.01022778, 141383.04049712, 159106.57051208, 231871.86707197,
       233361.12369078, 140206.8801076 , 356203.87108972, 125107.24732751,
       236557.13984092, 118583.44962369, 103727.34373498, 158205.16435346,
       146206.40366386, 414475.89127582, 337712.90757194, 292098.02863639,
       282089.2821333 , 275216.7535936 , 322615.55168033, 307605.46888381,
       202379.61396859, 141568.5679281 , 156911.80391702, 264235.34590098,
       208838.68106865, 148493.50013207, 104577.21175208, 149737.50802682,
       101508.22184593, 298194.31400841, 342555.6575081 , 255293.49747035,
       143932.11991761, 195052.21568926, 128768.16960182, 264721.74453629,
       122744.14619783, 133171.58386505, 307876.67207273, 167186.04489823,
       173665.54160608, 566793.1709959 , 182882.72846953, 142976.55390157,
       195389.49394275, 110000.52031067, 181425.05998225, 326697.98513269,
       138753.30659611, 197531.51535532, 116224.68828072,  57916.66921872,
       200892.30994253, 274120.15715879, 110287.72265007, 195830.60427729,
       234290.62236658, 121029.23351415, 137768.1152297 , 289104.52688668,
       200107.84920963, 364178.66342092, 116071.36887504, 125706.22322028,
       225918.36241065, 172279.20082007,  93552.9758358 , 178432.96188201,
       218235.04083371, 225391.98842006, 133701.6669314 ,  67245.43233413,
       243425.96315121, 138199.03345666, 113586.6774028 ,  84741.3662601 ,
       207895.82887217, 172018.96542022, 270379.96625141, 254301.48609865,
       103491.65134387, 221152.52727064, 184240.39768591, 170413.54111433,
       156767.28870157, 225874.2187855 ,  85067.50220226, 152614.17988863,
       180110.39735029, 200365.40564938, 181240.47868432, 225088.28569024,
        74809.64993436,  81056.03121343, 138531.42801458, 231176.80557566,
       152880.69516053, 322880.51400012, 196244.29857021,  56720.24928342,
       206525.2911202 , 109558.50672761, 196173.16696132, 100008.43053437,
       222772.90040602, 211538.69107333, 199089.61402063, 111290.16182517,
        98159.20986117, 236839.06037874, 184863.02084498, 189201.7028789 ,
       265541.99840726, 247878.42019041, 108244.54720887, 161881.3193246 ,
       267831.06097132, 116127.88591933, 220313.03060259,  98999.66742238,
       227727.64329725, 158059.62456719, 107633.36590698, 219812.49714578,
        81429.50018227,  76334.09584861,  97716.92609735, 303748.10861981,
       146166.41447046, 186676.02055521, 241412.73122161,  61413.0262216 ,
       144287.26086056, 108854.17859528, 440736.67250284, 120123.18222097,
       115307.43887445, 275889.34669779, 201914.61172598, 234443.75355834,
       201332.72073446,  98680.14396138, 142896.01985585, 211430.3435048 ,
       167955.3826565 , 161745.94065555, 150329.86791651, 191659.61015824,
        98638.48036052, 167674.58655871,  81251.41418888, 169289.1270314 ,
       175822.84359349, 165634.36808713, 132604.67079394, 205548.45244987,
       236536.65155886, 125744.91988448, 173333.02038791, 259092.37349783,
       227361.55928746])
```

In [35]:
```
ss_mse=metrics.mean_squared_error(y_test,ss_y_pred)
print("SS_MSE: ",ss_mse)
```

```
SS_MSE:  1461036570.1437423
```

Step-7

In [36]:
```
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: [257856. 113152.  83264. 204096. 209024.  69184. 235136. 205760. 187712.
 234304. 101376. 304192. 102080. 289216. 204608. 142144. 249536. 147968.
 210496. 277568.  91456. 153920. 160704. 243008. 372544. 219968. 116928.
 100800. 318720. 168256. 258176. 201280. 154688. 142656. 205120. 382208.
 124864. 141504. 273664. 214144. 170432. 124416. 197824. 376128. 148096.
 288000. 108608. 218688.  88512. 295360. 125632.  55232. 133440. 173632.
 214848. 110400. 101760. 198464. 139840. 313024.  71296. 185152. 153984.
 299584.  76864. 324416. 165120. 112128. 200768. 206528. 130752. 271488.
 119232. 144960.  75328. 136704.  35712. 164032. 273856. 116160. 320064.
 219072. 293824. 211776.  91328. 265664. 142656. 118592. 143680. 279168.
 269248. 301760. 184256. 165056. 149824. 213952. 249152. 224000. 274304.
 245952. 111552.  99264. 195200. 208192. 155008. 218048. 212736. 122560.
 163008. 222144. 147264. 110016. 319936. 298304. 336960.  88000. 147008.
 227520. 122816. 205376. 217536. 107456. 133760. 220224. 134592. 212416.
 145600. 292096. 184000.  97728. 325824. 168000. 122944. 134080. 309568.
 134464. 281920. 131008.  92864. 161024. 148224. 158784. 178240. 257088.
 115008. 138176. 238080. 318656. 104256. 193600. 172224. 144832.  95552.
 251008. 304640.  53120. 284160.  93248. 137920. 164352. 209728. 203392.
 176704. 250560. 142528. 134464. 171776. 344128. 182720.  99648. 131520.
 102720. 123712. 193216. 156608. 269888. 214080. 151872. 116800. 244992.
 134784. 265088. 302784. 112832. 146368. 141632. 159168. 231872. 233280.
 140096. 356288. 125120. 236608. 118464. 103744. 158272. 146240. 414272.
 337728. 292032. 282048. 275264. 322560. 307648. 202496. 141504. 156928.
 264064. 208832. 148544. 104768. 149824. 101696. 298304. 342592. 255424.
 144000. 195008. 128832. 264768. 122624. 133184. 307904. 167360. 173632.
 566656. 182912. 142976. 195520. 110080. 181440. 326720. 138688. 197504.
 116288.  58112. 200896. 274240. 110528. 195840. 234304. 121024. 137792.
 289088. 200192. 364288. 116160. 125632. 225920. 172224.  93504. 178496.
 218240. 225408. 133824.  67520. 243520. 138176. 113472.  84928. 207872.
 171840. 270400. 254144. 103488. 221120. 184448. 170432. 156864. 225856.
  85056. 152448. 180032. 200384. 181184. 225088.  74752.  80704. 138496.
 231232. 152896. 322880. 196224.  56768. 206528. 109568. 196224. 100224.
 222912. 211520. 199104. 111168.  98176. 236992. 184896. 189248. 265664.
 247872. 108224. 161728. 267840. 116160. 220224.  99008. 227712. 157888.
 107712. 219840.  81344.  76544.  97664. 303936. 146112. 186944. 241472.
  61632. 144320. 108736. 440640. 120256. 115264. 275776. 201984. 234560.
 201536.  98752. 142912. 211392. 167872. 161728. 150464. 191808.  98624.
 167872.  81344. 169024. 175936. 165824. 132608. 205632. 236544. 125824.
 173312. 259072. 227264.]
```

In [33]:
```
mm_mse=metrics.mean_squared_error(y_test,mm_y_pred)
print("MM_MSE: ",mm_mse)
```

```
MM_MSE:  1460764136.5826087
```

In [37]:
```
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: [256699.55675145 108270.81327652  84608.48944827 205217.983591
 210676.60505679  56503.68096047 238030.40747691 206252.90978211
 188812.0945854  235456.0814532   97755.46724458 309800.47972126
  96148.85980304 294704.31054947 204348.32393375 144552.50228988
 250920.67557439 148459.42318982 210010.6834599  281419.9385047
  89404.75473171 151164.16879069 162894.03232371 246076.19327909
 379278.67133696 221866.78513915 119176.6828244   96386.29638084
 327631.74310557 171496.91421866 260302.53310381 201781.91575768
 145124.0904969  141309.06067739 205798.50368186 389197.16063679
 118902.76628196 143656.53351216 277038.39461066 211322.41943309
 169899.3594215  122099.54955956 197998.96484562 384474.37016966
 147807.55103101 289716.90234936 103805.68232752 219198.08321274
  83492.9895041  301189.34796742 123065.22225171  45531.27099318
 138991.37791797 174530.63264005 216098.71948988 109322.54082394
  99965.58226367 202751.1400441  139364.80023436 318410.00337539
  61465.05191011 186101.79176137 153852.5526244  302420.50911111
  72858.06205264 332904.45266678 165463.95442892 107216.06422831
 198837.85410754 206864.29764512 130584.38881588 272770.34881489
 115074.6888065  142772.37997452  67312.76105467 131293.91250428
  25948.69491477 164181.88427806 277058.92676621 112548.97572315
 327792.21789442 222058.03851278 298116.1927334  215037.52003253
  92098.51093816 270454.55651086 138125.38974617 112418.18250265
 138443.7998042  284543.07384689 275299.08744923 305186.0958216
 184704.32243927 163686.32715906 151152.76167015 216143.33675991
 252558.94016011 225715.15098641 275994.45416427 247945.9492843
 109050.57866552  98370.98937964 195566.4734051  210968.73632705
 153252.44088019 220809.47728718 213385.38793903 122494.07004253
 160849.58536118 227236.68537699 144569.62498338 108511.27648307
 327355.28225282 301238.90812415 341669.35219495  81888.04248974
 145977.92487157 227412.16076279 122007.4604947  203774.59775366
 216884.77193532 108155.31350168 127462.64744236 224138.53038984
 135625.32274674 212838.58216615 142599.35288279 295037.55736448
 184399.97674879  91404.95616809 329919.26269325 169455.15572924
 119789.49636009 128391.84777813 315532.20456102 135543.26743824
 285406.97470614 127711.43114131  85074.29776806 164685.12420604
 144894.18393767 160839.84462897 178539.88428394 258530.71977865
 117968.36210765 134335.74479537 241867.79324415 324200.14930758
 101159.3837695  194607.11194963 171637.20314494 143939.12884803
  89608.16185877 252607.60847993 312032.37327327  44708.57545258
 288821.87542533  90804.5883612  135974.35699547 168067.76214621
 209549.26121221 204050.55662035 178801.9150161  256152.09207086
 146707.02247333 133011.72710008 171484.3394244  351793.22639509
 183446.22375476  93063.65540924 127034.42374449  96855.62685816
 116910.66650821 189593.17023991 154529.10572426 269790.90875521
 214061.1748815  153143.15035762 114534.09355955 248115.72334262
 129852.06958628 268569.5707596  308720.80609232 107440.50417918
 141592.48471796 136544.27383809 158556.30840844 235315.08104045
 236022.93571922 136528.17307914 363568.2982996  122435.73538513
 239649.32757173 114808.22963513  97707.51389001 160381.71651543
 146492.04456216 421065.18543469 346455.35665951 296083.80131606
 284925.10302275 279969.13705534 326994.40095288 312523.33547846
 204007.05130792 137822.64289246 155850.43083228 311320.31113969
 210475.4105662  146968.22601293 104367.63970701 149409.68714306
  95687.23202674 303947.45582609 350051.35329245 258979.40459528
 143522.07087284 196611.2312449  125857.30793037 268682.68247839
 123637.39069515 132976.25510871 316486.74751526 171008.41575475
 172640.76457231 578974.52451936 184373.65047588 141304.71700614
 197276.65654132 105837.70000478 180048.56725486 334866.91390354
 136199.04194821 195779.33652043 115335.26682229  48370.1360113
 200407.30066601 275394.65021453 105601.03360628 199282.86279696
 237672.18482035 117354.29120366 133819.26703689 292211.54078998
 203328.87773557 372428.48405617 114918.97101107 121668.57531644
 228652.8060939  171105.12805673  86514.76282147 179313.21013584
 219598.8346385  228672.58494993 131483.45590177  64534.19485579
 246780.9282554  139245.60299231 107948.46871067  83666.44097165
 208722.64190763 170069.38159987 275145.86683018 257744.56368554
  97415.18252297 220499.49853182 185617.77545484 169252.40785203
 155072.98098281 228849.80873161  79907.31981992 151392.51793364
 178002.01940198 200122.03392051 181735.7121151  226354.9389128
 116556.65240395  80988.55636625 131825.19540797 233757.15189133
 150575.01047897 326405.90266729 197146.13889169  50648.02656945
 209325.36667328 105699.07000544 196682.16300754  95141.12886462
 223294.38452658 215221.56374249 202460.28442983 109652.12935635
  95646.28803869 241230.88396729 188567.195621   191815.46868052
 269606.81336081 250230.86452371 106267.28113618 154874.90294568
 273535.61162641 113690.7974939  220201.13384305 100883.00442384
 228350.02584768 157421.06015042 107377.65642014 219666.37428052
  79801.85971533  71214.88378381  91563.12240827 308640.46097108
 201753.18037882 181486.5290299  251227.41164518  51341.60518713
 141787.32789983 110872.1322341  450669.56298093 118219.90331097
 111772.77101268 281463.61094962 201601.1452637  238424.04018681
 208764.81571736  96239.80082871 137165.11707895 211824.0583651
 158567.16848443 159826.21134029 149761.90919389 190181.08619274
  92176.56454238 168110.33803351  73997.69567682 161407.36778928
 177691.17709125 161420.8597646  130923.79609171 204090.29401821
 239163.36316816 121531.10119513 176084.09254369 263821.04742065
 227415.9752887 ]
```

In [ ]: