House Price Prediction Project

By Shaikh Abdul Quddus

Libraries

In [18]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder,StandardScaler,OneHotEncoder

from sklearn.feature_selection import SequentialFeatureSelector
from sklearn.linear_model import LinearRegression

from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer

from sklearn.model_selection import train_test_split

from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

Data Gathering

In [4]:
```
df=pd.read_csv('E:/Shaikh Quddus/Classes Recordings/project 1/training_set.csv')
```

EDA

In [5]:
```
df.isna().sum()
```

Out[5]:
```
Id                 0
MSSubClass         0
MSZoning           0
LotFrontage      259
LotArea            0
                ... 
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64
```

In [6]:
```
df.duplicated()
```

Out[6]:
```
0       False
1       False
2       False
3       False
4       False
        ...
1455    False
1456    False
1457    False
1458    False
1459    False
Length: 1460, dtype: bool
```

In [8]:
```
df.nunique()
```

Out[8]:
```
Id               1460
MSSubClass         15
MSZoning            5
LotFrontage       110
LotArea          1073
                 ...
MoSold             12
YrSold              5
SaleType            9
SaleCondition       6
SalePrice         663
Length: 81, dtype: int64
```

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             1460 non-null   int64  
 1   MSSubClass     1460 non-null   int64  
 2   MSZoning       1460 non-null   object 
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64  
 5   Street         1460 non-null   object 
 6   Alley          91 non-null     object 
 7   LotShape       1460 non-null   object 
 8   LandContour    1460 non-null   object 
 9   Utilities      1460 non-null   object 
 10  LotConfig      1460 non-null   object 
 11  LandSlope      1460 non-null   object 
 12  Neighborhood   1460 non-null   object 
 13  Condition1     1460 non-null   object 
 14  Condition2     1460 non-null   object 
 15  BldgType       1460 non-null   object 
 16  HouseStyle     1460 non-null   object 
 17  OverallQual    1460 non-null   int64  
 18  OverallCond    1460 non-null   int64  
 19  YearBuilt      1460 non-null   int64  
 20  YearRemodAdd   1460 non-null   int64  
 21  RoofStyle      1460 non-null   object 
 22  RoofMatl       1460 non-null   object 
 23  Exterior1st    1460 non-null   object 
 24  Exterior2nd    1460 non-null   object 
 25  MasVnrType     1452 non-null   object 
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object 
 28  ExterCond      1460 non-null   object 
 29  Foundation     1460 non-null   object 
 30  BsmtQual       1423 non-null   object 
 31  BsmtCond       1423 non-null   object 
 32  BsmtExposure   1422 non-null   object 
 33  BsmtFinType1   1423 non-null   object 
 34  BsmtFinSF1     1460 non-null   int64  
 35  BsmtFinType2   1422 non-null   object 
 36  BsmtFinSF2     1460 non-null   int64  
 37  BsmtUnfSF      1460 non-null   int64  
 38  TotalBsmtSF    1460 non-null   int64  
 39  Heating        1460 non-null   object 
 40  HeatingQC      1460 non-null   object 
 41  CentralAir     1460 non-null   object 
 42  Electrical     1459 non-null   object 
 43  1stFlrSF       1460 non-null   int64  
 44  2ndFlrSF       1460 non-null   int64  
 45  LowQualFinSF   1460 non-null   int64  
 46  GrLivArea      1460 non-null   int64  
 47  BsmtFullBath   1460 non-null   int64  
 48  BsmtHalfBath   1460 non-null   int64  
 49  FullBath       1460 non-null   int64  
 50  HalfBath       1460 non-null   int64  
 51  BedroomAbvGr   1460 non-null   int64  
 52  KitchenAbvGr   1460 non-null   int64  
 53  KitchenQual    1460 non-null   object 
 54  TotRmsAbvGrd   1460 non-null   int64  
 55  Functional     1460 non-null   object 
 56  Fireplaces     1460 non-null   int64  
 57  FireplaceQu    770 non-null    object 
 58  GarageType     1379 non-null   object 
```

```
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
 69  3SsnPorch      1460 non-null   int64
 70  ScreenPorch    1460 non-null   int64
 71  PoolArea       1460 non-null   int64
 72  PoolQC         7 non-null      object
 73  Fence          281 non-null    object
 74  MiscFeature    54 non-null     object
 75  MiscVal        1460 non-null   int64
 76  MoSold         1460 non-null   int64
 77  YrSold         1460 non-null   int64
 78  SaleType       1460 non-null   object
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

Define x and y

In [13]:
```
x=df.drop(['SalePrice','Id'],axis=1)
y=df['SalePrice']
```

Preprocessing for Feature Selection

Missing values handling

In [14]:
```
for i in x.columns:
    if x[i].dtypes=='object':
        x[i]=x[i].fillna(x[i].mode()[0])
    else:
        x[i]=x[i].fillna(x[i].median())
```

In [16]:
```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 79 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   MSSubClass     1460 non-null   int64  
 1   MSZoning       1460 non-null   object 
 2   LotFrontage    1460 non-null   float64
 3   LotArea        1460 non-null   int64  
 4   Street         1460 non-null   object 
 5   Alley          1460 non-null   object 
 6   LotShape       1460 non-null   object 
 7   LandContour    1460 non-null   object 
 8   Utilities      1460 non-null   object 
 9   LotConfig      1460 non-null   object 
 10  LandSlope      1460 non-null   object 
 11  Neighborhood   1460 non-null   object 
 12  Condition1     1460 non-null   object 
 13  Condition2     1460 non-null   object 
 14  BldgType       1460 non-null   object 
 15  HouseStyle     1460 non-null   object 
 16  OverallQual    1460 non-null   int64  
 17  OverallCond    1460 non-null   int64  
 18  YearBuilt      1460 non-null   int64  
 19  YearRemodAdd   1460 non-null   int64  
 20  RoofStyle      1460 non-null   object 
 21  RoofMatl       1460 non-null   object 
 22  Exterior1st    1460 non-null   object 
 23  Exterior2nd    1460 non-null   object 
 24  MasVnrType     1460 non-null   object 
 25  MasVnrArea     1460 non-null   float64
 26  ExterQual      1460 non-null   object 
 27  ExterCond      1460 non-null   object 
 28  Foundation     1460 non-null   object 
 29  BsmtQual       1460 non-null   object 
 30  BsmtCond       1460 non-null   object 
 31  BsmtExposure   1460 non-null   object 
 32  BsmtFinType1   1460 non-null   object 
 33  BsmtFinSF1     1460 non-null   int64  
 34  BsmtFinType2   1460 non-null   object 
 35  BsmtFinSF2     1460 non-null   int64  
 36  BsmtUnfSF      1460 non-null   int64  
 37  TotalBsmtSF    1460 non-null   int64  
 38  Heating        1460 non-null   object 
 39  HeatingQC      1460 non-null   object 
 40  CentralAir     1460 non-null   object 
 41  Electrical     1460 non-null   object 
 42  1stFlrSF       1460 non-null   int64  
 43  2ndFlrSF       1460 non-null   int64  
 44  LowQualFinSF   1460 non-null   int64  
 45  GrLivArea      1460 non-null   int64  
 46  BsmtFullBath   1460 non-null   int64  
 47  BsmtHalfBath   1460 non-null   int64  
 48  FullBath       1460 non-null   int64  
 49  HalfBath       1460 non-null   int64  
 50  BedroomAbvGr   1460 non-null   int64  
 51  KitchenAbvGr   1460 non-null   int64  
 52  KitchenQual    1460 non-null   object 
 53  TotRmsAbvGrd   1460 non-null   int64  
 54  Functional     1460 non-null   object 
 55  Fireplaces     1460 non-null   int64  
 56  FireplaceQu    1460 non-null   object 
 57  GarageType     1460 non-null   object 
 58  GarageYrBlt    1460 non-null   float64
```

```
 59  GarageFinish   1460 non-null   object
 60  GarageCars     1460 non-null   int64
 61  GarageArea     1460 non-null   int64
 62  GarageQual     1460 non-null   object
 63  GarageCond     1460 non-null   object
 64  PavedDrive     1460 non-null   object
 65  WoodDeckSF     1460 non-null   int64
 66  OpenPorchSF    1460 non-null   int64
 67  EnclosedPorch  1460 non-null   int64
 68  3SsnPorch      1460 non-null   int64
 69  ScreenPorch    1460 non-null   int64
 70  PoolArea       1460 non-null   int64
 71  PoolQC         1460 non-null   object
 72  Fence          1460 non-null   object
 73  MiscFeature    1460 non-null   object
 74  MiscVal        1460 non-null   int64
 75  MoSold         1460 non-null   int64
 76  YrSold         1460 non-null   int64
 77  SaleType       1460 non-null   object
 78  SaleCondition  1460 non-null   object
dtypes: float64(3), int64(33), object(43)
memory usage: 901.2+ KB
```

In [17]:
```
cat=x.select_dtypes(include='object')
con=x.select_dtypes(exclude='object')
```

In [19]:
```
le=LabelEncoder()
```

In [20]:
```
cat1=cat.apply(le.fit_transform)
cat1
```

Out[20]:

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighbor
0	3	1	0	3	3	0	4	0	
1	3	1	0	3	3	0	2	0	
2	3	1	0	0	3	0	4	0	
3	3	1	0	0	3	0	0	0	
4	3	1	0	0	3	0	2	0	
...	...	...	...	...	...	...	...	...	
1455	3	1	0	3	3	0	4	0	
1456	3	1	0	3	3	0	4	0	
1457	3	1	0	3	3	0	4	0	
1458	3	1	0	3	3	0	4	0	
1459	3	1	0	3	3	0	4	0	

1460 rows × 43 columns

In [21]:
```
ss=StandardScaler()
```

In [22]:
```
con1=pd.DataFrame(ss.fit_transform(con),columns=ss.get_feature_names_out())
```

In [23]:
```
con1
```

Out[23]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd
0	0.073375	-0.220875	-0.207142	0.651479	-0.517200	1.050994	0.878668
1	-0.872563	0.460320	-0.091886	-0.071836	2.179628	0.156734	-0.429577
2	0.073375	-0.084636	0.073480	0.651479	-0.517200	0.984752	0.830215
3	0.309859	-0.447940	-0.096897	0.651479	-0.517200	-1.863632	-0.720298
4	0.073375	0.641972	0.375148	1.374795	-0.517200	0.951632	0.733308
...	...	...	...	...	...	...	...
1455	0.073375	-0.357114	-0.260560	-0.071836	-0.517200	0.918511	0.733308
1456	-0.872563	0.687385	0.266407	-0.071836	0.381743	0.222975	0.151865
1457	0.309859	-0.175462	-0.147810	0.651479	3.078570	-1.002492	1.024029
1458	-0.872563	-0.084636	-0.080160	-0.795151	0.381743	-0.704406	0.539493
1459	-0.872563	0.233255	-0.058112	-0.795151	0.381743	-0.207594	-0.962566

1460 rows × 36 columns

In [24]:
```
x1=con1.join(cat1)
x1
```

Out[24]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd
0	0.073375	-0.220875	-0.207142	0.651479	-0.517200	1.050994	0.878668
1	-0.872563	0.460320	-0.091886	-0.071836	2.179628	0.156734	-0.429577
2	0.073375	-0.084636	0.073480	0.651479	-0.517200	0.984752	0.830215
3	0.309859	-0.447940	-0.096897	0.651479	-0.517200	-1.863632	-0.720298
4	0.073375	0.641972	0.375148	1.374795	-0.517200	0.951632	0.733308
...	...	...	...	...	...	...	...
1455	0.073375	-0.357114	-0.260560	-0.071836	-0.517200	0.918511	0.733308
1456	-0.872563	0.687385	0.266407	-0.071836	0.381743	0.222975	0.151865
1457	0.309859	-0.175462	-0.147810	0.651479	3.078570	-1.002492	1.024029
1458	-0.872563	-0.084636	-0.080160	-0.795151	0.381743	-0.704406	0.539493
1459	-0.872563	0.233255	-0.058112	-0.795151	0.381743	-0.207594	-0.962566

1460 rows × 79 columns

Feature Selection

In [25]:
```
lr=LinearRegression()
```

In [156…
```
sfs=SequentialFeatureSelector(lr,n_features_to_select='auto',tol=None)
```

In [157…
```
sfs.fit(x1,y)
```

Out[157]:
```
▸ SequentialFeatureSelector
▸ estimator: LinearRegression
    ▸ LinearRegression
```

In [158…
```
cols=sfs.get_feature_names_out()
```

In [159…
```
cols
```

Out[159]:
```
array(['MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt',
       'MasVnrArea', 'BsmtFinSF1', 'GrLivArea', 'BsmtFullBath',
       'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageCars',
       'WoodDeckSF', 'OpenPorchSF', 'ScreenPorch', 'PoolArea', 'YrSold',
       'Street', 'LandContour', 'Utilities', 'Neighborhood', 'BldgType',
       'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'MasVnrType',
       'ExterQual', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'HeatingQC',
       'KitchenQual', 'Functional', 'GarageCond', 'PavedDrive', 'Fence',
       'MiscFeature'], dtype=object)
```

Final Dataset

In [160…
```
x2=pd.DataFrame(df,columns=cols)
```

In [161…
```
x2
```

Out[161]:

	MSSubClass	LotArea	OverallQual	OverallCond	YearBuilt	MasVnrArea	BsmtFinSF1	GrLivA
0	60	8450	7	5	2003	196.0	706	1
1	20	9600	6	8	1976	0.0	978	1
2	60	11250	7	5	2001	162.0	486	1
3	70	9550	7	5	1915	0.0	216	1
4	60	14260	8	5	2000	350.0	655	2
...	...	...	...	...	...	...	...	
1455	60	7917	6	5	1999	0.0	0	1
1456	20	13175	6	6	1978	119.0	790	2
1457	70	9042	7	9	1941	0.0	275	2
1458	20	9717	5	6	1950	0.0	49	1
1459	20	9937	5	6	1965	0.0	830	1

1460 rows × 39 columns

Preprocessing

In [162…

```
cat=[]
con=[]

for i in x2.columns:
    if x2[i].dtypes=='object':
        cat.append(i)
    else:
        con.append(i)
```

In [163…

```
cat
```

Out[163]:

```
['Street',
 'LandContour',
 'Utilities',
 'Neighborhood',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'MasVnrType',
 'ExterQual',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'HeatingQC',
 'KitchenQual',
 'Functional',
 'GarageCond',
 'PavedDrive',
 'Fence',
 'MiscFeature']
```

In [164…

```
con
```

Out[164]:

```
['MSSubClass',
 'LotArea',
 'OverallQual',
 'OverallCond',
 'YearBuilt',
 'MasVnrArea',
 'BsmtFinSF1',
 'GrLivArea',
 'BsmtFullBath',
 'KitchenAbvGr',
 'TotRmsAbvGrd',
 'Fireplaces',
 'GarageCars',
 'WoodDeckSF',
 'OpenPorchSF',
 'ScreenPorch',
 'PoolArea',
 'YrSold']
```

In [165…

```
num_pipe=Pipeline(steps=[('impute',SimpleImputer(strategy='median')),('scaler',Star
cat_pipe=Pipeline(steps=[('Impute',SimpleImputer(strategy='most_frequent')),('encod
```

In [166…

```
pre=ColumnTransformer([('num_pipe',num_pipe,con),('cat_pipe',cat_pipe,cat)])
```

In [167…

```
x3=pd.DataFrame(pre.fit_transform(x2).toarray(),columns=pre.get_feature_names_out()
x3
```

Out[167]:

	num_pipe__MSSubClass	num_pipe__LotArea	num_pipe__OverallQual	num_pipe__OverallCond
0	0.073375	-0.207142	0.651479	-0.517200
1	-0.872563	-0.091886	-0.071836	2.179628
2	0.073375	0.073480	0.651479	-0.517200
3	0.309859	-0.096897	0.651479	-0.517200
4	0.073375	0.375148	1.374795	-0.517200
...	...	...	...	...
1455	0.073375	-0.260560	-0.071836	-0.517200
1456	-0.872563	0.266407	-0.071836	0.381743
1457	0.309859	-0.147810	0.651479	3.078570
1458	-0.872563	-0.080160	-0.795151	0.381743
1459	-0.872563	-0.058112	-0.795151	0.381743

1460 rows × 145 columns

Spliting data into Train and Test

In [176…

```
x_train,x_test,y_train,y_test=train_test_split(x3,y,test_size=0.2,random_state=18)
```

Model

In [177…

```
lr.fit(x_train,y_train)
```

Out[177]:

```
▾ LinearRegression
LinearRegression()
```

Training Data Evaluation

In [178…

```
y_pred_train=lr.predict(x_train)

mse=mean_squared_error(y_pred_train,y_train)
print('MSE:',mse)

rmse=mse**0.5
print('RMSE:',rmse)

mae=mean_absolute_error(y_pred_train,y_train)
print('MAE:',mae)

R=round(r2_score(y_pred_train,y_train),2)
print('R2:',R)
```

```
MSE: 593642586.3535959
RMSE: 24364.78168081126
MAE: 15167.007705479453
R2: 0.9
```

Testing Data Evaluation

In [179…

```
y_pred=lr.predict(x_test)

mse1=mean_squared_error(y_pred,y_test)
print('Mse:',mse1)

rmse1=mse1**0.5
print('RMSE:',rmse1)

mae1=mean_absolute_error(y_pred,y_test)
print('MAE:',mae1)

R1=r2_score(y_pred,y_test)
print('R2:',R1)
```

```
Mse: 7.396956465369425e+28
RMSE: 271973463142443.8
MAE: 19464887836652.95
R2: -0.0017301716410542678
```

Regularization

In [103…

```
from sklearn.linear_model import Ridge
```

In [180…

```
ra=Ridge()
```

In [181…

```
ra.fit(x_train, y_train)
```

Out[181]:

▾ Ridge

Ridge()

In [182…

```
y_pred_train=ra.predict(x_train)

mse=mean_squared_error(y_pred_train,y_train)
print('MSE:',mse)

rmse=mse**0.5
print('RMSE:',rmse)

mae=mean_absolute_error(y_pred_train,y_train)
print('MAE:',mae)

R=r2_score(y_pred_train,y_train)
print('R2:',R)
```

```
MSE: 659993445.507732
RMSE: 25690.337590380786
MAE: 16031.827815828041
R2: 0.8796527814919366
```

In [183…

```
y_pred=ra.predict(x_test)

mse1=mean_squared_error(y_pred,y_test)
print('Mse:',mse1)

rmse1=mse1**0.5
print('RMSE:',rmse1)

mae1=mean_absolute_error(y_pred,y_test)
print('MAE:',mae1)

R1=r2_score(y_pred,y_test)
print('R2:',R1)
```

```
Mse: 631833640.5118048
RMSE: 25136.30124962312
MAE: 16860.163469639647
R2: 0.8859275614048505
```

Hyperparameter Tuning

In [184…

```
import numpy as np
```

In [185…

```
grid={
    'alpha':np.arange(1,100,0.1)
}
```

In [186…

```
from sklearn.model_selection import GridSearchCV,RandomizedSearchCV
from warnings import filterwarnings
filterwarnings('ignore')
```

In [187…

```
rs=RandomizedSearchCV(ra,param_distributions=grid,cv=6)
rs.fit(x_train,y_train)
```

Out[187]:

```
▸ RandomizedSearchCV
  ▸ estimator: Ridge
        ▸ Ridge
```

In [188…

```
rs.best_params_
```

Out[188]:

```
{'alpha': 5.700000000000005}
```

In [189…

```
ra=Ridge(5.700000000000005)
```

In [190…

```
ra.fit(x_train, y_train)
```

Out[190]:

```
▾ Ridge
Ridge(alpha=5.700000000000005)
```

In [191…

```
y_pred_train=ra.predict(x_train)

mse=mean_squared_error(y_pred_train,y_train)
print('MSE:',mse)
```

```
rmse=mse**0.5
print('RMSE:',rmse)

mae=mean_absolute_error(y_pred_train,y_train)
print('MAE:',mae)

R=round(r2_score(y_pred_train,y_train),2)
print('R2:',R)
```

```
MSE: 751675950.4725881
RMSE: 27416.709329760713
MAE: 16537.012170830672
R2: 0.86
```

In [192...

```
y_pred=ra.predict(x_test)

mse1=mean_squared_error(y_pred,y_test)
print('Mse:',mse1)

rmse1=mse1**0.5
print('RMSE:',rmse1)

mae1=mean_absolute_error(y_pred,y_test)
print('MAE:',mae1)

R1=r2_score(y_pred,y_test)
print('R2:',R1)
```

```
Mse: 672023890.2534853
RMSE: 25923.42358280413
MAE: 16701.223788877203
R2: 0.8725162671547294
```

unseen data

In [194...

```
df2=pd.read_csv('E:/Shaikh Quddus/Classes Recordings/project 1/testing_set.csv')
```

In [195...

```
x_samp=pre.transform(df2).toarray()
```

In [196...

```
x_samp=pd.DataFrame(x_samp,columns=pre.get_feature_names_out())
```

In [197...

```
x_samp
```

Out[197]:

	num_pipe__MSSubClass	num_pipe__LotArea	num_pipe__OverallQual	num_pipe__OverallCond
0	-0.872563	0.110763	-0.795151	0.381743
1	-0.872563	0.375850	-0.071836	0.381743
2	0.073375	0.332053	-0.795151	-0.517200
3	0.073375	-0.054002	-0.071836	0.381743
4	1.492282	-0.552407	1.374795	-0.517200
...	...	...	...	...
1454	2.438219	-0.859988	-1.518467	1.280685
1455	2.438219	-0.864197	-1.518467	-0.517200
1456	-0.872563	0.950423	-0.795151	1.280685
1457	0.664586	-0.007600	-0.795151	-0.517200
1458	0.073375	-0.089180	0.651479	-0.517200

1459 rows × 145 columns

In [198…

```
y_pred1=ra.predict(x_samp)
y_pred1
```

Out[198]:

```
array([121338.04299669, 151894.75801705, 164133.46680722, ...,
       178881.60017101, 107994.00430503, 216021.19734466])
```

In [199…

```
df3=df2[['Id']]
```

In [200…

```
df3['SalePrice']=y_pred1
```

In [201…

```
df3
```

Out[201]:

	Id	SalePrice
0	1461	121338.042997
1	1462	151894.758017
2	1463	164133.466807
3	1464	184005.576569
4	1465	211708.459400
...	...	...
1454	2915	74927.375617
1455	2916	81225.776649
1456	2917	178881.600171
1457	2918	107994.004305
1458	2919	216021.197345

1459 rows × 2 columns

Saving the predicted salesprice in a csv

In [203…

```
df3.to_csv('E:/house_salesprice.csv',index=False)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: