House Pricing - Using Linear Regression

In [1]:

```
# data handling
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# stats testing | hypothesis testing
import scipy.stats as stats

# statistical modelling
import statsmodels.formula.api as smf

# subpackages from sklearn for data handling, variable selection and model evaluation
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_percentage_error

# for modelling - stats and ML
import sklearn
from sklearn.linear_model import LinearRegression
```

In [2]:

```
df = pd.read_csv('Housing.csv')
df.head()
```

Out[2]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	fu
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	

In [3]:

```
df.isna().sum()
```

Out[3]:
```
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
hotwaterheating     0
airconditioning     0
parking             0
prefarea            0
furnishingstatus    0
dtype: int64
```

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

In [5]:
```
def outlier_var(x):
    if ((x.dtype=='float') or (x.dtype=='int')):
        x= x.clip(lower = x.quantile(0.01), upper = x.quantile(0.99))
    else:
        x
    return x
```

In [6]:
```python
df = df.apply(outlier_var)
```

In [7]:
```python
# List of variables to map

varlist =  ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea']

# Defining the map function
def binary_map(x):
    return x.map({'yes': 1, "no": 0})

# Applying the function to the housing list
df[varlist] = df[varlist].apply(binary_map)
```

In [8]:
```python
status = pd.get_dummies(df['furnishingstatus'],drop_first=True)
status
```

Out[8]:

	semi-furnished	unfurnished
0	0	0
1	0	0
2	1	0
3	0	0
4	0	0
...	...	...
540	0	1
541	1	0
542	0	1
543	0	0
544	0	1

545 rows × 2 columns

In [9]:
```python
df
```

Out[9]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea
0	13300000	7420	4	2	3	1	0	0	0	1	2	1
1	12250000	8960	4	4	4	1	0	0	0	1	3	0
2	12250000	9960	3	2	2	1	0	1	0	0	2	1
3	12215000	7500	4	2	2	1	0	1	0	1	3	1
4	11410000	7420	4	1	2	1	1	1	0	1	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...
540	1820000	3000	2	1	1	1	0	1	0	0	2	0
541	1767150	2400	3	1	1	0	0	0	0	0	0	0
542	1750000	3620	2	1	1	1	0	0	0	0	0	0
543	1750000	2910	3	1	1	0	0	0	0	0	0	0
544	1750000	3850	3	1	2	1	0	0	0	0	0	0

545 rows × 13 columns

In [10]:

```
df_new = pd.concat([df,status],axis=1)
df_new
```

Out[10]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea
0	13300000	7420	4	2	3	1	0	0	0	1	2	1
1	12250000	8960	4	4	4	1	0	0	0	1	3	0
2	12250000	9960	3	2	2	1	0	1	0	0	2	1
3	12215000	7500	4	2	2	1	0	1	0	1	3	1
4	11410000	7420	4	1	2	1	1	1	0	1	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...
540	1820000	3000	2	1	1	1	0	1	0	0	2	0
541	1767150	2400	3	1	1	0	0	0	0	0	0	0
542	1750000	3620	2	1	1	1	0	0	0	0	0	0
543	1750000	2910	3	1	1	0	0	0	0	0	0	0
544	1750000	3850	3	1	2	1	0	0	0	0	0	0

545 rows × 15 columns

In [11]:
```
df_new=df_new.drop(['furnishingstatus'],axis=1)
```

In [12]:
```
df_new
```

Out[12]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea
0	13300000	7420	4	2	3	1	0	0	0	1	2	1
1	12250000	8960	4	4	4	1	0	0	0	1	3	0
2	12250000	9960	3	2	2	1	0	1	0	0	2	1
3	12215000	7500	4	2	2	1	0	1	0	1	3	1
4	11410000	7420	4	1	2	1	1	1	0	1	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...
540	1820000	3000	2	1	1	1	0	1	0	0	2	0
541	1767150	2400	3	1	1	0	0	0	0	0	0	0
542	1750000	3620	2	1	1	1	0	0	0	0	0	0
543	1750000	2910	3	1	1	0	0	0	0	0	0	0
544	1750000	3850	3	1	2	1	0	0	0	0	0	0

545 rows × 14 columns

In [13]:

```
df_new = df_new.rename(columns={'semi-furnished':'semi_furnished'})
```

In [14]:

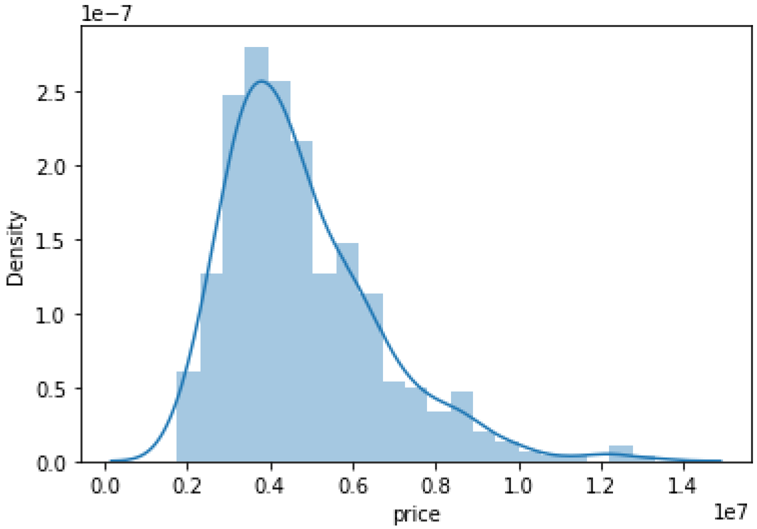
```
import seaborn as sns
# Normality assumption - Checks

sns.distplot(df_new.price)
```

```
C:\Users\gagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fun
ction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[14]:

```
<AxesSubplot:xlabel='price', ylabel='Density'>
```

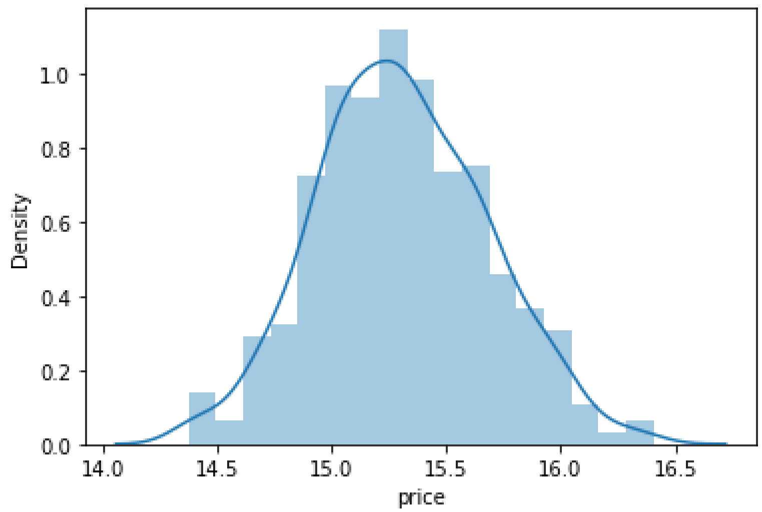


In [15]:
```
df_new.price.skew()
```

Out[15]:
```
1.2122388370279802
```

In [16]:
```
sns.distplot(np.log(df_new.price) )
plt.show()
```

```
C:\Users\gagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fun
ction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



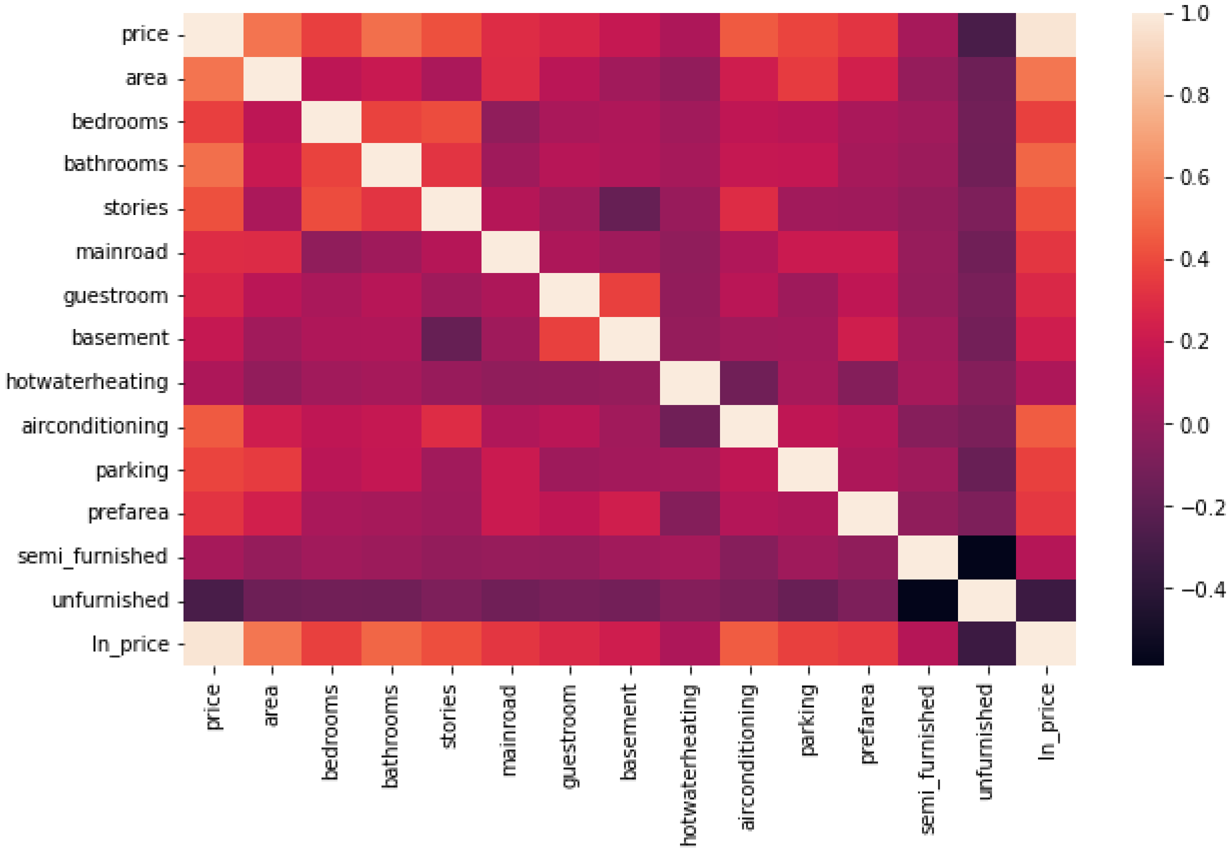
In [17]:
```
np.log(df.price).skew()
```

Out[17]:
```
0.14086257299872787
```

In [18]:
```
df_new['ln_price'] = df_new['price'].apply(lambda x: np.log(x))
df_new['ln_price']
```

Out[18]:
```
0      16.403275
1      16.321036
2      16.321036
3      16.318175
4      16.250001
         ...
540    14.414347
541    14.384879
542    14.375126
543    14.375126
544    14.375126
Name: ln_price, Length: 545, dtype: float64
```

In [19]:
```
plt.figure( figsize=(10, 6) )
sns.heatmap( df_new.corr() )
plt.show()
```



In [20]:
```
df_new.corr().to_excel('corr.xlsx')
```

In [21]:
```
# get only the X variable names
features = df_new.columns.difference(['ln_price'])
```

In [22]:
```
features
```

Out[22]:
```
Index(['airconditioning', 'area', 'basement', 'bathrooms', 'bedrooms',
       'guestroom', 'hotwaterheating', 'mainroad', 'parking', 'prefarea',
       'price', 'semi_furnished', 'stories', 'unfurnished'],
      dtype='object')
```

In [23]:
```
# get the significant variables : bi-variate regression
from sklearn.feature_selection import f_regression
```

In [24]:
```
# bivariate regression
f_score, p_value = f_regression(df_new[features], df_new.ln_price )
```

In [25]:
```
# combine the outputs in the dataframe
significant_variables = pd.DataFrame([features, f_score, p_value]).T
significant_variables.columns = [ 'features', 'f_score', 'p_value' ]
features = list( significant_variables.loc[ significant_variables.p_value <= 0.1, 'features' ] )
```

In [26]:
```
features
```

Out[26]:
```
['airconditioning',
 'area',
 'basement',
 'bathrooms',
 'bedrooms',
 'guestroom',
 'hotwaterheating',
 'mainroad',
 'parking',
 'prefarea',
 'price',
 'semi_furnished',
 'stories',
 'unfurnished']
```

In [27]:
```
features = ['airconditioning',
 #'area',
 'basement',
# 'bathrooms',
 #'bedrooms',
 'guestroom',
 'hotwaterheating',
 #'mainroad',
 'parking',
 'prefarea',
 #'price',
 'semi_furnished',
 'stories',
 'unfurnished'
           ]
```

In [28]:
```
len(features)
```

Out[28]: 9

In [29]:
```
#multicoliniеity
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [30]:
```
vif = pd.concat([pd.Series(features), pd.Series([variance_inflation_factor(df_new[features].values, i ) for i in range(
```

In [31]:
```
vif.columns = ['var', 'vif']
```

In [32]:
```
vif.sort_values(by='vif', ascending=False)
```

Out[32]:

	var	vif
7	stories	3.517470
6	semi_furnished	1.936355
1	basement	1.771836
0	airconditioning	1.733690
4	parking	1.640868
8	unfurnished	1.599363
2	guestroom	1.456931
5	prefarea	1.411160
3	hotwaterheating	1.084918

In [33]:
```
df_new[features].corr()
```

Out[33]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished
airconditioning	1.000000	0.047341	0.138179	-0.130023	0.159173	0.117382	-0.053179	0.293602	-0.094086
basement	0.047341	1.000000	0.372066	0.004385	0.051497	0.228083	0.050284	-0.172394	-0.117935
guestroom	0.138179	0.372066	1.000000	-0.010308	0.037466	0.160897	0.005821	0.043538	-0.099023
hotwaterheating	-0.130023	0.004385	-0.010308	1.000000	0.067864	-0.059411	0.063819	0.018847	-0.059194
parking	0.159173	0.051497	0.037466	0.067864	1.000000	0.091627	0.041327	0.045547	-0.165705
prefarea	0.117382	0.228083	0.160897	-0.059411	0.091627	1.000000	-0.011535	0.044425	-0.081271
semi_furnished	-0.053179	0.050284	0.005821	0.063819	0.041327	-0.011535	1.000000	-0.003648	-0.588405
stories	0.293602	-0.172394	0.043538	0.018847	0.045547	0.044425	-0.003648	1.000000	-0.082972
unfurnished	-0.094086	-0.117935	-0.099023	-0.059194	-0.165705	-0.081271	-0.588405	-0.082972	1.000000

In [34]:

```
df_new = df_new[features + ['ln_price']]
```

In [35]:

```
df_new
```

Out[35]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price
0	1	0	0	0	2	1	0	3	0	16.403275
1	1	0	0	0	3	0	0	4	0	16.321036
2	0	1	0	0	2	1	1	2	0	16.321036
3	1	1	0	0	3	1	0	2	0	16.318175
4	1	1	1	0	2	0	0	2	0	16.250001
...	...	...	...	...	...	...	...	...	...	...
540	0	1	0	0	2	0	0	1	1	14.414347
541	0	0	0	0	0	0	1	1	0	14.384879
542	0	0	0	0	0	0	0	1	1	14.375126
543	0	0	0	0	0	0	0	1	0	14.375126
544	0	0	0	0	0	0	0	2	1	14.375126

545 rows × 10 columns

In [ ]:

In [36]:

```
from sklearn.model_selection import train_test_split
```

In [37]:

```
train, test = train_test_split(df_new, test_size=0.3, random_state=123)
```

In [38]:

```
import statsmodels.formula.api as smf
```

In [39]:

```
train.columns
```

Out[39]:

```
Index(['airconditioning', 'basement', 'guestroom', 'hotwaterheating',
       'parking', 'prefarea', 'semi_furnished', 'stories', 'unfurnished',
       'ln_price'],
      dtype='object')
```

In [40]:

```
model_ols = smf.ols('ln_price ~ airconditioning +guestroom + parking + prefarea + stories+unfurnished', data=train).fit
```

In [41]:

```
print(model_ols.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               ln_price   R-squared:                       0.526
Model:                            OLS   Adj. R-squared:                  0.519
Method:                 Least Squares   F-statistic:                     69.20
Date:                Wed, 19 Apr 2023   Prob (F-statistic):           1.12e-57
Time:                        23:17:08   Log-Likelihood:                -22.971
No. Observations:                 381   AIC:                             59.94
Df Residuals:                     374   BIC:                             87.54
Df Model:                           6
Covariance Type:            nonrobust
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
Intercept          14.9073      0.036    414.475      0.000      14.837      14.978
airconditioning     0.2366      0.030      7.778      0.000       0.177       0.296
guestroom           0.1599      0.036      4.394      0.000       0.088       0.232
parking             0.0971      0.015      6.336      0.000       0.067       0.127
prefarea            0.2113      0.032      6.603      0.000       0.148       0.274
stories             0.1234      0.016      7.811      0.000       0.092       0.154
unfurnished        -0.1600      0.029     -5.528      0.000      -0.217      -0.103
==============================================================================
Omnibus:                       14.165   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               20.560
Skew:                           0.291   Prob(JB):                     3.43e-05
Kurtosis:                       3.978   Cond. No.                         7.68
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

ln_price=14.8706+0.2343*airconditioning+0.1283*guestroom+0.0974*parking+0.1929*prefarea+0.1325*stories-0.1524*unfurnished price = exp(ln_price)

In [42]:
```
#prediction of new house
airconditioning=1
guestroom=1
parking=1
prefarea=0
stories=0
unfurnished=1
```

In [43]:
```
ln_price=14.8706+0.2343*airconditioning+0.1283*guestroom+0.0974*parking+0.1929*prefarea+0.1325*stories-0.1524*unfurnish
```

In [44]:
```
import math
price = math.exp(ln_price)
```

```
price
```

Out[44]:

```
3906685.9912558687
```

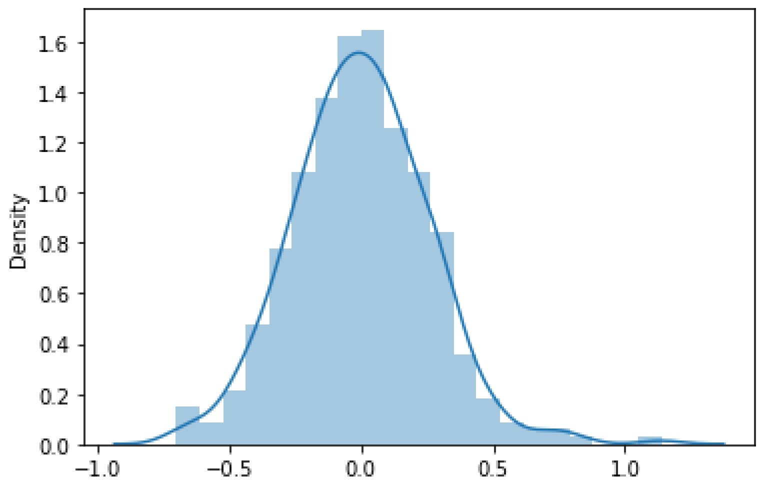
In [45]:

```
sns.distplot(model_ols.resid)
```

```
C:\Users\gagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fun
ction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[45]:

```
<AxesSubplot:ylabel='Density'>
```



In [46]:

```
train_pred = np.exp(model_ols.predict(train))
test_pred = np.exp(model_ols.predict(test))
train_act = np.exp(train.ln_price)
test_act = np.exp(test.ln_price)
```

In [47]:

```
print('train_MAPE:', np.mean(np.abs(train_act- train_pred)/train_act))
```

```
train_MAPE: 0.2032035825307896
```

In [48]:

```
print('test_MAPE:', np.mean(np.abs(test_act- test_pred)/test_act))
```

```
test_MAPE: 0.18536604794143313
```

In [49]:

```
print('train_RMSE:', np.sqrt(np.mean((train_act- train_pred)**2)))
print('test_RMSE:', np.sqrt(np.mean((test_act- test_pred)**2)))
```

```
train_RMSE: 1299254.6027223826
test_RMSE: 1403789.8780242563
```

In [50]:
```python
print('train_corr:', np.corrcoef(train_act, train_pred)[1][0])
print('test_corr:', np.corrcoef(test_act, test_pred)[1][0])
```

```
train_corr: 0.7191509253660459
test_corr: 0.687820126830479
```

In [51]:
```python
df_new.head()
```

Out[51]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price
0	1	0	0	0	2	1	0	3	0	16.403275
1	1	0	0	0	3	0	0	4	0	16.321036
2	0	1	0	0	2	1	1	2	0	16.321036
3	1	1	0	0	3	1	0	2	0	16.318175
4	1	1	1	0	2	0	0	2	0	16.250001

In [52]:
```python
train
```

Out[52]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price
248	0	1	1	0	0	0	1	1	0	15.329098
298	0	0	0	1	2	0	1	1	0	15.250595
148	0	0	0	0	0	1	1	3	0	15.538277
120	0	1	1	0	2	1	0	1	0	15.598902
494	0	0	0	0	0	0	0	1	1	14.819812
...	...	...	...	...	...	...	...	...	...	...
98	1	0	0	0	0	1	0	3	1	15.654948
322	1	1	0	0	1	0	0	1	0	15.208035
382	0	1	0	0	0	0	0	2	0	15.088076
365	0	0	0	0	0	0	0	1	0	15.124654
510	0	0	0	0	0	0	0	1	1	14.739769

381 rows × 10 columns

In [53]:

```
train['pre_ln_price'] = model_ols.predict(train)
```

In [54]:

```
train.head()
```

Out[54]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price	pre_ln_price
248	0	1	1	0	0	0	1	1	0	15.329098	15.190660
298	0	0	0	1	2	0	1	1	0	15.250595	15.224967
148	0	0	0	0	0	1	1	3	0	15.538277	15.488882
120	0	1	1	0	2	1	0	1	0	15.598902	15.596227
494	0	0	0	0	0	0	0	1	1	14.819812	14.870753

In [56]:

```
train['price_actual'] = np.exp(train.ln_price)
train['price_pred'] = np.exp(train.pre_ln_price)
```

In [57]:
```
train.head()
```

Out[57]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price	pre_ln_price
248	0	1	1	0	0	0	1	1	0	15.329098	15.190660
298	0	0	0	1	2	0	1	1	0	15.250595	15.224967
148	0	0	0	0	0	1	1	3	0	15.538277	15.488882
120	0	1	1	0	2	1	0	1	0	15.598902	15.596227
494	0	0	0	0	0	0	0	1	1	14.819812	14.870753

In [58]:
```
train['deciles'] = pd.qcut(train.price_pred, q=10, labels=False)
```

In [59]:
```
train.head()
```

Out[59]:

	airconditioning	basement	guestroom	hotwaterheating	parking	prefarea	semi_furnished	stories	unfurnished	ln_price	pre_ln_price
248	0	1	1	0	0	0	1	1	0	15.329098	15.190660
298	0	0	0	1	2	0	1	1	0	15.250595	15.224967
148	0	0	0	0	0	1	1	3	0	15.538277	15.488882
120	0	1	1	0	2	1	0	1	0	15.598902	15.596227
494	0	0	0	0	0	0	0	1	1	14.819812	14.870753

In [60]:
```
train[['deciles', 'price_pred','price_actual' ]].groupby('deciles').agg(np.mean)
```

Out[60]:

deciles	price_pred	price_actual
0	3.047189e+06	3.168073e+06
1	3.370976e+06	3.589178e+06
2	3.639910e+06	3.706952e+06
3	3.846829e+06	4.110909e+06
4	4.141588e+06	4.152235e+06
5	4.547649e+06	4.829243e+06
6	4.906281e+06	4.846686e+06
7	5.335350e+06	5.338221e+06
8	6.010954e+06	6.382931e+06
9	7.443506e+06	7.691987e+06

In [63]:

```
test['pre_ln_price'] = model_ols.predict(test)
test['price_actual'] = np.exp(test.ln_price)
test['price_pred'] = np.exp(test.pre_ln_price)

test['deciles'] = pd.qcut(test.price_pred, q=10, labels=False)

test_deciles = test[['deciles', 'price_pred','price_actual' ]].groupby('deciles').agg(np.mean)
test_deciles
```

Out[63]:

deciles	price_pred	price_actual
0	2.921506e+06	2.813222e+06
1	3.323659e+06	3.548391e+06
2	3.580293e+06	4.007500e+06
3	3.787762e+06	4.093895e+06
4	4.146669e+06	4.577403e+06
5	4.529378e+06	5.716200e+06
6	4.837494e+06	4.948067e+06
7	5.282517e+06	5.370312e+06
8	5.918293e+06	6.300412e+06
9	7.566604e+06	7.436809e+06

In [64]:

```
test_deciles.to_csv('test_deciles.csv')
```

In [ ]: