- Step 0 : Load the data

In [401]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [402]:
```
house_data = pd.read_csv(r"C:\Users\tlili\OneDrive\Bureau
```

- Step 1 : Explore the data

In [403]:
```
house_data.head()
```

Out[403]:

	id	date	price	bedrooms	bath
0	7129300520	20141013T000000	221900.0	3	
1	6414100192	20141209T000000	538000.0	3	
2	5631500400	20150225T000000	180000.0	2	
3	2487200875	20141209T000000	604000.0	4	
4	1954400510	20150218T000000	510000.0	3	

5 rows × 21 columns

In [404]:
```
house_data.columns
```

Out[404]:
```
Index(['id', 'date', 'price', 'bedrooms', 'bathroo
ms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'view',
'condition', 'grade',
       'sqft_above', 'sqft_basement', 'yr_built',
'yr_renovated', 'zipcode',
       'lat', 'long', 'sqft_living15', 'sqft_lot1
5'],
      dtype='object')
```

In [405]:
```
house_data.dtypes
```

Out[405]:
```
id                int64
date             object
price           float64
bedrooms          int64
bathrooms       float64
sqft_living       int64
sqft_lot          int64
floors          float64
waterfront        int64
view              int64
condition         int64
grade             int64
sqft_above        int64
sqft_basement     int64
yr_built          int64
yr_renovated      int64
zipcode           int64
lat             float64
long            float64
sqft_living15     int64
sqft_lot15        int64
dtype: object
```

In [406]:
```
house_data.describe()
```

Out[406]:

	id	price	bedrooms	bathroc
count	2.161300e+04	2.161300e+04	21613.000000	21613.000
mean	4.580302e+09	5.400881e+05	3.370842	2.114
std	2.876566e+09	3.671272e+05	0.930062	0.770
min	1.000102e+06	7.500000e+04	0.000000	0.000
25%	2.123049e+09	3.219500e+05	3.000000	1.750
50%	3.904930e+09	4.500000e+05	3.000000	2.250
75%	7.308900e+09	6.450000e+05	4.000000	2.500
max	9.900000e+09	7.700000e+06	33.000000	8.000

In [407]:
```
house_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             21613 non-null  int64  
 1   date           21613 non-null  object 
 2   price          21613 non-null  float64
 3   bedrooms       21613 non-null  int64  
 4   bathrooms      21613 non-null  float64
 5   sqft_living    21613 non-null  int64  
 6   sqft_lot       21613 non-null  int64  
 7   floors         21613 non-null  float64
 8   waterfront     21613 non-null  int64  
 9   view           21613 non-null  int64  
 10  condition      21613 non-null  int64  
 11  grade          21613 non-null  int64  
 12  sqft_above     21613 non-null  int64  
 13  sqft_basement  21613 non-null  int64  
 14  yr_built       21613 non-null  int64  
 15  yr_renovated   21613 non-null  int64  
 16  zipcode        21613 non-null  int64  
 17  lat            21613 non-null  float64
 18  long           21613 non-null  float64
 19  sqft_living15  21613 non-null  int64  
 20  sqft_lot15     21613 non-null  int64  
dtypes: float64(5), int64(15), object(1)
memory usage: 3.5+ MB
```

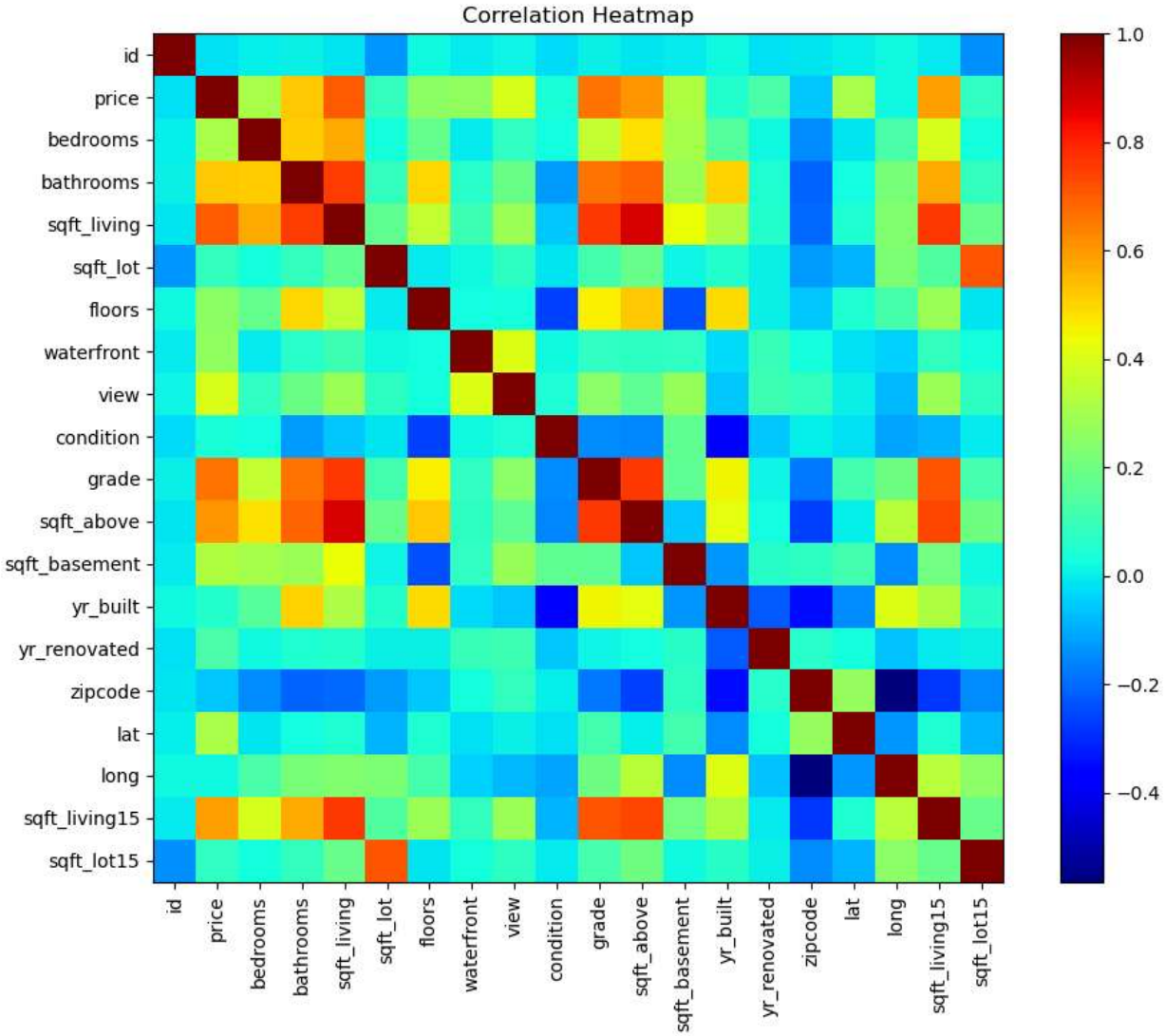
In [408]:
```
house_data.isna().sum()
```

Out[408]:

```
id               0
date             0
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated     0
zipcode          0
lat              0
long             0
sqft_living15    0
sqft_lot15       0
dtype: int64
```

In [409]:

```
correlation = house_data.select_dtypes(include='number').
# Plot heatmap
plt.figure(figsize=(10, 8))
plt.imshow(correlation, cmap='jet', interpolation='neares
plt.colorbar()
plt.xticks(range(len(correlation.columns)), correlation.c
plt.yticks(range(len(correlation.columns)), correlation.c
plt.title("Correlation Heatmap")
plt.tight_layout()
plt.show()
```



- Step 2 : Data cleaning

we will remove the following columns :

- id, date , view -> irrelevant for the model
- sqft_living,sqft_above,sqft_basement,sqft living15,sqft_lot15,lat,long -> redandant with the sqft_lot feature they are all about the house size
- condition -> redandant with grade feature
- zipcode -> categorical feature without described with numerical data , very difficult for the model to learn from it

- yr_renovated -> almost all houses haven't been renovated the feature is not very informative

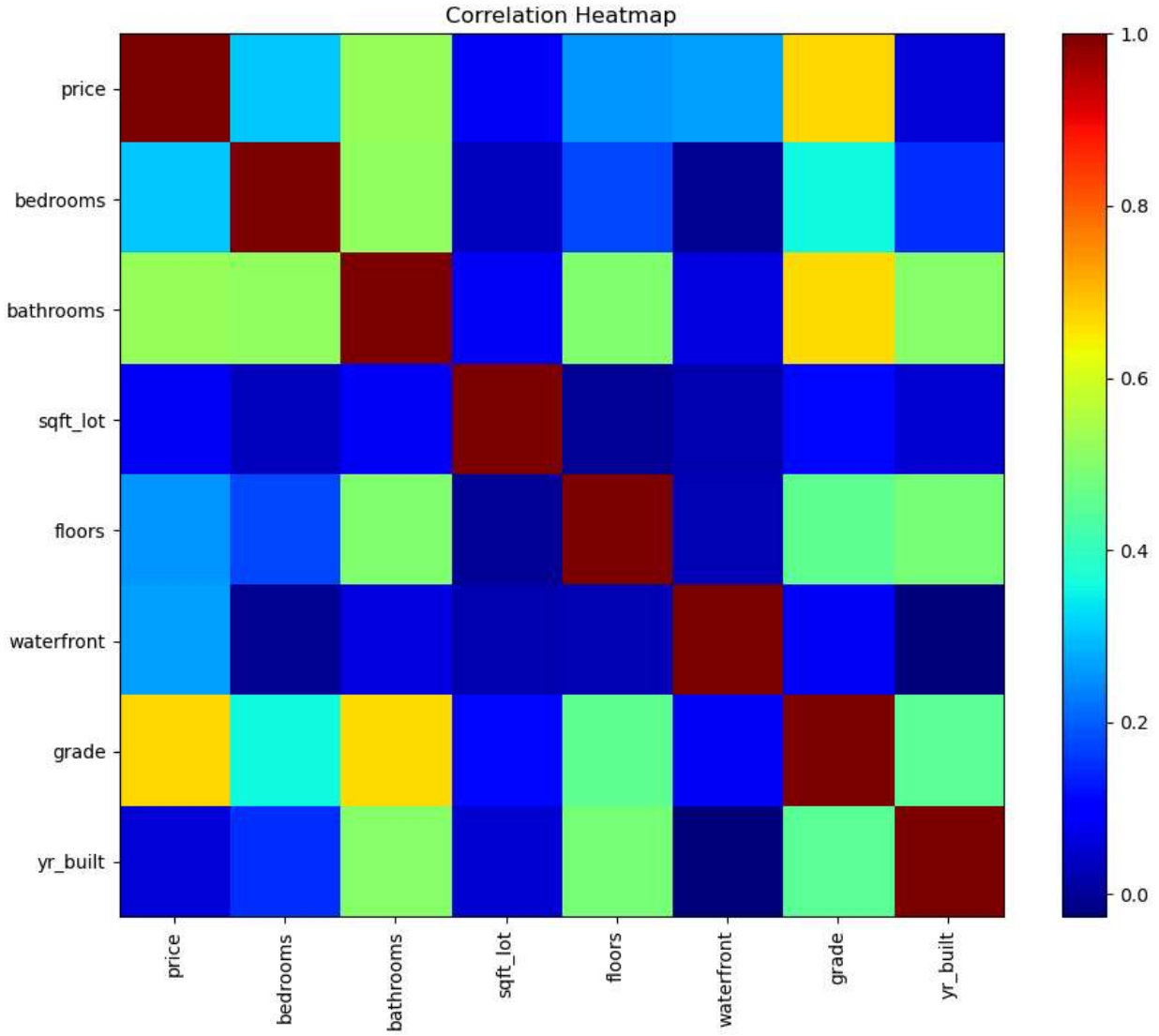
In [410]:
```
house_data_cleaned = house_data.drop(columns=["id","sqft_
```

In [411]:
```
house_data_cleaned.head()
```

Out[411]:

	price	bedrooms	bathrooms	sqft_lot	floors	waterfr
0	221900.0	3	1.00	5650	1.0	
1	538000.0	3	2.25	7242	2.0	
2	180000.0	2	1.00	10000	1.0	
3	604000.0	4	3.00	5000	1.0	
4	510000.0	3	2.00	8080	1.0	

In [412]:
```
correlation = house_data_cleaned.select_dtypes(include='nu
# Plot heatmap
plt.figure(figsize=(10, 8))
plt.imshow(correlation, cmap='jet', interpolation='neares
plt.colorbar()
plt.xticks(range(len(correlation.columns)), correlation.c
plt.yticks(range(len(correlation.columns)), correlation.c
plt.title("Correlation Heatmap")
plt.tight_layout()
plt.show()
```



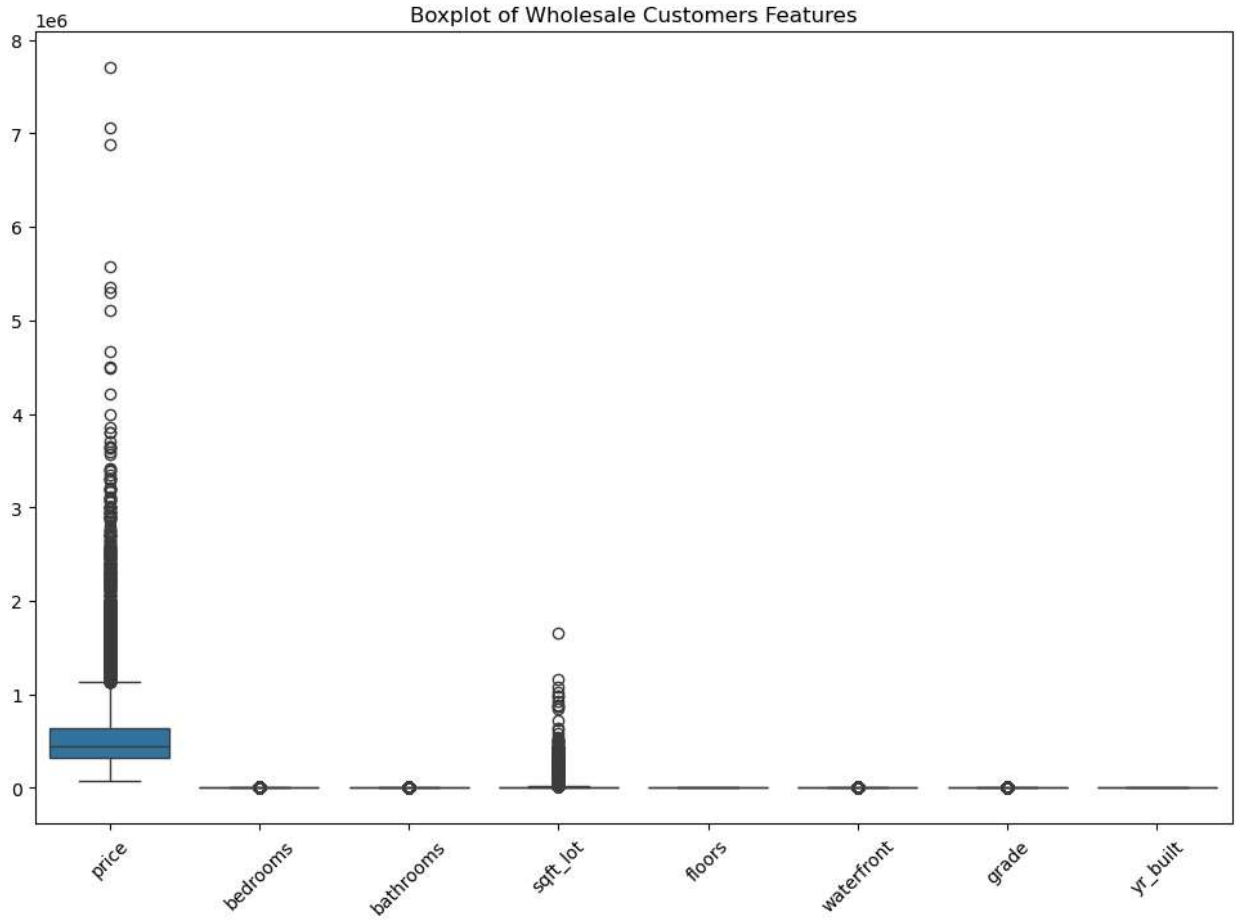
- there is no more highly correlated features in the dataset

In [413]:
```
#identify outliers
Q1 = house_data_cleaned.quantile(0.25)
Q3 = house_data_cleaned.quantile(0.75)
IQR = Q3 - Q1

outliers = ((house_data_cleaned < (Q1 - 1.5 * IQR)) | (ho
print(outliers.sum())
```

```
price         1146
bedrooms       546
bathrooms      571
sqft_lot      2425
floors           0
waterfront     163
grade         1911
yr_built         0
dtype: int64
```

In [414]:
```
#visualize outliers
plt.figure(figsize=(12,8))
sns.boxplot(data=house_data_cleaned)
plt.title("Boxplot of Wholesale Customers Features")
plt.xticks(rotation=45)
plt.show()
```

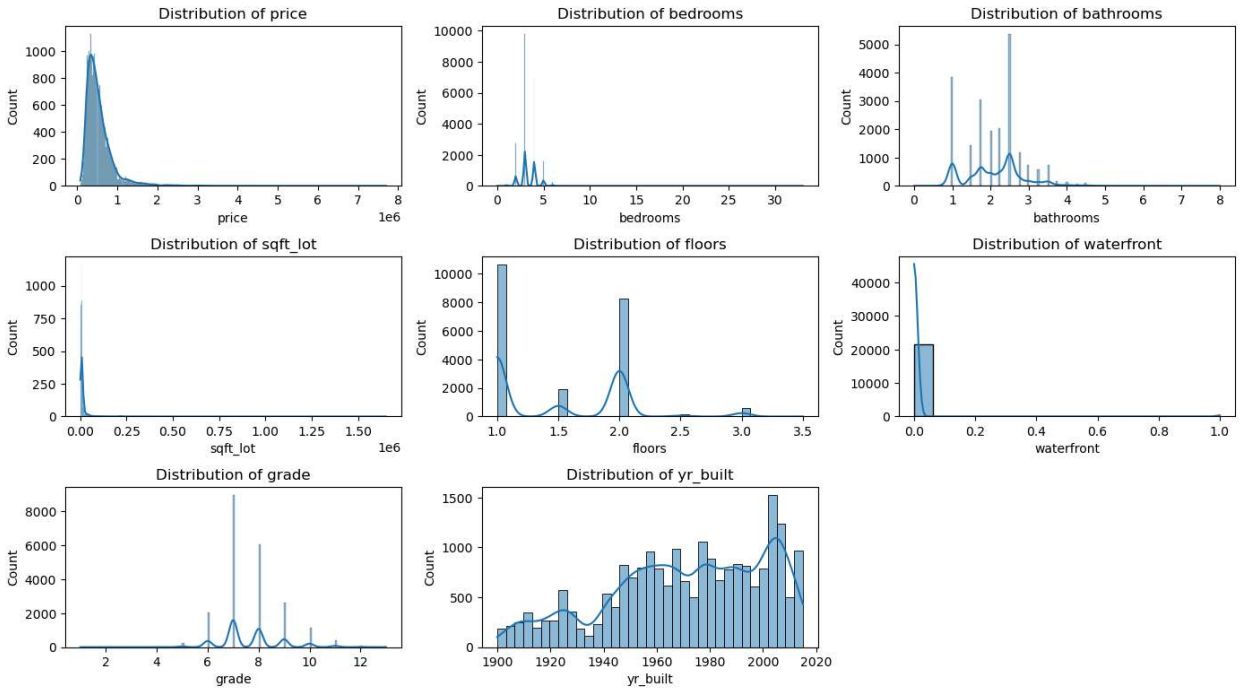


- the sqft_lot and price features shows a high number of outliers but we will keep them for the moment

In [415]:
```
#skewness
house_data_cleaned.skew()
```

Out[415]:
```
price          4.024069
bedrooms       1.974300
bathrooms      0.511108
sqft_lot      13.060019
floors         0.616177
waterfront    11.385108
grade          0.771103
yr_built      -0.469805
dtype: float64
```

In [416]:
```
#visualize the skewness
plt.figure(figsize=(14,10))
for i, col in enumerate(house_data_cleaned.columns, 1):
    plt.subplot(4, 3, i)
    sns.histplot(house_data_cleaned[col], kde=True)
    plt.title(f"Distribution of {col}")
plt.tight_layout()
plt.show()
```



- the data is skewed , we will try to fix it later depending on the model performance

- Step 3 : Machine learning Models

- Step 3.1 : linear regression model

In [417]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [418]:
```
#multi-linear regression model
X = house_data_cleaned.drop('price', axis=1)
y = house_data_cleaned['price']

#split data
X_train, X_test, y_train, y_test = train_test_split(X, y,

#model fitting
model_multi = LinearRegression()
model_multi.fit(X_train, y_train)

#prediction
y_pred = model_multi.predict(X_test)

#evaluation
print("MSE:", mean_squared_error(y_test, y_pred))
print("R² Score:", r2_score(y_test, y_pred))
```

```
MSE: 60587216642.93024
R² Score: 0.5992291687095985
```

- the R score is very low at this stage , to improve it we will replace the year of built by a new column called age of the house that is easier to understand by the model

In [419]:
```
reference_date = pd.to_datetime("2015-01-01")
# compute house age in years
house_data_cleaned["house_age"] = reference_date.year - h
```

In [420]:
```
house_data_cleaned.head()
```

Out[420]:

	price	bedrooms	bathrooms	sqft_lot	floors	waterfr
0	221900.0	3	1.00	5650	1.0	
1	538000.0	3	2.25	7242	2.0	
2	180000.0	2	1.00	10000	1.0	
3	604000.0	4	3.00	5000	1.0	
4	510000.0	3	2.00	8080	1.0	

In [421]:
```
#multi-linear regression model after creating the house ag
X1 = house_data_cleaned.drop('price', axis=1)
y1 = house_data_cleaned['price']

#split data
X1_train, X1_test, y1_train, y1_test = train_test_split(X

#model fitting
model_multi1 = LinearRegression()
model_multi1.fit(X1_train, y1_train)

#prediction
y1_pred = model_multi1.predict(X1_test)

#evaluation
print("MSE1:", mean_squared_error(y1_test, y1_pred))
print("R² Score1:", r2_score(y1_test, y1_pred))
```

```
MSE1: 60587216642.9378
R² Score1: 0.5992291687095486
```

- creating a new feature "age" slightly decreased the R score by −0.0000000000000499 that is might be du to the presence of the yr _built feature that creates redendancy let's remove it and see if the model improves

In [422]:
```
#let's drop the yr_built to avoide redundancy
house_data_cleaned2 = house_data_cleaned.drop(columns=["y
```

In [423]:
```
#multi-linear regression model after replacing the year o
X2 = house_data_cleaned2.drop('price', axis=1)
y2 = house_data_cleaned2['price']

#split data
X2_train, X2_test, y2_train, y2_test = train_test_split(X

#model fitting
model_multi2 = LinearRegression()
model_multi2.fit(X2_train, y2_train)

#prediction
y2_pred = model_multi2.predict(X2_test)

#evaluation
print("MSE2:", mean_squared_error(y2_test, y2_pred))
print("R² Score2:", r2_score(y2_test, y2_pred))
```

```
MSE2: 60587216642.93024
R² Score2: 0.5992291687095985
```

- Droping the year of built slightly increased the R score by 0.0000239089983574 , the score is still low we will try to create a statmodel and analyse the p value to select the most significant features

In [424]:
```
#what are the features that impacts the price the most

import statsmodels.api as sm

# Split predictors and target
X_stat = house_data_cleaned.drop('price', axis=1)
y_stat = house_data_cleaned['price']

# Add intercept (required by statsmodels)
X_sm = sm.add_constant(X_stat)

#fit the model
model_stat = sm.OLS(y_stat, X_sm).fit()
print(model_stat.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:
0.596
Model:                            OLS   Adj. R-squa
red:                  0.596
Method:                 Least Squares   F-statisti
c:                     4562.
Date:                Thu, 27 Nov 2025   Prob (F-sta
tistic):               0.00
Time:                        12:14:28   Log-Likelih
ood:            -2.9780e+05
No. Observations:               21613   AIC:
5.956e+05
Df Residuals:                   21605   BIC:
5.957e+05
Df Model:                           7
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>
|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.5499      0.030     51.098      0.
000       1.490       1.609
bedrooms   -1321.5960   2025.266     -0.653      0.
514   -5291.267    2648.075
bathrooms   1.332e+05   3291.657     40.474      0.
000    1.27e+05     1.4e+05
sqft_lot       0.0825      0.039      2.133      0.
033       0.007       0.158
floors      2844.4619   3640.212      0.781      0.
435   -4290.622    9979.546
waterfront  7.924e+05   1.85e+04     42.846      0.
000    7.56e+05    8.29e+05
grade        1.96e+05   1869.674    104.822      0.
000    1.92e+05       2e+05
yr_built    -719.6087      6.843   -105.158      0.
```

```
000    -733.022    -706.196
house_age   3842.6353     63.875     60.159      0.
000    3717.436    3967.835
==============================================
============================
Omnibus:                       18207.421   Durbin-Wats
on:                   1.974
Prob(Omnibus):                     0.000   Jarque-Bera
(JB):          1765999.506
Skew:                              3.524   Prob(JB):
0.00
Kurtosis:                         46.719   Cond. No.
1.91e+20
==============================================
============================

Notes:
[1] Standard Errors assume that the covariance matr
ix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.15e-27. This might
indicate that there are
strong multicollinearity problems or that the desig
n matrix is singular.
```

- the bedrooms , and the floors have a p value > 0,05 meaning they don't have significant impact we will remove them

In [432]:
```
house_data_cleaned3 = house_data_cleaned2.drop(columns=["
```

In [433]:
```
house_data_cleaned3.head()
```

Out[433]:

	price	bathrooms	sqft_lot	waterfront	grade	house_
0	221900.0	1.00	5650	0	7	
1	538000.0	2.25	7242	0	7	
2	180000.0	1.00	10000	0	6	
3	604000.0	3.00	5000	0	7	
4	510000.0	2.00	8080	0	8	

In [434]:

```
#multi-linear regression model after removing the bedrooms
X3 = house_data_cleaned3.drop('price', axis=1)
y3 = house_data_cleaned3['price']

#split data
X3_train, X3_test, y3_train, y3_test = train_test_split(X

#model fitting
model_multi3 = LinearRegression()
model_multi3.fit(X3_train, y3_train)

#prediction
y3_pred = model_multi3.predict(X3_test)

#evaluation
print("MSE3:", mean_squared_error(y3_test, y3_pred))
print("R² Score3:", r2_score(y3_test, y3_pred))
```

```
MSE3: 60583602159.16521
R² Score3: 0.5992530777079559
```

- The R score after removing the insignificant features flours and bedrooms stayed the same but the MSE (mean squared error) decreased a bit that means the model is slightly better ,

the score is still very low we already know that the data is skewed :

- sqft_lot is highly right skewed (skewness = 13.060019)
- price is right skewed (skewness = 4.021151) good feature for a log transform
- bathrooms is right skewed (skewness = 1.104209)
- waterfront is highly right skewed (skewness = 5.222993)
- grade is right skewed (skewness = 0.667137)
- house age is left skewed (skewness = -0.601305)

the skewness might be the reason of the model low performance we will try to fix it , the sqft_lot and price are good features for a log transform because they containes many outlieres and are right skewed

In [436]:
```
import numpy as np

house_data_cleaned4 = house_data_cleaned3.copy()

house_data_cleaned4['log_price'] = np.log(house_data_clea
house_data_cleaned4['log_sqft_lot'] = np.log(house_data_c

house_data_cleaned4.head()
```

Out[436]:

	price	bathrooms	sqft_lot	waterfront	grade	house_
0	221900.0	1.00	5650	0	7	
1	538000.0	2.25	7242	0	7	
2	180000.0	1.00	10000	0	6	
3	604000.0	3.00	5000	0	7	
4	510000.0	2.00	8080	0	8	

In [443]:
```
house_data_cleaned4.drop(columns=["price","sqft_lot"])
```

```
---------------------------------------------------
-----------------------
KeyError                                  Traceback
(most recent call last)
Cell In[443], line 1
----> 1 house_data_cleaned4.drop(columns=["pric
e","sqft_lot"])

File c:\Users\tlili\anaconda3\Lib\site-packages\pan
das\core\frame.py:5581, in DataFrame.drop(self, lab
els, axis, index, columns, level, inplace, errors)
   5433 def drop(
   5434     self,
   5435     labels: IndexLabel | None = None,
   (...)
   5442     errors: IgnoreRaise = "raise",
   5443 ) -> DataFrame | None:
   5444     """
   5445     Drop specified labels from rows or colu
mns.
   5446
   (...)
   5579             weight  1.0     0.8
   5580     """
-> 5581     return super().drop(
   5582         labels=labels,
   5583         axis=axis,
   5584         index=index,
   5585         columns=columns,
   5586         level=level,
   5587         inplace=inplace,
   5588         errors=errors,
   5589     )

File c:\Users\tlili\anaconda3\Lib\site-packages\pan
das\core\generic.py:4788, in NDFrame.drop(self, lab
els, axis, index, columns, level, inplace, errors)
   4786 for axis, labels in axes.items():
   4787     if labels is not None:
-> 4788         obj = obj._drop_axis(labels, axis,
```

```
level=level, errors=errors)
   4790 if inplace:
   4791     self._update_inplace(obj)

File c:\Users\tlili\anaconda3\Lib\site-packages\pan
das\core\generic.py:4830, in NDFrame._drop_axis(sel
f, labels, axis, level, errors, only_slice)
   4828         new_axis = axis.drop(labels, level=
level, errors=errors)
   4829     else:
-> 4830         new_axis = axis.drop(labels, errors
=errors)
   4831     indexer = axis.get_indexer(new_axis)
   4833 # Case for non-unique axis
   4834 else:

File c:\Users\tlili\anaconda3\Lib\site-packages\pan
das\core\indexes\base.py:7070, in Index.drop(self,
labels, errors)
   7068 if mask.any():
   7069     if errors != "ignore":
-> 7070         raise KeyError(f"{labels[mask].toli
st()} not found in axis")
   7071     indexer = indexer[~mask]
   7072 return self.delete(indexer)

KeyError: "['price', 'sqft_lot'] not found in axis"
```

In [445]:

```
import numpy as np


y4 = house_data_cleaned4["log_price"]
X4 = house_data_cleaned4.drop("log_price", axis=1)

#train split
X4_train, X4_test, y4_train, y4_test = train_test_split(X4

#fit"
model_multi4 = LinearRegression()
model_multi4.fit(X4_train, y4_train)
```

```
#predict
y4_pred_log = model_multi4.predict(X4_test)

#convert back to real price
y4_pred_real = np.exp(y4_pred_log)
y4_test_real = np.exp(y4_test)

# evaluate
print("MSE4:", mean_squared_error(y4_test_real, y4_pred_r
print("R² Score4:", r2_score(y4_test_real, y4_pred_real))
```

```
MSE4: 55131609059.838264
R² Score4: 0.6353167876401031
```

- after log trasforming the sqft_lot and price features the R score improved significantly to reach 0.63, the MSE also decreased a lot that means the model is better
- we want to improve the performance more by applying ridge and lasso regression

In [451]:
```
from sklearn.linear_model import Ridge, Lasso

# Target
y5 = house_data_cleaned4['log_price']

# Features
X5 = house_data_cleaned4.drop('log_price', axis=1)

# Split
X5_train, X5_test, y5_train, y5_test = train_test_split(X5
```

In [459]:
```
# Ridge
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X5_train, y5_train)

# Predict in log scale
y5_pred_log_ridge = ridge_model.predict(X5_test)
```

```
# Convert back to real prices
y5_pred_ridge = np.exp(y5_pred_log_ridge)
y5_test_real = np.exp(y5_test)

# Evaluate
MSE_ridge5 = mean_squared_error(y5_test_real, y5_pred_ridg
R2_ridge5 = r2_score(y5_test_real, y5_pred_ridge)


print("MSE 5:", MSE_ridge5)
print("R² score 5:", R2_ridge5)
```

```
MSE 5: 55137703963.52317
R² score 5: 0.6352764712210335
```

- ridge regression decreased slightly the Rscore by 0.0000403164190696 , think that we rached the maximum performance with this dataset using the linear regression which is a basic model , lets work with an other model using the same dataset

In [454]:
```
house_data_cleaned4.to_csv("house_data_cleaned4.csv", ind
```

- Step 3.2 : XGBoost Regressor

In [458]:
```
from xgboost import XGBRegressor

X6 = house_data_cleaned4[["bathrooms", "waterfront", "gra
y6 = house_data_cleaned4["log_price"]

X6_train, X6_test, y6_train, y6_test = train_test_split(X

model6 = XGBRegressor(
    n_estimators=500,
    learning_rate=0.05,
    max_depth=6,
```

```
    subsample=0.8,
    colsample_bytree=0.8,
    objective='reg:squarederror'
)

model6.fit(X6_train, y6_train)

# Predictions in log space
y_pred_log6 = model6.predict(X6_test)

# Convert back to real prices
y_pred6 = np.exp(y_pred_log6)
y_test_real6 = np.exp(y6_test)

print("MSE 6:", mean_squared_error(y_test_real6, y_pred6))
print("R² score 6:", r2_score(y_test_real6, y_pred6))
```

```
MSE 6: 43951175282.48184
R² score 6: 0.7092728461523448
```

- with the exact same dataset the model XGBoost regressor reached 0,70 R score that means it increased by 0.0739963749313113 compared to the linear regression model . let's try to improve the xgboost by changing its hyperparameters

- to find the optimal hyperparameters we will use either the Random search, Grid search or the Optuna , calculation will be done using google collab

- running the Randomsearch in google collab gave the following result : Best Hyperparameters: { 'subsample': 0.7 , 'reg_lambda': 1.0, 'reg_alpha': 0, 'n_estimators': 300, 'max_depth': 6, 'learning_rate': 0.03,'gamma':0,3, 'colsample_bytree': 1.0}

In [462]:
```
from xgboost import XGBRegressor

X7 = house_data_cleaned4[["bathrooms", "waterfront", "gra
y7 = house_data_cleaned4["log_price"]

X7_train, X7_test, y7_train, y7_test = train_test_split(X

model7 = XGBRegressor(
    n_estimators=300,
    learning_rate=0.03,
    max_depth=6,
    subsample=0.7,
    colsample_bytree=1.0,
    reg_lambda=1.0,
    reg_alpha= 0,
    gamma= 0.3,
    objective='reg:squarederror'
)

model7.fit(X7_train, y7_train)

# Predictions in log space
y_pred_log7 = model7.predict(X7_test)

# Convert back to real prices
y_pred7 = np.exp(y_pred_log7)
y_test_real7 = np.exp(y7_test)

print("MSE 7:", mean_squared_error(y_test_real7, y_pred7))
print("R² score 7:", r2_score(y_test_real7, y_pred7))
```

```
MSE 7: 46449543694.64782
R² score 7: 0.6927467001946186
```

- using the randomsearch hyperparameters the model performance got worse with a decrease of the rscore to reach 0,69 . we will try the optuna method next (calculate using google collab)

- the hyperparameters found using Optuna are : Best Hyperparameters: {'colsample_bytree': 0.83, 'gamma': 0.2, 'learning_rate': 0.040, 'max_depth': 6, 'n_estimators': 520, 'reg_alpha': 0.35, 'reg_lambda': 1.21, 'subsample': 0.85}

In [ ]:
```
from xgboost import XGBRegressor
X8 = house_data_cleaned4[["bathrooms", "waterfront", "gra
y8 = house_data_cleaned4["log_price"]
X8_train, X8_test, y8_train, y8_test = train_test_split(X

model8  = XGBRegressor(
    n_estimators=520,
    learning_rate=0.040,
    max_depth=6,
    subsample=0.85,
    colsample_bytree=0.83,
    reg_lambda=1.21,
    reg_alpha= 0.35,
    gamma= 0.2,
    objective='reg:squarederror'
)

model8.fit(X8_train, y8_train)

# Predictions in log space
y_pred_log8 = model8.predict(X8_test)

# Convert back to real prices
y_pred8 = np.exp(y_pred_log8 )
y_test_real8 = np.exp(y8_test)

print("MSE 8:", mean_squared_error(y_test_real8, y_pred8))
print("R² score 8:", r2_score(y_test_real8, y_pred8))
```

```
MSE 8: 45758722217.38722
R² score 8: 0.6973163291205795
```

- using the optuna hyperparameters the model performance got worse with a decrease of the rscore to reach 0,69

- Step 3.3: CatBoost Regressor

In [476]:
```
pip install catboost
```

```
Requirement already satisfied: catboost in c:\users
\tlili\anaconda3\lib\site-packages (1.2.8)
Requirement already satisfied: graphviz in c:\users
\tlili\anaconda3\lib\site-packages (from catboost)
(0.21)
Requirement already satisfied: matplotlib in c:\use
rs\tlili\anaconda3\lib\site-packages (from catboos
t) (3.10.0)
Requirement already satisfied: numpy<3.0,>=1.16.0 i
n c:\users\tlili\anaconda3\lib\site-packages (from
catboost) (2.1.3)
Requirement already satisfied: pandas>=0.24 in c:\u
sers\tlili\anaconda3\lib\site-packages (from catboo
st) (2.2.3)
Requirement already satisfied: scipy in c:\users\tl
ili\anaconda3\lib\site-packages (from catboost) (1.
16.3)
Requirement already satisfied: plotly in c:\users\t
lili\anaconda3\lib\site-packages (from catboost)
(5.24.1)
Requirement already satisfied: six in c:\users\tlil
i\anaconda3\lib\site-packages (from catboost) (1.1
7.0)
Requirement already satisfied: python-dateutil>=2.
8.2 in c:\users\tlili\anaconda3\lib\site-packages
(from pandas>=0.24->catboost) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\u
sers\tlili\anaconda3\lib\site-packages (from pandas
>=0.24->catboost) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\tlili\anaconda3\lib\site-packages (from pa
ndas>=0.24->catboost) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\tlili\anaconda3\lib\site-packages (from ma
tplotlib->catboost) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\u
sers\tlili\anaconda3\lib\site-packages (from matplo
tlib->catboost) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\tlili\anaconda3\lib\site-packages (from ma
```

```
tplotlib->catboost) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in
c:\users\tlili\anaconda3\lib\site-packages (from ma
tplotlib->catboost) (1.4.8)
Requirement already satisfied: packaging>=20.0 in
c:\users\tlili\anaconda3\lib\site-packages (from ma
tplotlib->catboost) (24.2)
Requirement already satisfied: pillow>=8 in c:\user
s\tlili\anaconda3\lib\site-packages (from matplotli
b->catboost) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\tlili\anaconda3\lib\site-packages (from ma
tplotlib->catboost) (3.2.0)
Requirement already satisfied: tenacity>=6.2.0 in
c:\users\tlili\anaconda3\lib\site-packages (from pl
otly->catboost) (9.0.0)
Note: you may need to restart the kernel to use upd
ated packages.
```

In [477]:

```
from catboost import CatBoostRegressor

# Features and target
X9 = house_data_cleaned4[["bathrooms", "waterfront", "gra
y9 = house_data_cleaned4["log_price"]

# Split data
X9_train, X9_test, y9_train, y9_test = train_test_split(X

# CatBoost Regressor
model9_cat = CatBoostRegressor(
    iterations=500,
    learning_rate=0.05,
    depth=6,
    l2_leaf_reg=1.21,
    random_seed=42,
    verbose=0
)

# Fit model
model9_cat.fit(X9_train, y9_train)
```

```
# Predictions in log space
y_pred_log9_cat = model9_cat.predict(X9_test)

# Convert back to real prices
y_pred9_cat = np.exp(y_pred_log9_cat)
y_test_real9 = np.exp(y9_test)

# Metrics
print("MSE 9:", mean_squared_error(y_test_real9, y_pred9_
print("R² score 9:", r2_score(y_test_real9, y_pred9_cat))
```

```
MSE 9: 44745784138.97843
R² score 9: 0.7040166870215224
```

In [478]:
```
pip install nbconvert
```

```
Requirement already satisfied: nbconvert in c:\user
s\tlili\anaconda3\lib\site-packages (7.16.6)
Requirement already satisfied: beautifulsoup4 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (4.12.3)
Requirement already satisfied: bleach!=5.0.0 in
c:\users\tlili\anaconda3\lib\site-packages (from bl
each[css]!=5.0.0->nbconvert) (6.2.0)
Requirement already satisfied: defusedxml in c:\use
rs\tlili\anaconda3\lib\site-packages (from nbconver
t) (0.7.1)
Requirement already satisfied: jinja2>=3.0 in c:\us
ers\tlili\anaconda3\lib\site-packages (from nbconve
rt) (3.1.6)
Requirement already satisfied: jupyter-core>=4.7 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (5.7.2)
Requirement already satisfied: jupyterlab-pygments
in c:\users\tlili\anaconda3\lib\site-packages (from
nbconvert) (0.3.0)
Requirement already satisfied: markupsafe>=2.0 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (3.0.2)
Requirement already satisfied: mistune<4,>=2.0.3 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (3.1.2)
Requirement already satisfied: nbclient>=0.5.0 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (0.10.2)
Requirement already satisfied: nbformat>=5.7 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (5.10.4)
Requirement already satisfied: packaging in c:\user
s\tlili\anaconda3\lib\site-packages (from nbconver
t) (24.2)
Requirement already satisfied: pandocfilters>=1.4.1
in c:\users\tlili\anaconda3\lib\site-packages (from
nbconvert) (1.5.0)
Requirement already satisfied: pygments>=2.4.1 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
```

```
convert) (2.19.1)
Requirement already satisfied: traitlets>=5.1 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
convert) (5.14.3)
Requirement already satisfied: webencodings in c:\u
sers\tlili\anaconda3\lib\site-packages (from bleac
h!=5.0.0->bleach[css]!=5.0.0->nbconvert) (0.5.1)
Requirement already satisfied: tinycss2<1.5,>=1.1.0
in c:\users\tlili\anaconda3\lib\site-packages (from
bleach[css]!=5.0.0->nbconvert) (1.4.0)
Requirement already satisfied: platformdirs>=2.5 in
c:\users\tlili\anaconda3\lib\site-packages (from ju
pyter-core>=4.7->nbconvert) (4.3.7)
Requirement already satisfied: pywin32>=300 in c:\u
sers\tlili\anaconda3\lib\site-packages (from jupyte
r-core>=4.7->nbconvert) (308)
Requirement already satisfied: jupyter-client>=6.1.
12 in c:\users\tlili\anaconda3\lib\site-packages (f
rom nbclient>=0.5.0->nbconvert) (8.6.3)
Requirement already satisfied: python-dateutil>=2.
8.2 in c:\users\tlili\anaconda3\lib\site-packages
(from jupyter-client>=6.1.12->nbclient>=0.5.0->nbco
nvert) (2.9.0.post0)
Requirement already satisfied: pyzmq>=23.0 in c:\us
ers\tlili\anaconda3\lib\site-packages (from jupyter
-client>=6.1.12->nbclient>=0.5.0->nbconvert) (26.2.
0)
Requirement already satisfied: tornado>=6.2 in c:\u
sers\tlili\anaconda3\lib\site-packages (from jupyte
r-client>=6.1.12->nbclient>=0.5.0->nbconvert) (6.5.
1)
Requirement already satisfied: fastjsonschema>=2.15
in c:\users\tlili\anaconda3\lib\site-packages (from
nbformat>=5.7->nbconvert) (2.20.0)
Requirement already satisfied: jsonschema>=2.6 in
c:\users\tlili\anaconda3\lib\site-packages (from nb
format>=5.7->nbconvert) (4.23.0)
Requirement already satisfied: attrs>=22.2.0 in
c:\users\tlili\anaconda3\lib\site-packages (from js
onschema>=2.6->nbformat>=5.7->nbconvert) (24.3.0)
Requirement already satisfied: jsonschema-specifica
```

```
tions>=2023.03.6 in c:\users\tlili\anaconda3\lib\si
te-packages (from jsonschema>=2.6->nbformat>=5.7->n
bconvert) (2023.7.1)
Requirement already satisfied: referencing>=0.28.4
in c:\users\tlili\anaconda3\lib\site-packages (from
jsonschema>=2.6->nbformat>=5.7->nbconvert) (0.30.2)
Requirement already satisfied: rpds-py>=0.7.1 in
c:\users\tlili\anaconda3\lib\site-packages (from js
onschema>=2.6->nbformat>=5.7->nbconvert) (0.22.3)
Requirement already satisfied: six>=1.5 in c:\users
\tlili\anaconda3\lib\site-packages (from python-dat
eutil>=2.8.2->jupyter-client>=6.1.12->nbclient>=0.
5.0->nbconvert) (1.17.0)
Requirement already satisfied: soupsieve>1.2 in
c:\users\tlili\anaconda3\lib\site-packages (from be
autifulsoup4->nbconvert) (2.5)
Note: you may need to restart the kernel to use upd
ated packages.
```