King County Development Recommendations

Business Understanding

A midsized real estate developer in King County, Washington, has engaged the Argon Team to gather and analyze available data and determine what factors of a home most influence price and by how much. The Argon Team has identified trends in real estate prices in King County to determine a) the optimal locations in which to develop and b) the most potentially profitable housing configurations.

Data understanding

The primary source of the available data was provided as a CSV file containing the King County House Sales dataset and was gathered between May, 2014, and May, 2015. It should be noted that the data is already several years old and King County (which includes Seattle) has continued to grow at a rate far exceeding the rest of the country. It has also recently experienced the effects of the global Covid pandemic. This data set predates these factors and therefore cannot reflect their impact if, indeed, there was an impact.

A secondary source of data was taken from the [point 2 website (https://www.point2homes.com/US/Neighborhood/WA/King-County-Demographics.html#MedianIncomeByZipcode)](https://www.point2homes.com/US/Neighborhood/WA/King-County-Demographics.html#MedianIncomeByZipcode). This provided population figures and the median and mean income of each zip code in King County.

Import Relevant Modules

In [85]:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.datasets import make_regression
from sklearn.linear_model import LinearRegression
import sklearn.metrics as metrics
from random import gauss
from mpl_toolkits.mplot3d import Axes3D
from scipy import stats as stats
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_validate
from statsmodels.formula.api import ols
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.feature_selection import RFE
from sklearn.preprocessing import PolynomialFeatures, StandardScaler

%matplotlib inline
```

Exploring Data and Cleaning Up Null Values

In [86]:

```
df = pd.read_csv('data/kc_house_data.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21597 entries, 0 to 21596
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             21597 non-null  int64  
 1   date           21597 non-null  object 
 2   price          21597 non-null  float64
 3   bedrooms       21597 non-null  int64  
 4   bathrooms      21597 non-null  float64
 5   sqft_living    21597 non-null  int64  
 6   sqft_lot       21597 non-null  int64  
 7   floors         21597 non-null  float64
 8   waterfront     19221 non-null  object 
 9   view           21534 non-null  object 
 10  condition      21597 non-null  object 
 11  grade          21597 non-null  object 
 12  sqft_above     21597 non-null  int64  
 13  sqft_basement  21597 non-null  object 
 14  yr_built       21597 non-null  int64  
 15  yr_renovated   17755 non-null  float64
 16  zipcode        21597 non-null  int64  
 17  lat            21597 non-null  float64
 18  long           21597 non-null  float64
 19  sqft_living15  21597 non-null  int64  
 20  sqft_lot15     21597 non-null  int64  
dtypes: float64(6), int64(9), object(6)
memory usage: 3.5+ MB
```

In [87]:

```
df
```

Out[87]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 21 columns

Creating a graph that plots the location of houses sold.

For more information see this blog post: [https://towardsdatascience.com/easy-steps-to-plot-geographic-data-on-a-map-python-11217859a2db (https://towardsdatascience.com/easy-steps-to-plot-geographic-data-on-a-map-python-11217859a2db)](https://towardsdatascience.com/easy-steps-to-plot-geographic-data-on-a-map-python-11217859a2db)

In [142]:

```
# Using an online resource to create a map within specefic longitude and
BBox = (-122.434, -121.604,
         47.3643, 47.777)
BBox
```

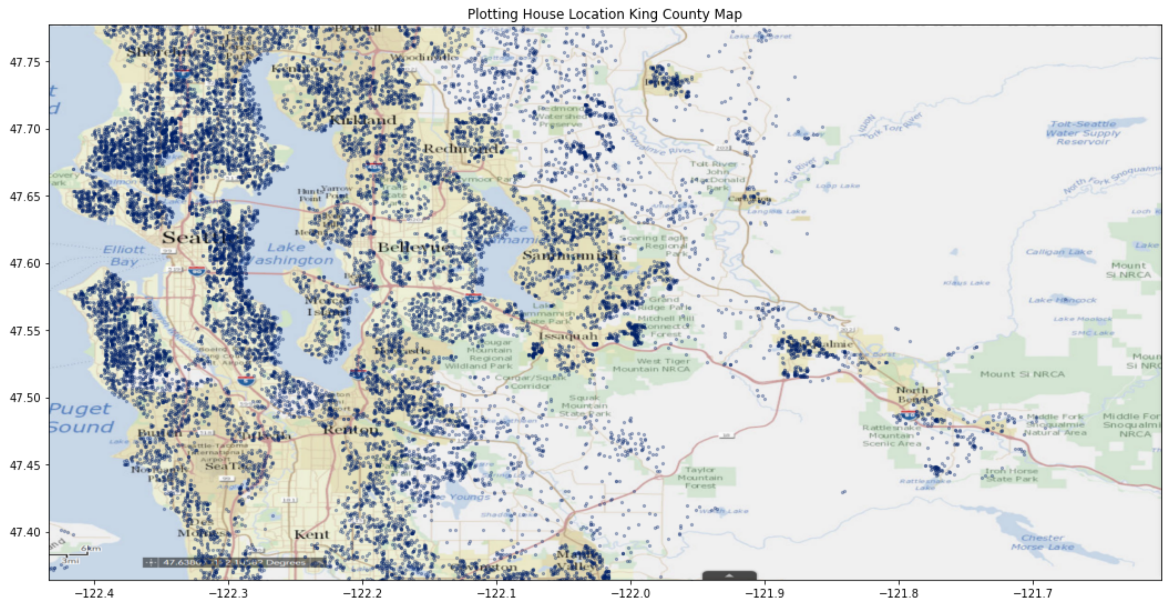
Out[142]: (-122.434, -121.604, 47.3643, 47.777)

In [143]:

```
# create a more interpretable variable for the map
kc_map = plt.imread('images/king_county_map_1.png')

```

In [144]:

```python
fig, ax = plt.subplots(figsize=(18, 15))

ax.scatter(df.long, df.lat, alpha= 0.3, c='#012169', s=5)
ax.set_title('Plotting House Location King County Map')
ax.set_xlim(BBox[0],BBox[1])
ax.set_ylim(BBox[2],BBox[3])
ax.imshow(kc_map, zorder=0, extent = BBox, aspect= 'equal')
plt.show()
```



Eliminating Outliers

In [ ]:

```python
# During analyses we saw that one home had 33 bedrooms with the next high
# so we are going to get rid of that outlier as it was most likely a mis
df = df[df['bedrooms'] < 10]
```

Make Grade only Numeric

In [88]:

```
# Create a new dataframe using genres listed in my existing data frame.
grade_df = df['grade'].str.split(' ', expand = True)
#Set the column names
grade_df.rename(columns = {0 : 'numerical_grade'}, inplace = True)
grade_df.drop([1 , 2], axis = 1, inplace = True)
grade_df
# Add the expanded columns back to the original df
df = df.merge(grade_df, right_index = True, left_index = True)
df
```

Out[88]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 22 columns

In [89]:

```
# Make numerical grade column numeric
df['numerical_grade'] = df['numerical_grade'].astype('float64')

```

Fill in null values with the null values associated to the column

Looking briefly at the null values, we can safely assume that the majority of them are data entry problems. There is a null value for each column that is an overwhelming proportion, this is safe to do.

In [90]:

```
df['waterfront'].fillna('NO', inplace=True)
df['yr_renovated'].fillna(0.0, inplace=True)
df['view'].fillna('NONE', inplace = True)
```

Replace View Column with numeric Values

In [91]:

```
view_map = {'EXCELLENT':5, 'GOOD':4, 'AVERAGE':3, 'FAIR':2, 'NONE':1, np
df['view_num'] = df['view'].map(view_map)
df
```

Out[91]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 23 columns

Add Zipcode Data

Data is from: https://www.point2homes.com/US/Neighborhood/WA/King-County-Demographics.html#MedianIncomeByZipcaode (https://www.point2homes.com/US/Neighborhood/WA/King-County-Demographics.html#MedianIncomeByZipcaode)

In [92]:

```
df_income = pd.read_csv('data/kc_zipcode_data.csv')
df_income.head()
```

Out[92]:

	ZipCode	Population	Number of Households	Median Income	Average Income
0	98001	34,455	11,648	$88,962.00	$102,586.00
1	98002	33,947	13,162	$59,097.00	$70,945.00
2	98003	49,445	18,515	$59,560.00	$76,753.00
3	98004	37,265	17,460	$142,173.00	$210,129.00
4	98005	21,414	8,590	$135,225.00	$186,020.00

In [93]:

```python
df_income.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 84 entries, 0 to 83
Data columns (total 5 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   ZipCode               84 non-null     int64 
 1   Population            84 non-null     object
 2   Number of Households  84 non-null     object
 3   Median Income         84 non-null     object
 4   Average Income        84 non-null     object
dtypes: int64(1), object(4)
memory usage: 3.4+ KB
```

Clean up the data to make each column numerical

In [94]:

```python
# Rename zipcode to match existing datafram
df_income.rename(columns = {'ZipCode': 'zipcode'}, inplace=True)
# Merge the dataframes together
df = df.merge(df_income, how = 'left', on = 'zipcode')
```

In [95]:

```python
# Get rid of commas in our new data
df['Median Income']=df['Median Income'].str.replace(',','')
df['Average Income']=df['Average Income'].str.replace(',','')
df['Number of Households']=df['Number of Households'].str.replace(',',''
df['Population']=df['Population'].str.replace(',','')
df['Population']=df['Population'].str.replace(',','')

#Get rid of dollar signs in our new data
df['Median Income']=df['Median Income'].str.replace('$','')
df['Average Income']=df['Average Income'].str.replace('$','')
```

In [96]:

```python
# Turn our new data into floats so we can use it for regression analysis
df['Median Income'] = df['Median Income'].astype('float64')
df['Average Income'] = df['Average Income'].astype('float64')
df['Number of Households'] = df['Number of Households'].astype('float64'
df['Population'] = df['Population'].astype('float64')
```

In [97]:

```python
# Sanity Check
df
```

Out[97]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 27 columns

Using One Hot Encoder for Waterfront Collumn

In [98]:

```python
# creating instance of one-hot-encoder
enc = OneHotEncoder(handle_unknown='error', drop = 'first')
# passing bridge-types-cat column (label encoded values of bridge_types)
enc_df = pd.DataFrame(enc.fit_transform(df[['waterfront']]).toarray())
enc_df
# merge with main df bridge_df on key values
df = df.merge(enc_df, right_index = True, left_index = True)
```

In [99]:

```
#sanity check
df
```

Out[99]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 28 columns

In [100]:

```
# rename the column created to something meaningfull
df.rename(columns={0: "Waterfront"}, inplace = True)
df
```

Out[100]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0
...	...	...	...	...	...	...	...	...
21592	263000018	5/21/2014	360000.0	3	2.50	1530	1131	3.0
21593	6600060120	2/23/2015	400000.0	4	2.50	2310	5813	2.0
21594	1523300141	6/23/2014	402101.0	2	0.75	1020	1350	2.0
21595	291310100	1/16/2015	400000.0	3	2.50	1600	2388	2.0
21596	1523300157	10/15/2014	325000.0	2	0.75	1020	1076	2.0

21597 rows × 28 columns

Create a Dataframe with Only numeric columns

Since we plan on running regression analyses we can drop the non-numeric columns

In [101]:

```
df.drop(['sqft_basement', 'view', 'condition', 'date', 'id', 'grade', 'wa

```

In [102]:

```
#sanity check
df.info()
```

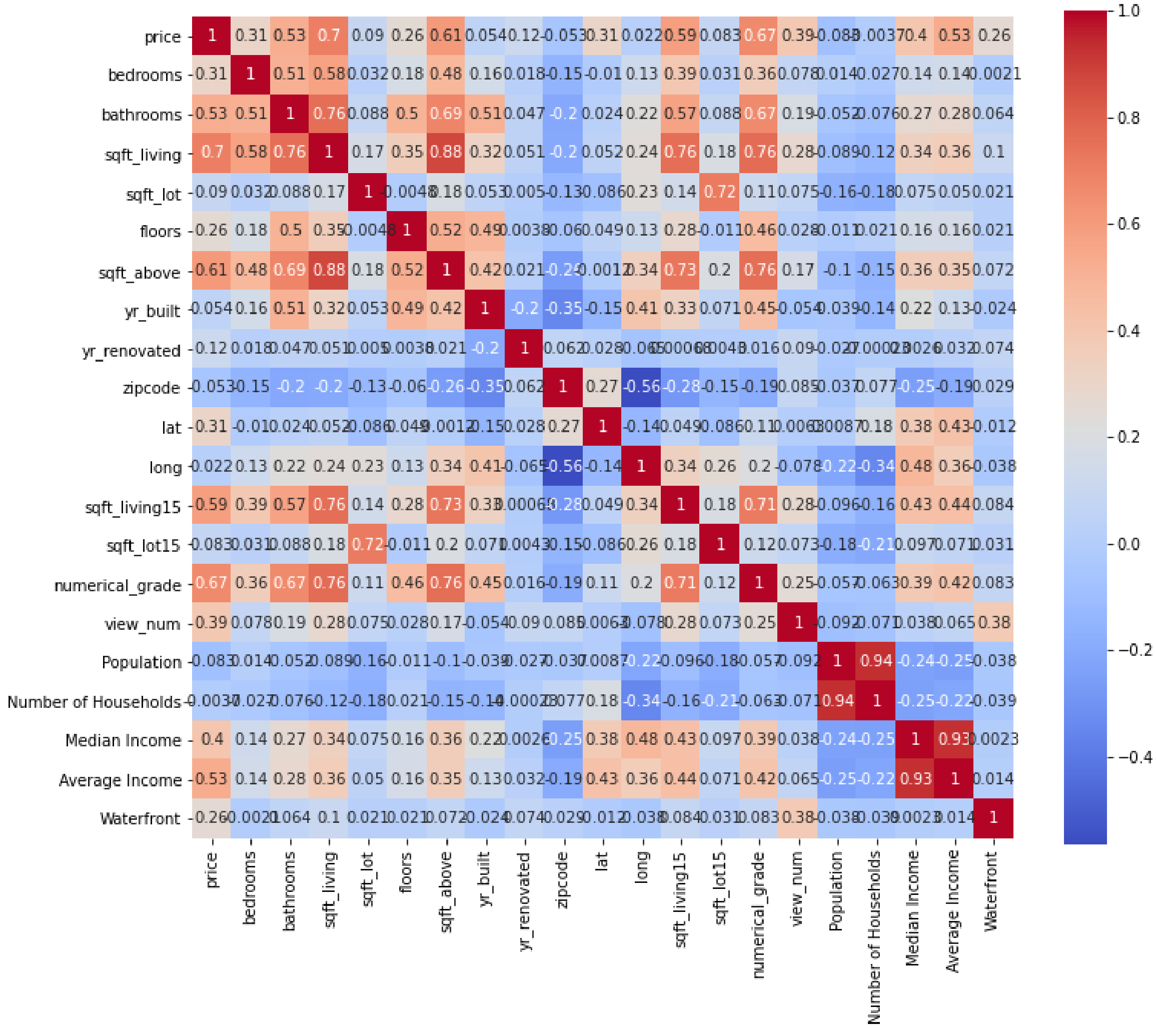
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 21597 entries, 0 to 21596
Data columns (total 21 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   price                  21597 non-null  float64
 1   bedrooms               21597 non-null  int64  
 2   bathrooms              21597 non-null  float64
 3   sqft_living            21597 non-null  int64  
 4   sqft_lot               21597 non-null  int64  
 5   floors                 21597 non-null  float64
 6   sqft_above             21597 non-null  int64  
 7   yr_built               21597 non-null  int64  
 8   yr_renovated           21597 non-null  float64
 9   zipcode                21597 non-null  int64  
 10  lat                    21597 non-null  float64
 11  long                   21597 non-null  float64
 12  sqft_living15          21597 non-null  int64  
 13  sqft_lot15             21597 non-null  int64  
 14  numerical_grade        21597 non-null  float64
 15  view_num               21597 non-null  int64  
 16  Population             21597 non-null  float64
 17  Number of Households   21597 non-null  float64
 18  Median Income          21597 non-null  float64
 19  Average Income         21597 non-null  float64
 20  Waterfront             21597 non-null  float64
dtypes: float64(12), int64(9)
memory usage: 4.2 MB
```

Creating a Heat Chart to Judge Correlation

In [103]:

```
plt.figure(figsize = (12, 10))

sns.heatmap(df.corr(), cmap="coolwarm", annot = True, square = True)
```

Out[103]: <AxesSubplot:>



Examining Linearity of Potential X Variables

In [104]:

```
# rename our dataframes for simpler code.
y = df['price']
X = df.drop(columns=['price'], axis=1)

```

In [105]:

```
# sanity check
X.columns
```

Out[105]:

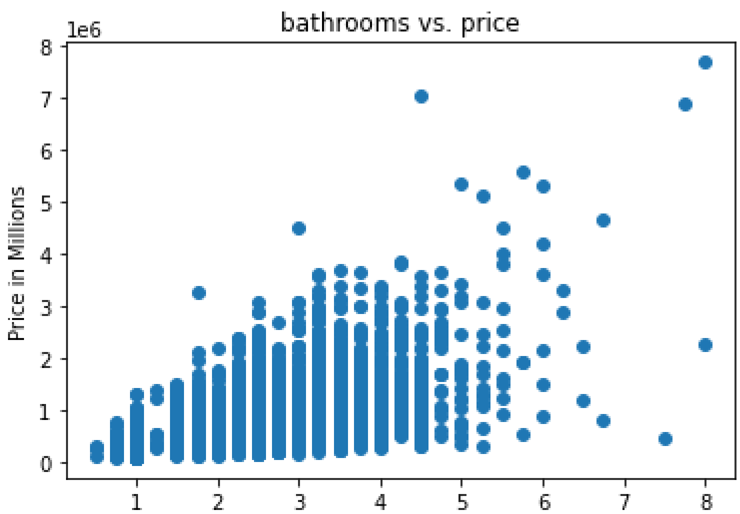
```
Index(['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',
       'sqft_above', 'yr_built', 'yr_renovated', 'zipcode', 'lat', 'long',
       'sqft_living15', 'sqft_lot15', 'numerical_grade', 'view_num',
       'Population', 'Number of Households', 'Median Income', 'Average Inco
me',
       'Waterfront'],
      dtype='object')
```

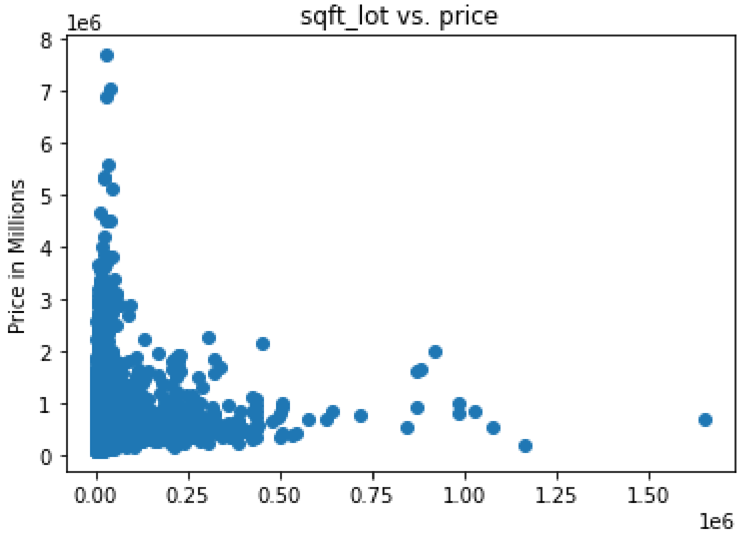
In [106]:

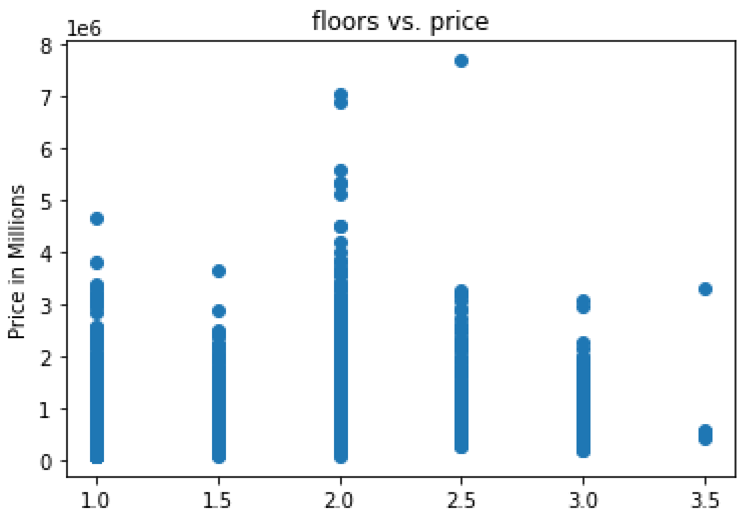
```
# Rename our columns to be consistant
df.rename(columns={"Number of Households": 'number_of_households', 'Medi
                   , 'Average Income': 'average_income'}, inplace = True

```

In [107]:

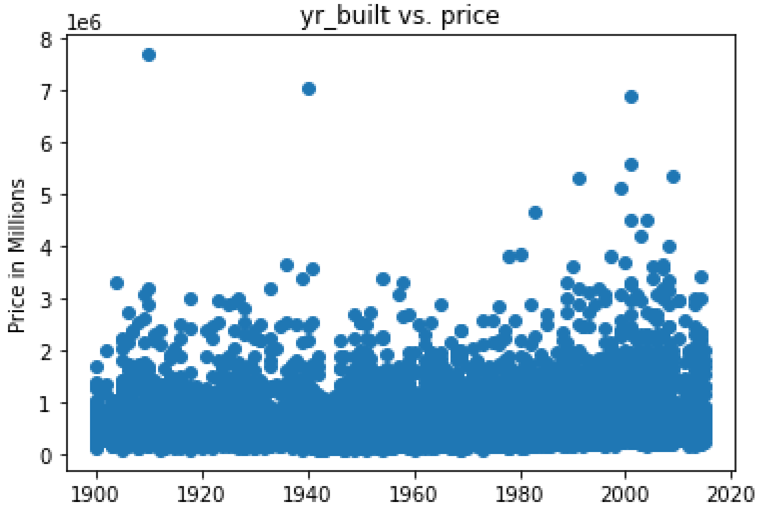
```
for col in X.columns:
    plt.scatter(X[col], y)
    plt.title(f"{col} vs. price")
    plt.ylabel("Price in Millions")
    plt.show()
```

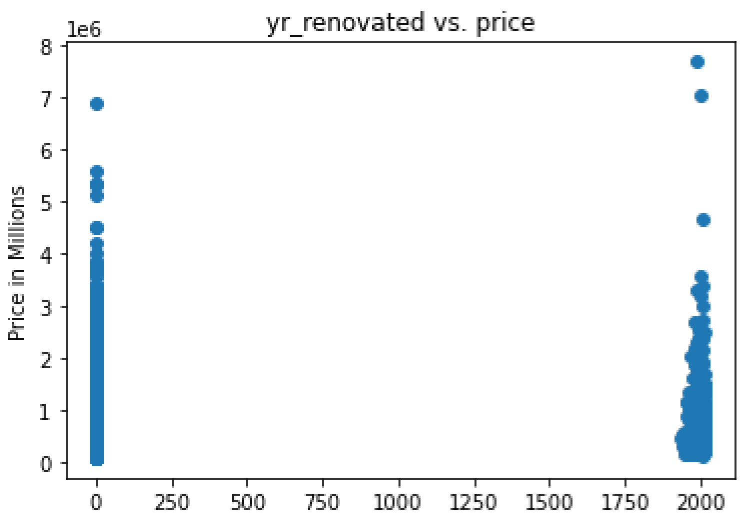


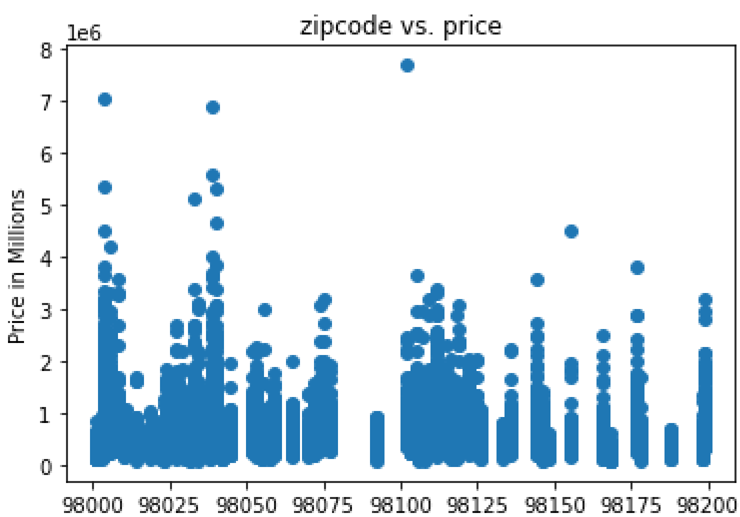

sqft_lot vs. price
1e6
8
7
6
5
4
3
2
1
0
Price in Millions
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1e6

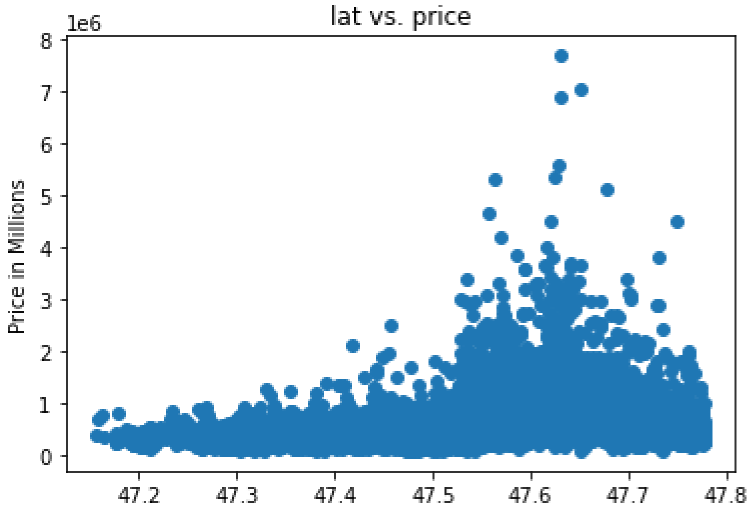

floors vs. price
1e6
8
7
6
5
4
3
2
1
0
Price in Millions
1.0
1.5
2.0
2.5
3.0
3.5

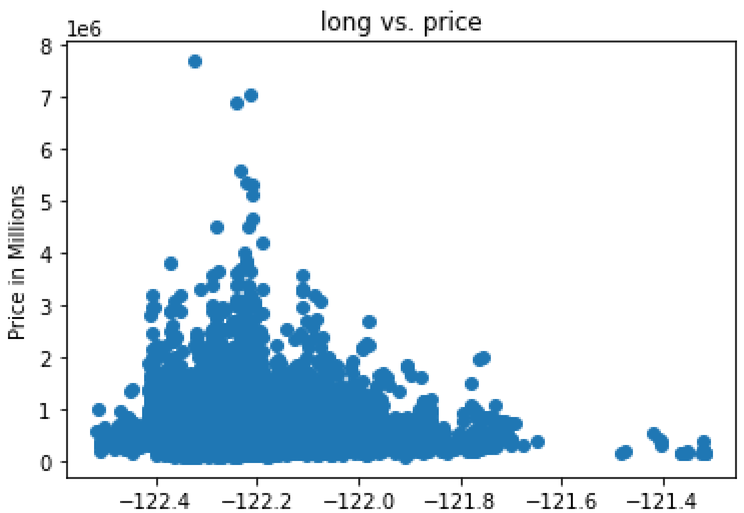

sqft_above vs. price
1e6

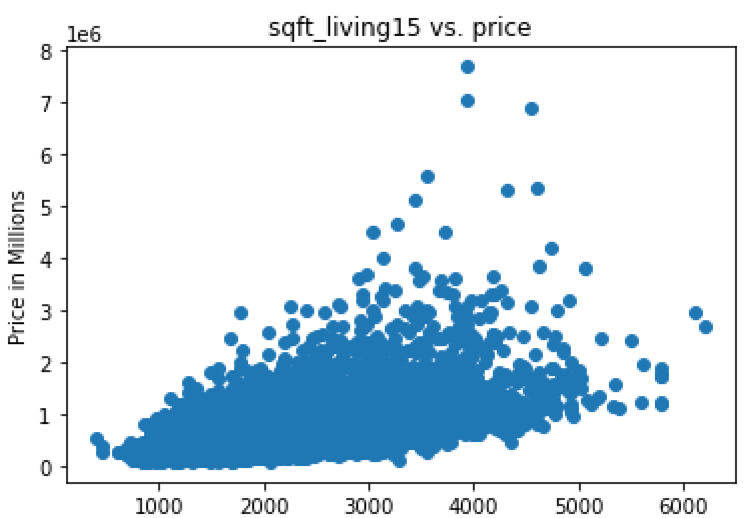

yr_built vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
1900
1920
1940
1960
1980
2000
2020

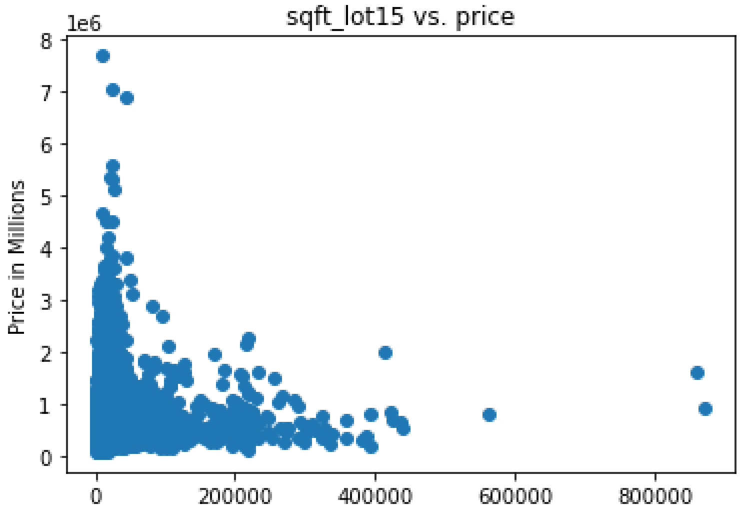

yr_renovated vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
0
250
500
750
1000
1250
1500
1750
2000

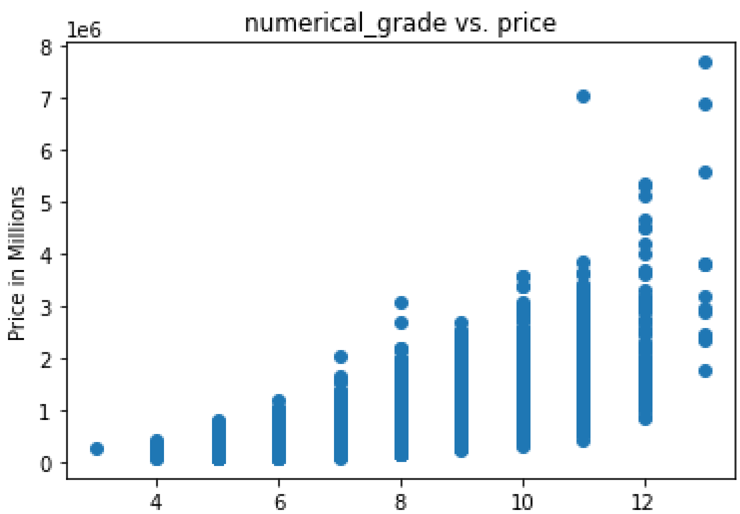

zipcode vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
98000
98025
98050
98075
98100
98125
98150
98175
98200

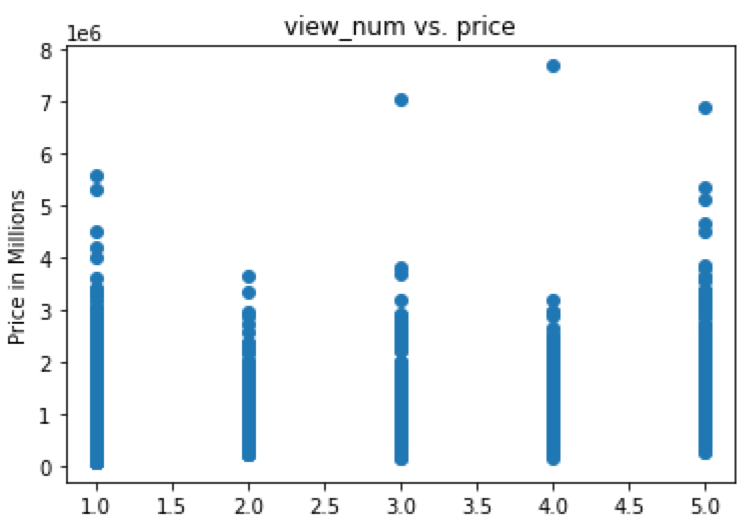

lat vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
47.2
47.3
47.4
47.5
47.6
47.7
47.8

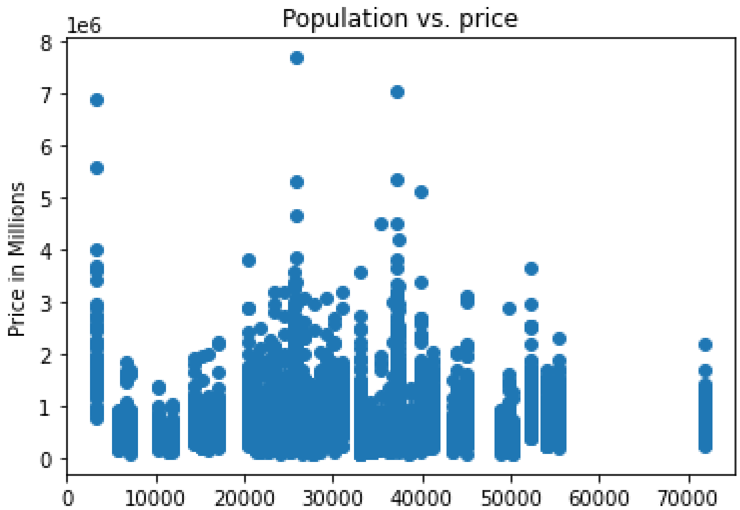

long vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
−122.4
−122.2
−122.0
−121.8
−121.6
−121.4

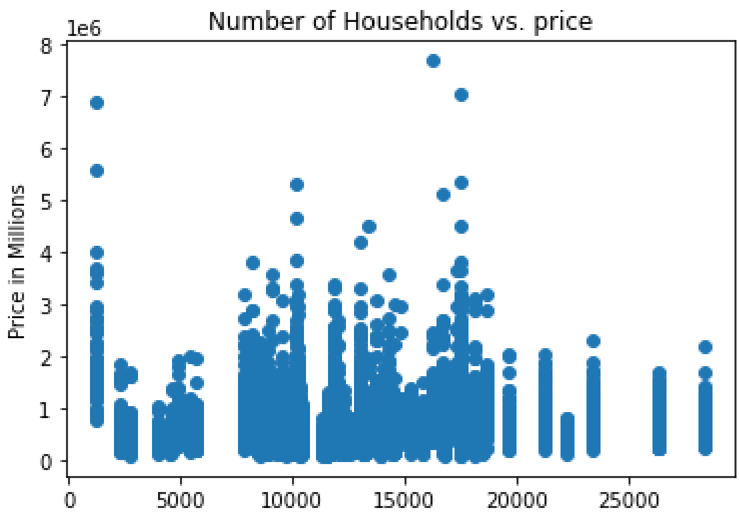

sqft_living15 vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
1000
2000
3000
4000
5000
6000

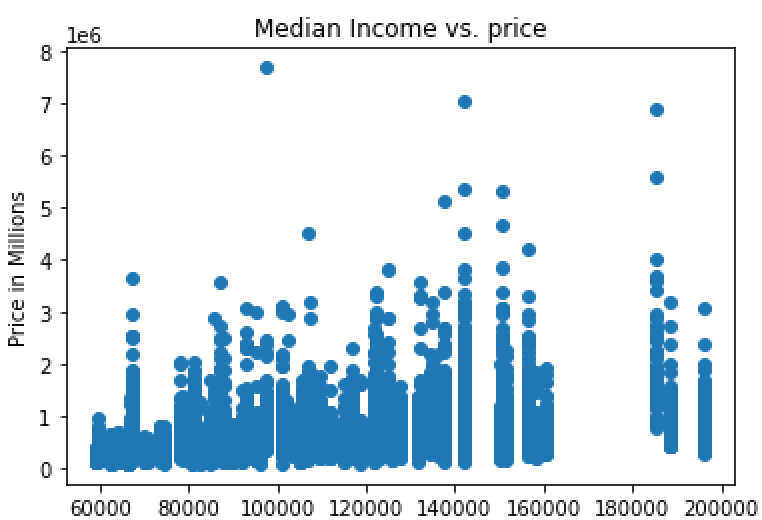

sqft_lot15 vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
0
200000
400000
600000
800000

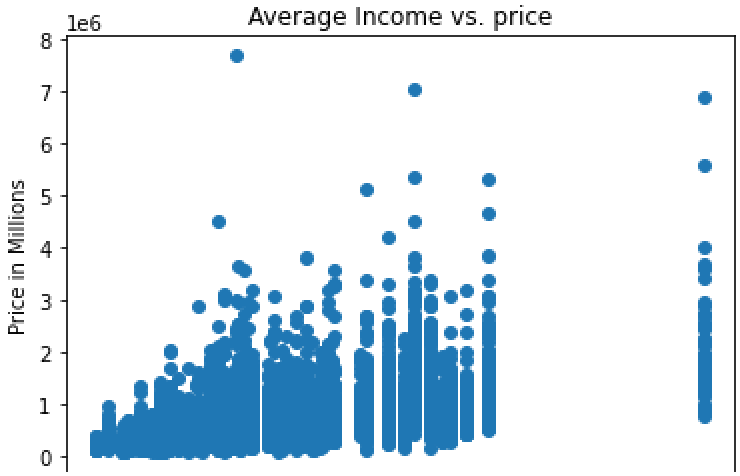

numerical_grade vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
4
6
8
10
12

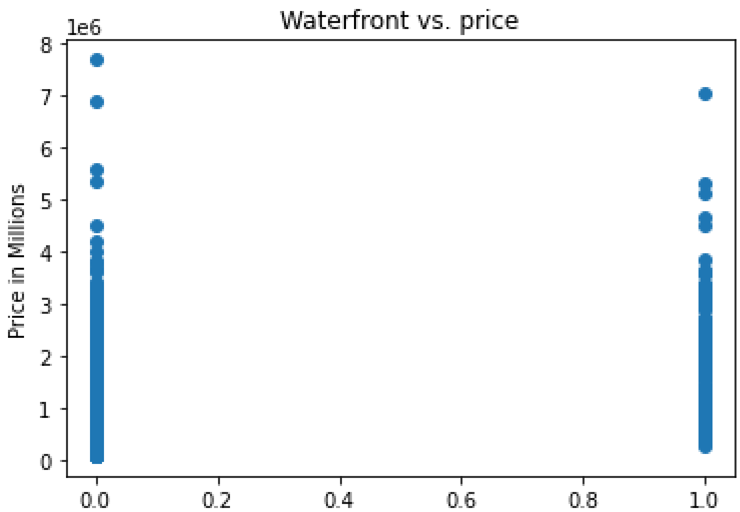

view_num vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Population vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
0
10000
20000
30000
40000
50000
60000
70000


Number of Households vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
0
5000
10000
15000
20000
25000


Median Income vs. price
1e6
Price in Millions
0
1
2
3
4
5
6
7
8
60000
80000
100000
120000
140000
160000
180000
200000





Linear Relationship Analysis

Most of the variables we looked at don't appear to have a linear relationship with price.

Create a Renovated Column

Although there doesn't appear to be much correlation with when a house was built to price. Intuitively it would make sense that houses that have recently been worked on would be more desirable than those that havent. Creating a column that is binary that says whether work has ever been done on it may be useful for our analyses.

In [108]:

```
# Create a new column and set it equal to zero
df['renovated'] = 0
# If renovated year is  after the year built set it equal to 1
df['renovated'].loc[df['yr_renovated'] >df['yr_built']] = 1
df
```

```
C:\Users\Johnn\anaconda3\envs\learn-env\lib\site-packages\pandas\core\index
ing.py:670: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://p
andas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  iloc._setitem_with_indexer(indexer, value)
```

Out[108]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	sqft_above	yr_built	yr_r
0	221900.0	3	1.00	1180	5650	1.0	1180	1955	
1	538000.0	3	2.25	2570	7242	2.0	2170	1951	
2	180000.0	2	1.00	770	10000	1.0	770	1933	
3	604000.0	4	3.00	1960	5000	1.0	1050	1965	
4	510000.0	3	2.00	1680	8080	1.0	1680	1987	
...	...	...	...	...	...	...	...	...	
21592	360000.0	3	2.50	1530	1131	3.0	1530	2009	
21593	400000.0	4	2.50	2310	5813	2.0	2310	2014	
21594	402101.0	2	0.75	1020	1350	2.0	1020	2009	
21595	400000.0	3	2.50	1600	2388	2.0	1600	2004	
21596	325000.0	2	0.75	1020	1076	2.0	1020	2008	

21597 rows × 22 columns

In [ ]:

```

```

Examine the Price Distribution

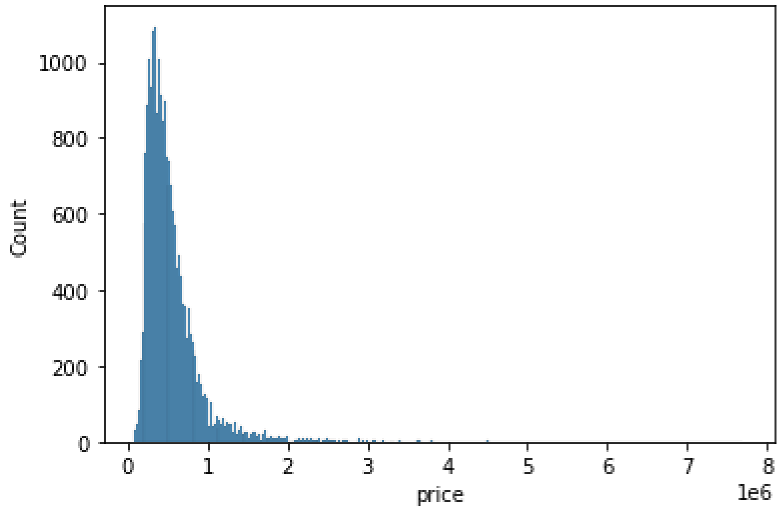
It's important to examine the distribution of price because to see if it is skewed or their is a large tail. This will have an impact on our regressions that we may need to address when building our final model.

In [109]:

```python
sns.histplot(df['price'])
```

Out[109]:

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



It appears that their is large right tail for our price model that we may want to address in our final model.

Data Analysis

Reserve some data for model validation

In [110]:

```python
y = df['price']
X = df.drop(columns=['price'], axis=1)


X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=.2,
                                                    random_state=42
)

```

Modelless Base

With no model we would use the price mean of our training data to predict the price of our testing data.

In [111]:

```
train_target_mean = y_train.mean()

baseline_train_preds = [train_target_mean] * len(y_train)
baseline_test_preds = [train_target_mean] * len(y_test)

# R squared
print('Model fit for training data')
print(f"R2: {metrics.r2_score(y_train, baseline_train_preds):.2f}")
# Mean squared error
print(f"Mean Square Error: {metrics.mean_squared_error(y_train, baseline_
# Mean absolute error
print(f"Mean Absolute Error: {metrics.mean_absolute_error(y_train, basel
print("\n")

# R squared
print('Model fit for test data')
print(f"R2: {metrics.r2_score(y_test, baseline_test_preds):.2f}")
# Mean squared error
print(f"Mean Square Error: {metrics.mean_squared_error(y_test, baseline_
# Mean absolute error
print(f"Mean Absolute Error: {metrics.mean_absolute_error(y_test, baseli
```

```
Model fit for training data
R2: 0.00
Mean Square Error: 368,958
Mean Absolute Error: 235,058


Model fit for test data
R2: -0.00
Mean Square Error: 360,907
Mean Absolute Error: 231,542
```

Create Functions to help make regression analysis easier

Inspiration for this code came from: [https://github.com/brooke57 (https://github.com/brooke57)](https://github.com/brooke57)

In [112]:

```
def model(ind_variable, data):
    formula = 'price ~ ' + ' + '.join(ind_variable)
    multi_model = ols(formula, data).fit()
    multi_model_summ = multi_model.summary()
    return multi_model,multi_model_summ
```

In [113]:

```
def assess(model):
    tr_preds=model.predict(X_train)
    te_preds=model.predict(X_test)
    y_tr = y_train
    y_te = y_test
# Format the string output using f-strings
    print(f"Train R2: {r2_score(y_tr, tr_preds)}")
    print(f"Test R2: {r2_score(y_te, te_preds)}")
    print('----')
    print(f"Train RMSE: {mean_squared_error(y_tr, tr_preds, squared = Fa
    print(f"Test RMSE: {mean_squared_error(y_te, te_preds, squared = Fal
    print('----')
    print(f"Train MAE: {mean_absolute_error(y_tr, tr_preds)}")
    print(f"Test MAE: {mean_absolute_error(y_te, te_preds)}")


# Set Variables for graphing
    tr_res= y_tr - tr_preds
    te_res= y_te - te_preds

# Graph Syntax
    plt.scatter(tr_preds, tr_res, label = 'Train')
    plt.scatter(te_preds, te_res, label = 'Test')

    plt.axhline(y=0, color = 'red', label = '0')
    plt.xlabel('predictions')
    plt.ylabel('residuals')
    plt.legend()
    plt.show
```

In [114]:

```
def scaled_model(ind_variable, data):
    formula = 'price ~ ' + ' + '.join(ind_variable)
    data_scaled = (data - np.mean(data)) / np.std(data)
    model_scaled = ols(formula, data_scaled).fit()
    model_scaled_summ = model_scaled.summary()
    return model_scaled_summ
```

In [115]:

```
def model_and_assess(ind_variable, data):
    multi_model, multi_model_summ = model(ind_variable,data)
    assessment = assess(multi_model)
    scaled_summ = scaled_model(ind_variable,data)
    qq = sm.graphics.qqplot(multi_model.resid, dist=stats.norm, line='45
    print('         ')
    print('This is the summary of the model')
    print('         ')
    print(multi_model_summ)
    print('         ')
    print('This is the summary of the scaled model')
    print('         ')
    print(scaled_summ)
    print('         ')
    print('This is the correlation table between variables')
    print('         ')
    print(data[ind_variable].corr())
    print('         ')
    print('This is the residual plot and qq plot')
    print('         ')
    print(assessment)
    print(qq)
```

Create a base model with the highest correlated column as the only feature

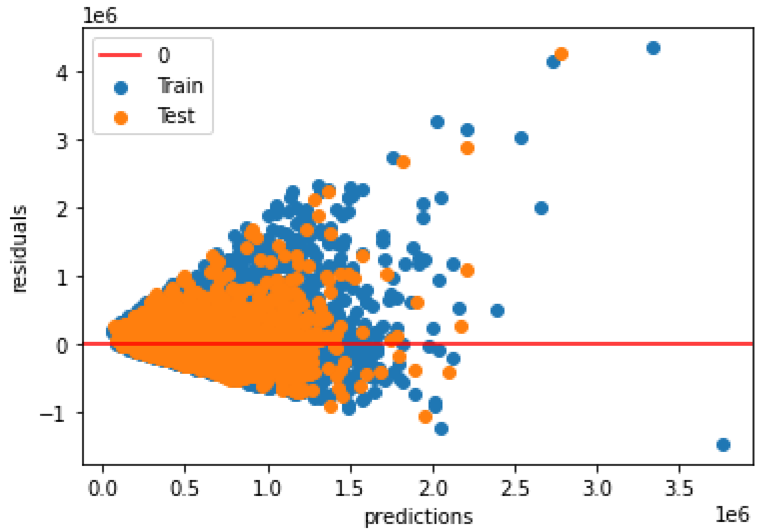
In [116]:

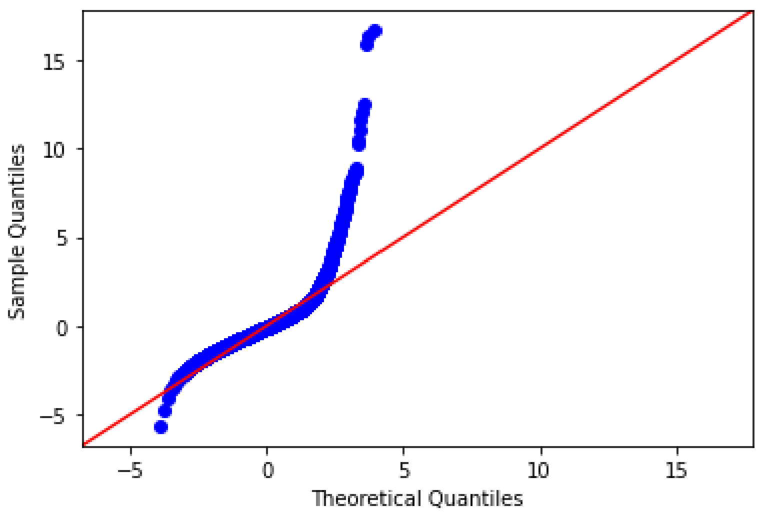
```
1 model_and_assess(['sqft_living'],df)
```

```
Train R2: 0.49248102591707765
Test R2: 0.4934364209286596
----
Train RMSE: 262847.0640099154
Test RMSE: 256832.28945676202
----
Train MAE: 174592.15379749413
Test MAE: 170756.35511471998

This is the summary of the model

                            OLS Regression Results
==============================================================================
======
Dep. Variable:                  price   R-squared:
0.493
Model:                            OLS   Adj. R-squared:
0.493
Method:                 Least Squares   F-statistic:                     2.0
97e+04
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:20   Log-Likelihood:              -3.00
06e+05
No. Observations:               21597   AIC:                             6.0
01e+05
Df Residuals:                   21595   BIC:                             6.0
01e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
=======
                  coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-------
Intercept   -4.399e+04   4410.023     -9.975      0.000   -5.26e+04    -
3.53e+04
sqft_living   280.8630      1.939    144.819      0.000     277.062
284.664
==============================================================================
======
Omnibus:                    14801.942   Durbin-Watson:
1.982
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              5426
62.604
Skew:                           2.820   Prob(JB):
0.00
Kurtosis:                      26.901   Cond. No.                         5.
63e+03
==============================================================================
======

Notes:
```

```
[1] Standard Errors assume that the covariance matrix of the errors is c
orrectly specified.
[2] The condition number is large, 5.63e+03. This might indicate that th
ere are
strong multicollinearity or other numerical problems.

This is the summary of the scaled model

                            OLS Regression Results
==========================================================================
======
Dep. Variable:                  price   R-squared:
0.493
Model:                            OLS   Adj. R-squared:
0.493
Method:                 Least Squares   F-statistic:                   2.0
97e+04
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:21   Log-Likelihood:                  -
23317.
No. Observations:               21597   AIC:                           4.6
64e+04
Df Residuals:                   21595   BIC:                           4.6
65e+04
Df Model:                           1
Covariance Type:            nonrobust
==========================================================================
=======
                  coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------
-------
Intercept   -6.353e-17      0.005  -1.31e-14      1.000      -0.010
0.010
sqft_living     0.7019      0.005    144.819      0.000       0.692
0.711
==========================================================================
======
Omnibus:                    14801.942   Durbin-Watson:
1.982
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             5426
62.604
Skew:                           2.820   Prob(JB):
0.00
Kurtosis:                      26.901   Cond. No.
1.00
==========================================================================
======

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

This is the correlation table between variables

             sqft_living
```

```
sqft_living         1.0

This is the residual plot and qq plot

None
Figure(432x288)
```





Add Additional features to our Model based on Correlation Scores

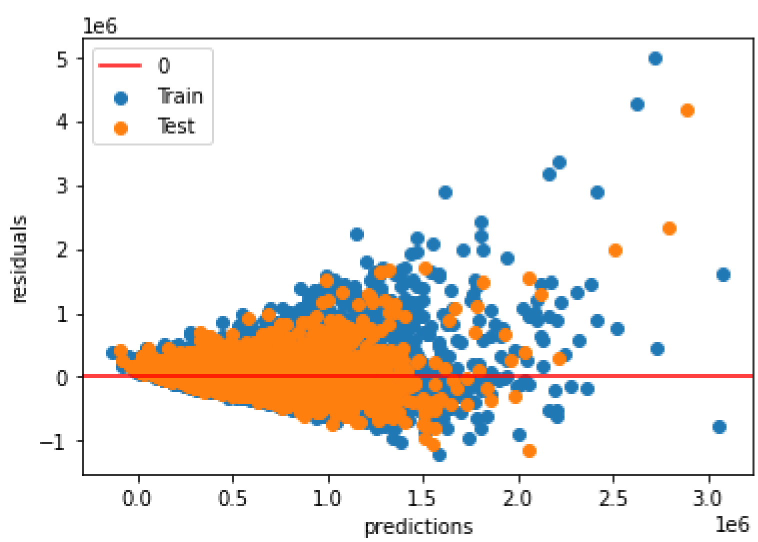
In [117]:

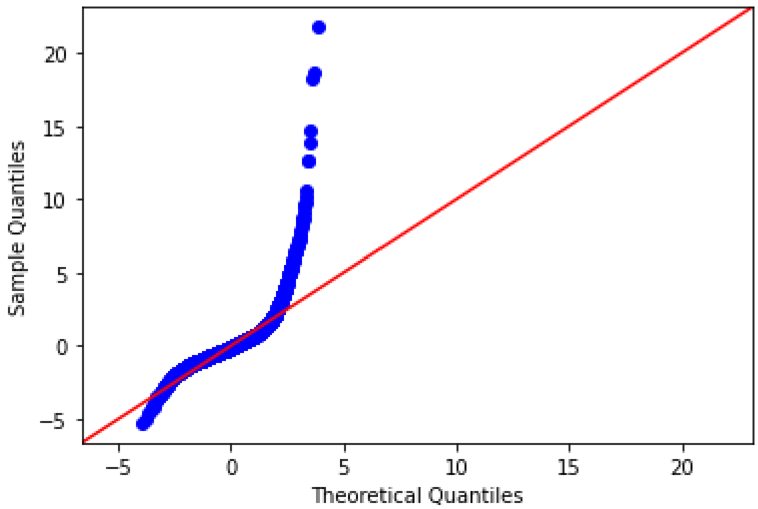
```
model_and_assess(['sqft_living', 'median_income', 'numerical_grade', 'vi
                  'Waterfront', 'renovated'],df)
```

```
Train R2: 0.611810655421914
Test R2: 0.6120897235946879
----
Train RMSE: 229878.7324419146
Test RMSE: 224749.2674320919
----
Train MAE: 149374.77828280875
Test MAE: 147593.1951313988

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                        0.
612
Model:                            OLS   Adj. R-squared:                   0.
612
Method:                 Least Squares   F-statistic:                      68
08.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:21   Log-Likelihood:              -2.9717e
+05
No. Observations:               21597   AIC:                         5.944e
+05
Df Residuals:                   21591   BIC:                         5.944e
+05
Df Model:                           5
Covariance Type:            nonrobust
===========================================================================
========
                      coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
--------
Intercept       -7.065e+05   1.25e+04    -56.744      0.000   -7.31e+05    -
6.82e+05
sqft_living       157.8112      2.661     59.305      0.000     152.595
163.027
median_income       1.9855      0.055     36.003      0.000       1.877
2.094
numerical_grade  7.866e+04   2106.822     37.335      0.000    7.45e+04
8.28e+04
view_num         7.675e+04   2292.007     33.486      0.000    7.23e+04
8.12e+04
Waterfront       6.324e+05   2.06e+04     30.762      0.000    5.92e+05
6.73e+05
===========================================================================
===
Omnibus:                    17407.752   Durbin-Watson:                    1.
969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            1310297.
```

```
296
Skew:                           3.360   Prob(JB):                         
0.00
Kurtosis:                      40.562   Cond. No.                     1.50e
+06
==============================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The condition number is large, 1.5e+06. This might indicate that there
are
strong multicollinearity or other numerical problems.

This is the summary of the scaled model

                            OLS Regression Results                            
==============================================================================
===
Dep. Variable:                  price   R-squared:                         0.
612
Model:                            OLS   Adj. R-squared:                    0.
612
Method:                 Least Squares   F-statistic:                       68
08.
Date:                Sun, 15 May 2022   Prob (F-statistic):                
0.00
Time:                        20:52:21   Log-Likelihood:                  -204
25.
No. Observations:               21597   AIC:                           4.086e
+04
Df Residuals:                   21591   BIC:                           4.091e
+04
Df Model:                           5                                         
Covariance Type:            nonrobust                                         
==============================================================================
========
                      coef    std err          t      P>|t|      [0.025      
0.975]
------------------------------------------------------------------------------
--------
Intercept       -6.353e-17      0.004   -1.5e-14      1.000      -0.008      
0.008
sqft_living         0.3944      0.007     59.305      0.000       0.381      
0.407
median_income       0.1665      0.005     36.003      0.000       0.157      
0.176
numerical_grade     0.2512      0.007     37.335      0.000       0.238      
0.264
view_num            0.1598      0.005     33.486      0.000       0.150      
0.169
Waterfront          0.1411      0.005     30.762      0.000       0.132      
0.150
==============================================================================
===
Omnibus:                    17407.752   Durbin-Watson:                     1.
```

```
969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         1310297.
296
Skew:                           3.360   Prob(JB):                 
0.00
Kurtosis:                      40.562   Cond. No.                 
3.05
===========================================================================
===

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

This is the correlation table between variables

                 sqft_living  median_income  numerical_grade  view_num  \
sqft_living         1.000000       0.337315         0.762779  0.281715   
median_income       0.337315       1.000000         0.387134  0.038370   
numerical_grade     0.762779       0.387134         1.000000  0.249082   
view_num            0.281715       0.038370         0.249082  1.000000   
Waterfront          0.104637       0.002275         0.082818  0.380543   

                 Waterfront  
sqft_living        0.104637  
median_income      0.002275  
numerical_grade    0.082818  
view_num           0.380543  
Waterfront         1.000000  

This is the residual plot and qq plot

None
Figure(432x288)
```





Evaluation

The model improved and none of the features had a significance level above an alpha of .05. The model is definitely an improvement but it is possable that numerical grade and price is not linearly correlated. One way to account for this in our model is to one hot encode the grade column. This would allow us to see how each grade affects the price based off of the coefficient values.

In [118]:

```
# creating instance of one-hot-encoder
enc = OneHotEncoder(handle_unknown='error')
# passing bridge-types-cat column (label encoded values of bridge_types)
enc_df = pd.DataFrame(enc.fit_transform(df[['numerical_grade']]).toarray
enc_df
# merge with main df bridge_df on key values
df = df.merge(enc_df, right_index = True, left_index = True)
list(enc.get_feature_names())
```

Out[118]:

```
['x0_3.0',
 'x0_4.0',
 'x0_5.0',
 'x0_6.0',
 'x0_7.0',
 'x0_8.0',
 'x0_9.0',
 'x0_10.0',
 'x0_11.0',
 'x0_12.0',
 'x0_13.0']
```

In [119]:

```
#sanity check
df.columns
```

Out[119]:

```
Index([                'price',                'bedrooms',               'bathroom
s',
                 'sqft_living',                'sqft_lot',                  'floor
s',
                  'sqft_above',                'yr_built',            'yr_renovate
d',
                     'zipcode',                     'lat',                    'lon
g',
               'sqft_living15',              'sqft_lot15',        'numerical_grad
e',
                    'view_num',              'Population', 'number_of_household
s',
               'median_income',          'average_income',              'Waterfron
t',
                   'renovated',                         0,
1,
                             2,                         3,
4,
                             5,                         6,
7,
                             8,                         9,
10],
      dtype='object')
```

In [120]:

```
# Rename the new columns for easier interpretability

df.rename(columns={0: 'grade_3', 1: 'grade_4', 2: 'grade_5', 3: 'grade_6
                   6: 'grade_9', 7: 'grade_10', 8: 'grade_11', 9: 'grade_
                  }, inplace = True)

```

In [121]:

```
#sanity check
df.columns
```

Out[121]:

```
Index(['price', 'bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floor
s',
       'sqft_above', 'yr_built', 'yr_renovated', 'zipcode', 'lat', 'long',
       'sqft_living15', 'sqft_lot15', 'numerical_grade', 'view_num',
       'Population', 'number_of_households', 'median_income', 'average_inco
me',
       'Waterfront', 'renovated', 'grade_3', 'grade_4', 'grade_5', 'grade_
6',
       'grade_7', 'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grade_12',
       'grade_13'],
      dtype='object')
```

One limitation of the function used is that whenever new columns are added, we have to reset the train-test split

In [122]:

```
y = df['price']
X = df.drop(columns=['price'], axis=1)


X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=.2,
                                                    random_state=42
)

```

In [123]:

```
model_and_assess(['sqft_living', 'median_income','view_num', 'Waterfront
                     'grade_3', 'grade_4', 'grade_5', 'grade_6', 'grade_
                     'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grad
                     'grade_13', 'renovated'], df)
```

```
Train R2: 0.6579508928418821
Test R2: 0.6320766773376292
----
Train RMSE: 215784.983697498
Test RMSE: 218882.63166376975
----
Train MAE: 140456.2658171113
Test MAE: 139838.85552357975

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
653
Model:                            OLS   Adj. R-squared:                  0.
653
Method:                 Least Squares   F-statistic:                     29
01.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:22   Log-Likelihood:            -2.9596e
+05
No. Observations:               21597   AIC:                        5.920e
+05
Df Residuals:                   21582   BIC:                        5.921e
+05
Df Model:                          14
Covariance Type:            nonrobust
===========================================================================
======
                    coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept      2.573e+05   2.11e+04     12.208      0.000    2.16e+05      2.
99e+05
sqft_living     134.2730      2.565     52.347      0.000     129.245       1
39.301
median_income     2.0572      0.052     39.287      0.000       1.955
2.160
view_num        7.08e+04   2173.470     32.574      0.000    6.65e+04      7.
51e+04
Waterfront     6.025e+05   1.95e+04     30.925      0.000    5.64e+05      6.
41e+05
grade_3       -3.215e+05   1.99e+05     -1.619      0.105   -7.11e+05      6.
78e+04
grade_4       -4.266e+05   4.29e+04     -9.948      0.000   -5.11e+05     -3.
43e+05
grade_5       -4.262e+05   2.34e+04    -18.222      0.000   -4.72e+05      -
```

```
3.8e+05
grade_6       -3.874e+05      2e+04    -19.351      0.000   -4.27e+05   -3.
48e+05
grade_7       -3.697e+05   1.95e+04    -19.001      0.000   -4.08e+05   -3.
32e+05
grade_8       -3.308e+05   1.94e+04    -17.046      0.000   -3.69e+05   -2.
93e+05
grade_9       -2.316e+05   1.96e+04    -11.796      0.000    -2.7e+05   -1.
93e+05
grade_10      -7.235e+04   2.03e+04     -3.567      0.000   -1.12e+05   -3.
26e+04
grade_11       1.916e+05   2.21e+04      8.650      0.000    1.48e+05    2.
35e+05
grade_12        6.66e+05   2.94e+04     22.623      0.000    6.08e+05    7.
24e+05
grade_13       1.966e+06   5.94e+04     33.081      0.000    1.85e+06    2.
08e+06
============================================================================
===
Omnibus:                    13364.232   Durbin-Watson:                    1.
976
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            456517.
394
Skew:                           2.437   Prob(JB):
0.00
Kurtosis:                      24.990   Cond. No.                       1.95e
+19
============================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 7.34e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the summary of the scaled model

                            OLS Regression Results
============================================================================
===
Dep. Variable:                  price   R-squared:                        0.
653
Model:                            OLS   Adj. R-squared:                   0.
653
Method:                 Least Squares   F-statistic:                      29
00.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:22   Log-Likelihood:                 -192
17.
No. Observations:               21597   AIC:                           3.846e
+04
Df Residuals:                   21582   BIC:                           3.858e
+04
Df Model:                          14
Covariance Type:            nonrobust
```

```
==============================================================================
======
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
------
Intercept   -6.353e-17      0.004  -1.58e-14      1.000      -0.008
0.008
sqft_living     0.3362      0.007     50.844      0.000       0.323
0.349
median_income   0.1724      0.004     39.207      0.000       0.164
0.181
view_num        0.1474      0.005     32.575      0.000       0.139
0.156
Waterfront      0.1343      0.004     30.888      0.000       0.126
0.143
grade_3      2.751e+09    7.3e+09      0.377      0.706   -1.16e+10    1.
71e+10
grade_4      1.429e+10   3.79e+10      0.377      0.706   -6.01e+10    8.
86e+10
grade_5      4.256e+10   1.13e+11      0.377      0.706   -1.79e+11    2.
64e+11
grade_6      1.182e+11   3.14e+11      0.377      0.706   -4.97e+11    7.
33e+11
grade_7      1.993e+11   5.29e+11      0.377      0.706   -8.38e+11    1.
24e+12
grade_8      1.817e+11   4.82e+11      0.377      0.706   -7.64e+11    1.
13e+12
grade_9      1.319e+11    3.5e+11      0.377      0.706   -5.54e+11    8.
18e+11
grade_10     9.019e+10   2.39e+11      0.377      0.706   -3.79e+11    5.
59e+11
grade_11     5.445e+10   1.45e+11      0.377      0.706   -2.29e+11    3.
38e+11
grade_12      2.59e+10   6.88e+10      0.377      0.706   -1.09e+11    1.
61e+11
grade_13     9.917e+09   2.63e+10      0.377      0.706   -4.17e+10    6.
15e+10
==============================================================================
===
Omnibus:                    13364.361   Durbin-Watson:                   1.
977
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           454355.
323
Skew:                           2.439   Prob(JB):
0.00
Kurtosis:                      24.934   Cond. No.                      3.48e
+14
==============================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 4.18e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

This is the correlation table between variables

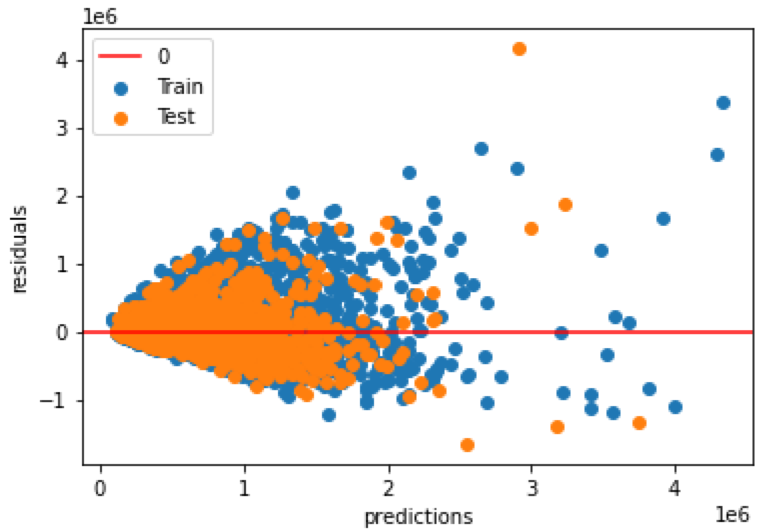
	sqft_living	median_income	view_num	Waterfront	grade_3
sqft_living	1.000000	0.337315	0.281715	0.104637	-0.011565
median_income	0.337315	1.000000	0.038370	0.002275	-0.004238
view_num	0.281715	0.038370	1.000000	0.380543	-0.002075
Waterfront	0.104637	0.002275	0.380543	1.000000	-0.000561
grade_3	-0.011565	-0.004238	-0.002075	-0.000561	1.000000
grade_4	-0.053935	-0.009219	-0.003934	-0.002919	-0.000241
grade_5	-0.127198	-0.050634	-0.013479	0.012691	-0.000724
grade_6	-0.312486	-0.169862	-0.059287	-0.007301	-0.002197
grade_7	-0.358915	-0.215072	-0.147272	-0.045482	-0.005738
grade_8	0.071115	0.065800	0.010612	-0.011317	-0.004252
grade_9	0.318499	0.185681	0.094153	0.007487	-0.002526
grade_10	0.369228	0.215769	0.127753	0.051514	-0.001602
grade_11	0.345964	0.134906	0.140282	0.068410	-0.000934
grade_12	0.238136	0.068073	0.114607	0.082899	-0.000438
grade_13	0.144424	0.021730	0.051769	-0.002025	-0.000167

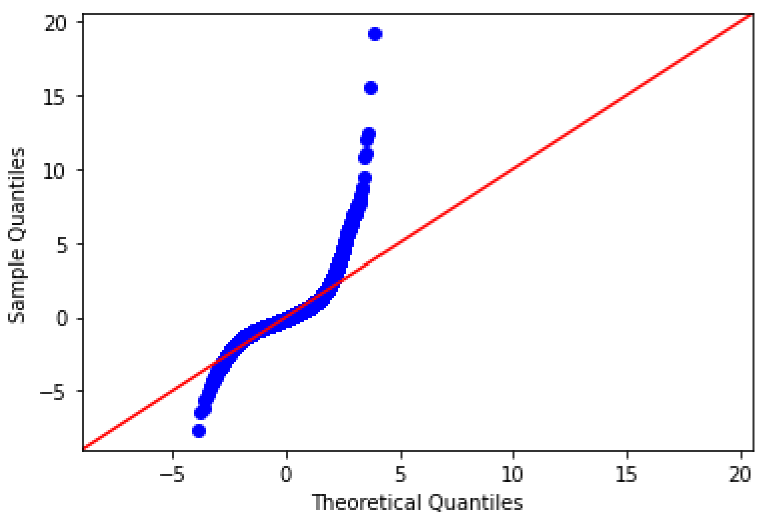
	grade_4	grade_5	grade_6	grade_7	grade_8	grade_9
sqft_living	-0.053935	-0.127198	-0.312486	-0.358915	0.071115	0.318499
median_income	-0.009219	-0.050634	-0.169862	-0.215072	0.065800	0.185681
view_num	-0.003934	-0.013479	-0.059287	-0.147272	0.010612	0.094153
Waterfront	-0.002919	0.012691	-0.007301	-0.045482	-0.011317	0.007487
grade_3	-0.000241	-0.000724	-0.002197	-0.005738	-0.004252	-0.002526
grade_4	1.000000	-0.003766	-0.011421	-0.029831	-0.022108	-0.013132
grade_5	-0.003766	1.000000	-0.034363	-0.089757	-0.066521	-0.039511
grade_6	-0.011421	-0.034363	1.000000	-0.272170	-0.201711	-0.119810
grade_7	-0.029831	-0.089757	-0.272170	1.000000	-0.526882	-0.312951
grade_8	-0.022108	-0.066521	-0.201711	-0.526882	1.000000	-0.231935
grade_9	-0.013132	-0.039511	-0.119810	-0.312951	-0.231935	1.000000
grade_10	-0.008329	-0.025060	-0.075989	-0.198488	-0.147104	-0.087375
grade_11	-0.004854	-0.014605	-0.044286	-0.115678	-0.085732	-0.050922
grade_12	-0.002276	-0.006848	-0.020765	-0.054238	-0.040197	-0.023876
grade_13	-0.000868	-0.002613	-0.007922	-0.020693	-0.015336	-0.009109

	grade_10	grade_11	grade_12	grade_13
sqft_living	0.369228	0.345964	0.238136	0.144424
median_income	0.215769	0.134906	0.068073	0.021730
view_num	0.127753	0.140282	0.114607	0.051769
Waterfront	0.051514	0.068410	0.082899	-0.002025
grade_3	-0.001602	-0.000934	-0.000438	-0.000167
grade_4	-0.008329	-0.004854	-0.002276	-0.000868
grade_5	-0.025060	-0.014605	-0.006848	-0.002613
grade_6	-0.075989	-0.044286	-0.020765	-0.007922
grade_7	-0.198488	-0.115678	-0.054238	-0.020693
grade_8	-0.147104	-0.085732	-0.040197	-0.015336
grade_9	-0.087375	-0.050922	-0.023876	-0.009109
grade_10	1.000000	-0.032297	-0.015143	-0.005777
grade_11	-0.032297	1.000000	-0.008825	-0.003367
grade_12	-0.015143	-0.008825	1.000000	-0.001579
grade_13	-0.005777	-0.003367	-0.001579	1.000000

This is the residual plot and qq plot

```
None
Figure(432x288)
```





Once again the model improved, but using another proxy for school district make help us imrpove it farther.

One way to improve it farther without creating multi-collinearity is to use some of the locational data. We thusly wanted to see the model with latitude, longitude, and both added.

In [124]:

```
model_and_assess(['sqft_living', 'median_income','view_num', 'Waterfront
                  'grade_3', 'grade_4', 'grade_5', 'grade_6', 'grade_
                  'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grad
                  'grade_13', 'long', 'renovated'], df)
```

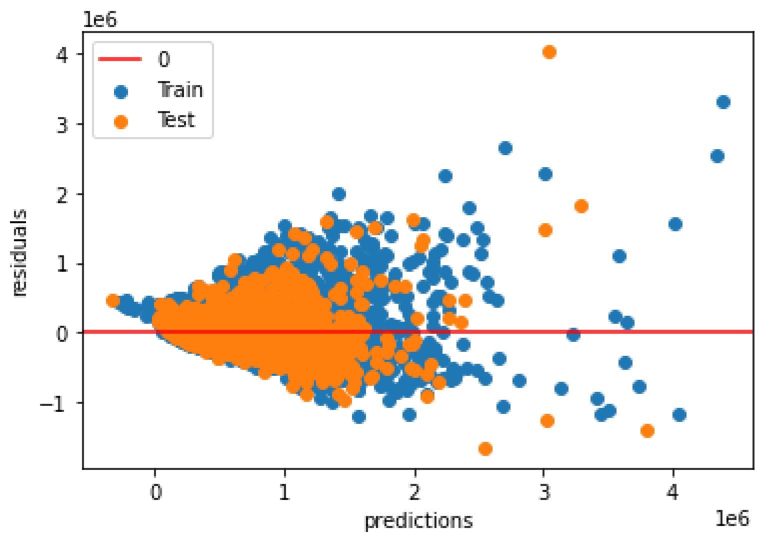
```
Train R2: 0.7020826026817109
Test R2: 0.6745321834053045
----
Train RMSE: 201383.98072491345
Test RMSE: 205866.96766016827
----
Train MAE: 127537.7141849933
Test MAE: 126660.88941980782

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
697
Model:                            OLS   Adj. R-squared:                  0.
697
Method:                 Least Squares   F-statistic:                     33
06.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:23   Log-Likelihood:            -2.9451e
+05
No. Observations:               21597   AIC:                         5.890e
+05
Df Residuals:                   21581   BIC:                         5.892e
+05
Df Model:                          15
Covariance Type:            nonrobust
===========================================================================
======
                    coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept      -7.12e+07   1.28e+06    -55.630      0.000   -7.37e+07     -6.
87e+07
sqft_living     153.4639      2.422     63.356      0.000     148.716       1
58.212
median_income     3.3773      0.054     62.130      0.000       3.271
3.484
view_num       5.618e+04   2048.512     27.424      0.000    5.22e+04      6.
02e+04
Waterfront     5.992e+05   1.82e+04     32.901      0.000    5.64e+05      6.
35e+05
grade_3       -6.633e+06   2.17e+05    -30.514      0.000   -7.06e+06     -6.
21e+06
grade_4       -6.862e+06   1.22e+05    -56.234      0.000    -7.1e+06     -6.
62e+06
grade_5        -6.88e+06   1.18e+05    -58.487      0.000   -7.11e+06     -6.
```

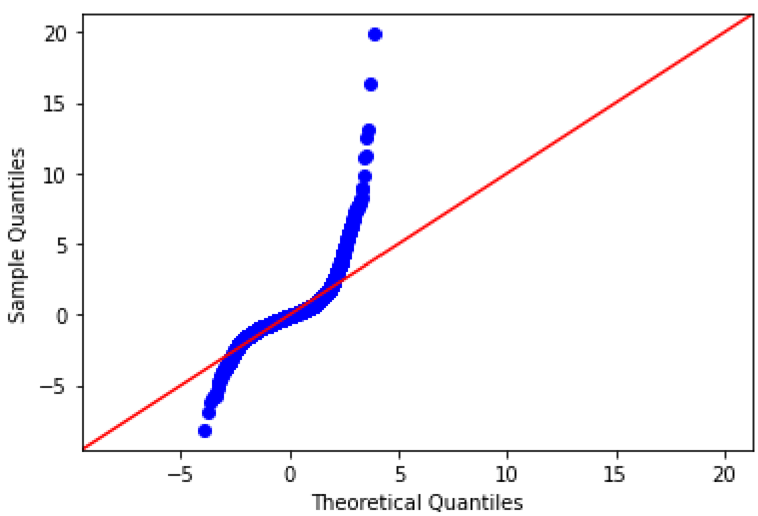
```
65e+06
grade_6       -6.884e+06   1.18e+05    -58.416      0.000   -7.12e+06   -6.
65e+06
grade_7       -6.868e+06   1.18e+05    -58.306      0.000    -7.1e+06   -6.
64e+06
grade_8       -6.835e+06   1.18e+05    -57.978      0.000   -7.07e+06    -
6.6e+06
grade_9       -6.736e+06   1.18e+05    -57.121      0.000   -6.97e+06    -
6.5e+06
grade_10      -6.594e+06   1.18e+05    -55.727      0.000   -6.83e+06   -6.
36e+06
grade_11      -6.343e+06   1.19e+05    -53.372      0.000   -6.58e+06   -6.
11e+06
grade_12       -5.88e+06    1.2e+05    -48.829      0.000   -6.12e+06   -5.
64e+06
grade_13      -4.684e+06   1.31e+05    -35.641      0.000   -4.94e+06   -4.
43e+06
long          -6.365e+05   1.14e+04    -55.837      0.000   -6.59e+05   -6.
14e+05
===========================================================================
===
Omnibus:                    13836.203   Durbin-Watson:                   1.
975
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           587501.
087
Skew:                           2.494   Prob(JB):
0.00
Kurtosis:                      28.060   Cond. No.                      1.95e
+19
===========================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 7.34e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the summary of the scaled model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
697
Model:                            OLS   Adj. R-squared:                  0.
697
Method:                 Least Squares   F-statistic:                     33
05.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:23   Log-Likelihood:                -177
61.
No. Observations:               21597   AIC:                         3.555e
+04
Df Residuals:                   21581   BIC:                         3.568e
+04
```

```
Df Model:                          15
Covariance Type:            nonrobust
===========================================================================
======
                 coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept        0.0012      0.005      0.234      0.815      -0.009
0.011
sqft_living      0.3840      0.006     61.650      0.000       0.372
0.396
median_income    0.2830      0.005     61.982      0.000       0.274
0.292
view_num         0.1170      0.004     27.409      0.000       0.109
0.125
Waterfront       0.1336      0.004     32.877      0.000       0.126
0.142
grade_3        2.36e+09   6.83e+09      0.346      0.730    -1.1e+10    1.
57e+10
grade_4       1.225e+10   3.54e+10      0.346      0.730   -5.72e+10    8.
17e+10
grade_5       3.651e+10   1.06e+11      0.346      0.730    -1.7e+11    2.
43e+11
grade_6       1.014e+11   2.93e+11      0.346      0.730   -4.73e+11    6.
76e+11
grade_7       1.709e+11   4.94e+11      0.346      0.730   -7.98e+11    1.
14e+12
grade_8       1.559e+11   4.51e+11      0.346      0.730   -7.28e+11    1.
04e+12
grade_9       1.131e+11   3.27e+11      0.346      0.730   -5.28e+11    7.
55e+11
grade_10      7.736e+10   2.24e+11      0.346      0.730   -3.61e+11    5.
16e+11
grade_11       4.67e+10   1.35e+11      0.346      0.730   -2.18e+11    3.
11e+11
grade_12      2.222e+10   6.43e+10      0.346      0.730   -1.04e+11    1.
48e+11
grade_13      8.506e+09   2.46e+10      0.346      0.730   -3.97e+10    5.
67e+10
long            -0.2440      0.004    -55.653      0.000      -0.253
-0.235
===========================================================================
===
Omnibus:                    13828.125   Durbin-Watson:                   1.
975
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           584428.
272
Skew:                           2.494   Prob(JB):
0.00
Kurtosis:                      27.992   Cond. No.                      3.58e
+14
===========================================================================
===

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

```
ectly specified.
[2] The smallest eigenvalue is 4.18e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the correlation table between variables

               sqft_living  median_income  view_num  Waterfront   grade_3  \
sqft_living       1.000000       0.337315  0.281715    0.104637 -0.011565
median_income     0.337315       1.000000  0.038370    0.002275 -0.004238
view_num          0.281715       0.038370  1.000000    0.380543 -0.002075
Waterfront        0.104637       0.002275  0.380543    1.000000 -0.000561
grade_3          -0.011565      -0.004238 -0.002075   -0.000561  1.000000
grade_4          -0.053935      -0.009219 -0.003934   -0.002919 -0.000241
grade_5          -0.127198      -0.050634 -0.013479    0.012691 -0.000724
grade_6          -0.312486      -0.169862 -0.059287   -0.007301 -0.002197
grade_7          -0.358915      -0.215072 -0.147272   -0.045482 -0.005738
grade_8           0.071115       0.065800  0.010612   -0.011317 -0.004252
grade_9           0.318499       0.185681  0.094153    0.007487 -0.002526
grade_10          0.369228       0.215769  0.127753    0.051514 -0.001602
grade_11          0.345964       0.134906  0.140282    0.068410 -0.000934
grade_12          0.238136       0.068073  0.114607    0.082899 -0.000438
grade_13          0.144424       0.021730  0.051769   -0.002025 -0.000167
long              0.241214       0.482594 -0.077702   -0.037628  0.010589

                grade_4   grade_5   grade_6   grade_7   grade_8   grade_9  \
sqft_living   -0.053935 -0.127198 -0.312486 -0.358915  0.071115  0.318499
median_income -0.009219 -0.050634 -0.169862 -0.215072  0.065800  0.185681
view_num      -0.003934 -0.013479 -0.059287 -0.147272  0.010612  0.094153
Waterfront    -0.002919  0.012691 -0.007301 -0.045482 -0.011317  0.007487
grade_3       -0.000241 -0.000724 -0.002197 -0.005738 -0.004252 -0.002526
grade_4        1.000000 -0.003766 -0.011421 -0.029831 -0.022108 -0.013132
grade_5       -0.003766  1.000000 -0.034363 -0.089757 -0.066521 -0.039511
grade_6       -0.011421 -0.034363  1.000000 -0.272170 -0.201711 -0.119810
grade_7       -0.029831 -0.089757 -0.272170  1.000000 -0.526882 -0.312951
grade_8       -0.022108 -0.066521 -0.201711 -0.526882  1.000000 -0.231935
grade_9       -0.013132 -0.039511 -0.119810 -0.312951 -0.231935  1.000000
grade_10      -0.008329 -0.025060 -0.075989 -0.198488 -0.147104 -0.087375
grade_11      -0.004854 -0.014605 -0.044286 -0.115678 -0.085732 -0.050922
grade_12      -0.002276 -0.006848 -0.020765 -0.054238 -0.040197 -0.023876
grade_13      -0.000868 -0.002613 -0.007922 -0.020693 -0.015336 -0.009109
long           0.012278  0.011084 -0.111258 -0.113233  0.026554  0.126992

               grade_10  grade_11  grade_12  grade_13      long
sqft_living    0.369228  0.345964  0.238136  0.144424  0.241214
median_income  0.215769  0.134906  0.068073  0.021730  0.482594
view_num       0.127753  0.140282  0.114607  0.051769 -0.077702
Waterfront     0.051514  0.068410  0.082899 -0.002025 -0.037628
grade_3       -0.001602 -0.000934 -0.000438 -0.000167  0.010589
grade_4       -0.008329 -0.004854 -0.002276 -0.000868  0.012278
grade_5       -0.025060 -0.014605 -0.006848 -0.002613  0.011084
grade_6       -0.075989 -0.044286 -0.020765 -0.007922 -0.111258
grade_7       -0.198488 -0.115678 -0.054238 -0.020693 -0.113233
grade_8       -0.147104 -0.085732 -0.040197 -0.015336  0.026554
grade_9       -0.087375 -0.050922 -0.023876 -0.009109  0.126992
grade_10       1.000000 -0.032297 -0.015143 -0.005777  0.103756
```

```
grade_11      -0.032297  1.000000 -0.008825 -0.003367  0.061840
grade_12      -0.015143 -0.008825  1.000000 -0.001579  0.031744
grade_13      -0.005777 -0.003367 -0.001579  1.000000 -0.008562
long           0.103756  0.061840  0.031744 -0.008562  1.000000

This is the residual plot and qq plot

None
Figure(432x288)
```





In [125]:

```
model_and_assess(['sqft_living', 'median_income','view_num', 'Waterfront
                     'grade_3', 'grade_4', 'grade_5', 'grade_6', 'grade_
                     'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grad
                     'grade_13', 'lat', 'renovated'], df)
```

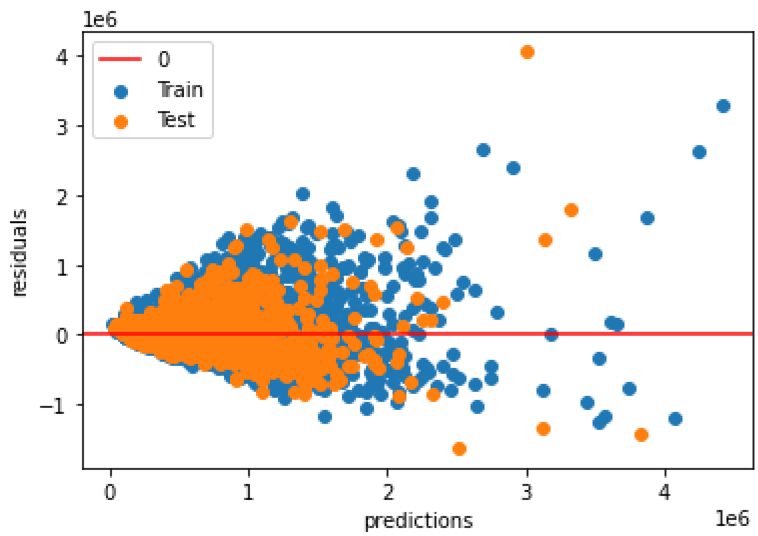
```
Train R2: 0.7019463457743972
Test R2: 0.6786339187739434
----
Train RMSE: 201430.0284244488
Test RMSE: 204565.62676040918
----
Train MAE: 124064.40659388344
Test MAE: 123106.3960231164

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
697
Model:                            OLS   Adj. R-squared:                  0.
697
Method:                 Least Squares   F-statistic:                     33
17.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:24   Log-Likelihood:             -2.9448e
+05
No. Observations:               21597   AIC:                         5.890e
+05
Df Residuals:                   21581   BIC:                         5.891e
+05
Df Model:                          15
Covariance Type:            nonrobust
===========================================================================
======
                   coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept    -2.611e+07   4.68e+05    -55.734      0.000    -2.7e+07   -2.
52e+07
sqft_living    145.6088      2.403     60.582      0.000     140.898      1
50.320
median_income    0.9598      0.053     18.236      0.000       0.857
1.063
view_num      6.866e+04   2029.781     33.824      0.000    6.47e+04     7.
26e+04
Waterfront    6.136e+05   1.82e+04     33.726      0.000    5.78e+05     6.
49e+05
grade_3      -2.525e+06    1.9e+05    -13.320      0.000    -2.9e+06    -2.
15e+06
grade_4      -2.794e+06    5.8e+04    -48.133      0.000   -2.91e+06    -2.
68e+06
grade_5      -2.806e+06   4.76e+04    -59.004      0.000    -2.9e+06    -2.
```

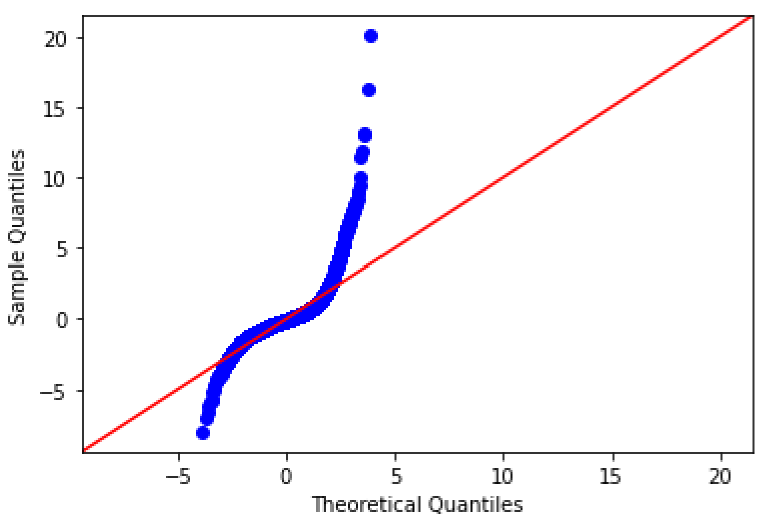
```
71e+06
grade_6       -2.792e+06   4.66e+04    -59.915      0.000   -2.88e+06    -
2.7e+06
grade_7       -2.783e+06   4.65e+04    -59.807      0.000   -2.87e+06   -2.
69e+06
grade_8       -2.745e+06   4.65e+04    -58.995      0.000   -2.84e+06   -2.
65e+06
grade_9       -2.646e+06   4.66e+04    -56.763      0.000   -2.74e+06   -2.
55e+06
grade_10      -2.489e+06   4.69e+04    -53.077      0.000   -2.58e+06    -
2.4e+06
grade_11      -2.237e+06   4.78e+04    -46.787      0.000   -2.33e+06   -2.
14e+06
grade_12       -1.77e+06   5.12e+04    -34.545      0.000   -1.87e+06   -1.
67e+06
grade_13      -5.197e+05   7.09e+04     -7.330      0.000   -6.59e+05   -3.
81e+05
lat            6.071e+05   1.08e+04     56.333      0.000    5.86e+05    6.
28e+05
===========================================================================
===
Omnibus:                    14707.135   Durbin-Watson:                   1.
991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           675619.
487
Skew:                           2.713   Prob(JB):
0.00
Kurtosis:                      29.858   Cond. No.                     1.95e
+19
===========================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 7.34e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the summary of the scaled model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
697
Model:                            OLS   Adj. R-squared:                  0.
697
Method:                 Least Squares   F-statistic:                     33
16.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:24   Log-Likelihood:                -177
36.
No. Observations:               21597   AIC:                         3.550e
+04
Df Residuals:                   21581   BIC:                         3.563e
+04
```

```
Df Model:                          15
Covariance Type:            nonrobust
===========================================================================
======
                 coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept      0.0015      0.005      0.295      0.768      -0.008
0.011
sqft_living    0.3645      0.006     59.348      0.000       0.352
0.376
median_income  0.0806      0.004     18.235      0.000       0.072
0.089
view_num       0.1429      0.004     33.809      0.000       0.135
0.151
Waterfront     0.1368      0.004     33.700      0.000       0.129
0.145
grade_3     2.967e+09   6.82e+09      0.435      0.663   -1.04e+10    1.
63e+10
grade_4     1.541e+10   3.54e+10      0.435      0.663    -5.4e+10    8.
48e+10
grade_5      4.59e+10   1.05e+11      0.435      0.663   -1.61e+11    2.
53e+11
grade_6     1.275e+11   2.93e+11      0.435      0.663   -4.47e+11    7.
02e+11
grade_7     2.149e+11   4.94e+11      0.435      0.663   -7.53e+11    1.
18e+12
grade_8     1.959e+11    4.5e+11      0.435      0.663   -6.87e+11    1.
08e+12
grade_9     1.422e+11   3.27e+11      0.435      0.663   -4.99e+11    7.
83e+11
grade_10    9.725e+10   2.24e+11      0.435      0.663   -3.41e+11    5.
35e+11
grade_11    5.871e+10   1.35e+11      0.435      0.663   -2.06e+11    3.
23e+11
grade_12    2.793e+10   6.42e+10      0.435      0.663   -9.79e+10    1.
54e+11
grade_13    1.069e+10   2.46e+10      0.435      0.663   -3.75e+10    5.
89e+10
lat            0.2289      0.004     56.266      0.000       0.221
0.237
===========================================================================
===
Omnibus:                    14695.214   Durbin-Watson:                   1.
990
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           674133.
950
Skew:                           2.710   Prob(JB):
0.00
Kurtosis:                      29.828   Cond. No.                      3.51e
+14
===========================================================================
===

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

```
ectly specified.
[2] The smallest eigenvalue is 4.18e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the correlation table between variables

               sqft_living  median_income  view_num  Waterfront   grade_3
\
sqft_living       1.000000       0.337315  0.281715    0.104637 -0.011565
median_income     0.337315       1.000000  0.038370    0.002275 -0.004238
view_num          0.281715       0.038370  1.000000    0.380543 -0.002075
Waterfront        0.104637       0.002275  0.380543    1.000000 -0.000561
grade_3          -0.011565      -0.004238 -0.002075   -0.000561  1.000000
grade_4          -0.053935      -0.009219 -0.003934   -0.002919 -0.000241
grade_5          -0.127198      -0.050634 -0.013479    0.012691 -0.000724
grade_6          -0.312486      -0.169862 -0.059287   -0.007301 -0.002197
grade_7          -0.358915      -0.215072 -0.147272   -0.045482 -0.005738
grade_8           0.071115       0.065800  0.010612   -0.011317 -0.004252
grade_9           0.318499       0.185681  0.094153    0.007487 -0.002526
grade_10          0.369228       0.215769  0.127753    0.051514 -0.001602
grade_11          0.345964       0.134906  0.140282    0.068410 -0.000934
grade_12          0.238136       0.068073  0.114607    0.082899 -0.000438
grade_13          0.144424       0.021730  0.051769   -0.002025 -0.000167
lat               0.052155       0.375826  0.006321   -0.012157 -0.017283

                grade_4   grade_5   grade_6   grade_7   grade_8   grade_9
\
sqft_living   -0.053935 -0.127198 -0.312486 -0.358915  0.071115  0.318499
median_income -0.009219 -0.050634 -0.169862 -0.215072  0.065800  0.185681
view_num      -0.003934 -0.013479 -0.059287 -0.147272  0.010612  0.094153
Waterfront    -0.002919  0.012691 -0.007301 -0.045482 -0.011317  0.007487
grade_3       -0.000241 -0.000724 -0.002197 -0.005738 -0.004252 -0.002526
grade_4        1.000000 -0.003766 -0.011421 -0.029831 -0.022108 -0.013132
grade_5       -0.003766  1.000000 -0.034363 -0.089757 -0.066521 -0.039511
grade_6       -0.011421 -0.034363  1.000000 -0.272170 -0.201711 -0.119810
grade_7       -0.029831 -0.089757 -0.272170  1.000000 -0.526882 -0.312951
grade_8       -0.022108 -0.066521 -0.201711 -0.526882  1.000000 -0.231935
grade_9       -0.013132 -0.039511 -0.119810 -0.312951 -0.231935  1.000000
grade_10      -0.008329 -0.025060 -0.075989 -0.198488 -0.147104 -0.087375
grade_11      -0.004854 -0.014605 -0.044286 -0.115678 -0.085732 -0.050922
grade_12      -0.002276 -0.006848 -0.020765 -0.054238 -0.040197 -0.023876
grade_13      -0.000868 -0.002613 -0.007922 -0.020693 -0.015336 -0.009109
lat           -0.016323 -0.046573 -0.062851 -0.040532  0.026330  0.042136

               grade_10  grade_11  grade_12  grade_13       lat
sqft_living    0.369228  0.345964  0.238136  0.144424  0.052155
median_income  0.215769  0.134906  0.068073  0.021730  0.375826
view_num       0.127753  0.140282  0.114607  0.051769  0.006321
Waterfront     0.051514  0.068410  0.082899 -0.002025 -0.012157
grade_3       -0.001602 -0.000934 -0.000438 -0.000167 -0.017283
grade_4       -0.008329 -0.004854 -0.002276 -0.000868 -0.016323
grade_5       -0.025060 -0.014605 -0.006848 -0.002613 -0.046573
grade_6       -0.075989 -0.044286 -0.020765 -0.007922 -0.062851
grade_7       -0.198488 -0.115678 -0.054238 -0.020693 -0.040532
grade_8       -0.147104 -0.085732 -0.040197 -0.015336  0.026330
grade_9       -0.087375 -0.050922 -0.023876 -0.009109  0.042136
grade_10       1.000000 -0.032297 -0.015143 -0.005777  0.052262
```

```
grade_11      -0.032297  1.000000 -0.008825 -0.003367  0.039372
grade_12      -0.015143 -0.008825  1.000000 -0.001579  0.016946
grade_13      -0.005777 -0.003367 -0.001579  1.000000  0.013142
lat            0.052262  0.039372  0.016946  0.013142  1.000000

This is the residual plot and qq plot

None
Figure(432x288)
```





In [126]:

```
model_and_assess(['sqft_living', 'median_income','view_num', 'Waterfront
                  'grade_3', 'grade_4', 'grade_5', 'grade_6', 'grade_
                  'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grad
                  'grade_13', 'long', 'lat', 'renovated'], df)
```

```
Train R2: 0.7218344833703131
Test R2: 0.6954068078694756
----
Train RMSE: 194593.63646492016
Test RMSE: 199155.6989571329
----
Train MAE: 120193.77376613383
Test MAE: 120193.87414294749

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
717
Model:                            OLS   Adj. R-squared:                  0.
717
Method:                 Least Squares   F-statistic:                     34
13.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:24   Log-Likelihood:            -2.9377e
+05
No. Observations:               21597   AIC:                          5.876e
+05
Df Residuals:                   21580   BIC:                          5.877e
+05
Df Model:                          16
Covariance Type:            nonrobust
===========================================================================
======
                   coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept     -7.024e+07   1.24e+06    -56.767      0.000   -7.27e+07   -6.
78e+07
sqft_living     156.2939      2.342     66.726      0.000     151.703     1
60.885
median_income     2.2097      0.060     36.541      0.000       2.091
2.328
view_num       5.873e+04   1981.045     29.648      0.000    5.49e+04     6.
26e+04
Waterfront     6.082e+05   1.76e+04     34.546      0.000    5.74e+05     6.
43e+05
grade_3       -6.457e+06    2.1e+05    -30.725      0.000   -6.87e+06    -6.
04e+06
grade_4        -6.77e+06   1.18e+05    -57.392      0.000       -7e+06   -6.
54e+06
grade_5       -6.792e+06   1.14e+05    -59.729      0.000   -7.02e+06    -6.
```

```
57e+06
grade_6       -6.802e+06   1.14e+05    -59.709      0.000   -7.03e+06   -6.
58e+06
grade_7       -6.792e+06   1.14e+05    -59.648      0.000   -7.02e+06   -6.
57e+06
grade_8       -6.758e+06   1.14e+05    -59.299      0.000   -6.98e+06   -6.
53e+06
grade_9       -6.659e+06   1.14e+05    -58.413      0.000   -6.88e+06   -6.
44e+06
grade_10      -6.513e+06   1.14e+05    -56.944      0.000   -6.74e+06   -6.
29e+06
grade_11      -6.268e+06   1.15e+05    -54.559      0.000   -6.49e+06   -6.
04e+06
grade_12      -5.807e+06   1.16e+05    -49.887      0.000   -6.04e+06   -5.
58e+06
grade_13      -4.618e+06   1.27e+05    -36.352      0.000   -4.87e+06   -4.
37e+06
long          -4.575e+05   1.19e+04    -38.332      0.000   -4.81e+05   -4.
34e+05
lat            4.406e+05   1.13e+04     39.003      0.000    4.18e+05    4.
63e+05
===========================================================================
===
Omnibus:                    14769.316   Durbin-Watson:                   1.
985
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           736858.
002
Skew:                           2.703   Prob(JB):
0.00
Kurtosis:                      31.100   Cond. No.                     1.95e
+19
===========================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 7.35e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the summary of the scaled model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
717
Model:                            OLS   Adj. R-squared:                  0.
717
Method:                 Least Squares   F-statistic:                     34
13.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:25   Log-Likelihood:                -170
24.
No. Observations:               21597   AIC:                          3.408e
+04
```

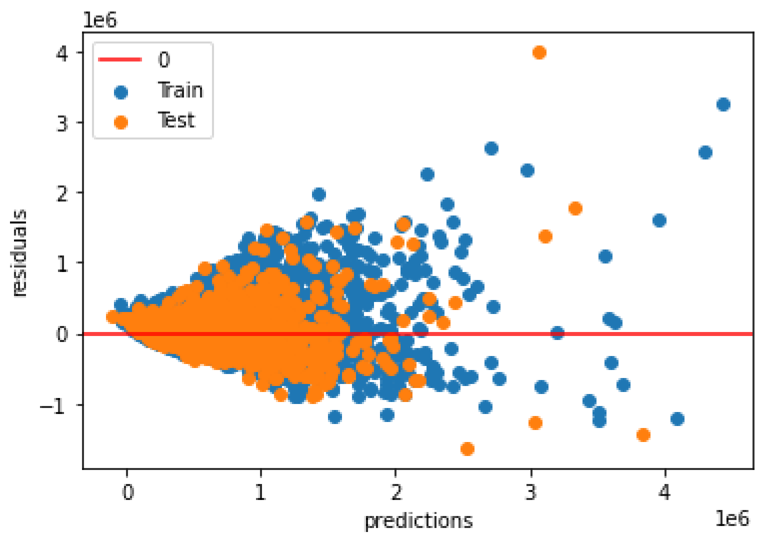
```
Df Residuals:                   21580   BIC:                             3.422e
+04
Df Model:                          16
Covariance Type:            nonrobust
==============================================================================
======
                  coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
------
Intercept       0.0013      0.005      0.270      0.787      -0.008
0.011
sqft_living     0.3910      0.006     65.700      0.000       0.379
0.403
median_income   0.1853      0.005     36.542      0.000       0.175
0.195
view_num        0.1224      0.004     29.530      0.000       0.114
0.131
Waterfront      0.1357      0.004     34.543      0.000       0.128
0.143
grade_3      2.626e+09    6.6e+09      0.398      0.691   -1.03e+10    1.
56e+10
grade_4      1.364e+10   3.43e+10      0.398      0.691   -5.35e+10    8.
08e+10
grade_5      4.062e+10   1.02e+11      0.398      0.691   -1.59e+11    2.
41e+11
grade_6      1.128e+11   2.83e+11      0.398      0.691   -4.43e+11    6.
68e+11
grade_7      1.902e+11   4.78e+11      0.398      0.691   -7.46e+11    1.
13e+12
grade_8      1.734e+11   4.36e+11      0.398      0.691   -6.81e+11    1.
03e+12
grade_9      1.259e+11   3.16e+11      0.398      0.691   -4.94e+11    7.
46e+11
grade_10     8.607e+10   2.16e+11      0.398      0.691   -3.38e+11
5.1e+11
grade_11     5.197e+10   1.31e+11      0.398      0.691   -2.04e+11    3.
08e+11
grade_12     2.472e+10   6.21e+10      0.398      0.691    -9.7e+10    1.
46e+11
grade_13     9.465e+09   2.38e+10      0.398      0.691   -3.71e+10    5.
61e+10
long           -0.1754      0.005    -38.247      0.000      -0.184
-0.166
lat             0.1662      0.004     39.004      0.000       0.158
0.175
==============================================================================
===
Omnibus:                    14757.418   Durbin-Watson:                       1.
986
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               735190.
797
Skew:                           2.700   Prob(JB):
0.00
Kurtosis:                      31.068   Cond. No.                        3.60e
+14
==============================================================================
```

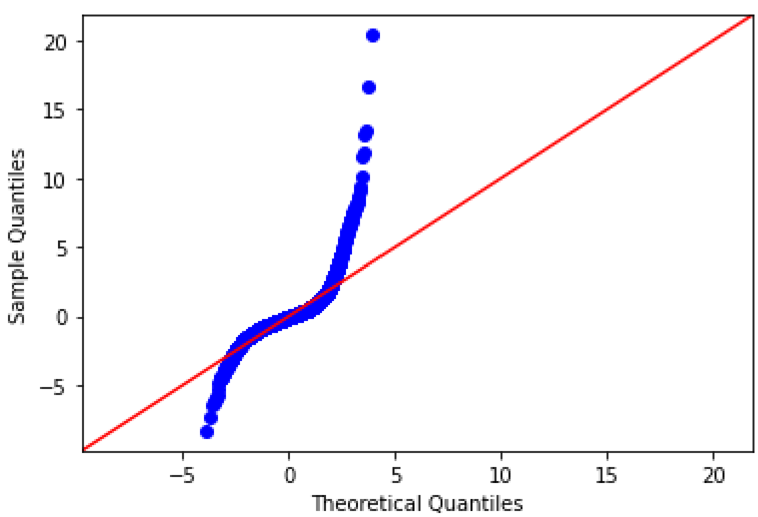
```
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 4.18e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the correlation table between variables

               sqft_living  median_income  view_num  Waterfront   grade_3  \
sqft_living       1.000000       0.337315  0.281715    0.104637 -0.011565
median_income     0.337315       1.000000  0.038370    0.002275 -0.004238
view_num          0.281715       0.038370  1.000000    0.380543 -0.002075
Waterfront        0.104637       0.002275  0.380543    1.000000 -0.000561
grade_3          -0.011565      -0.004238 -0.002075   -0.000561  1.000000
grade_4          -0.053935      -0.009219 -0.003934   -0.002919 -0.000241
grade_5          -0.127198      -0.050634 -0.013479    0.012691 -0.000724
grade_6          -0.312486      -0.169862 -0.059287   -0.007301 -0.002197
grade_7          -0.358915      -0.215072 -0.147272   -0.045482 -0.005738
grade_8           0.071115       0.065800  0.010612   -0.011317 -0.004252
grade_9           0.318499       0.185681  0.094153    0.007487 -0.002526
grade_10          0.369228       0.215769  0.127753    0.051514 -0.001602
grade_11          0.345964       0.134906  0.140282    0.068410 -0.000934
grade_12          0.238136       0.068073  0.114607    0.082899 -0.000438
grade_13          0.144424       0.021730  0.051769   -0.002025 -0.000167
long              0.241214       0.482594 -0.077702   -0.037628  0.010589
lat               0.052155       0.375826  0.006321   -0.012157 -0.017283

                grade_4   grade_5   grade_6   grade_7   grade_8   grade_9  \
sqft_living   -0.053935 -0.127198 -0.312486 -0.358915  0.071115  0.318499
median_income -0.009219 -0.050634 -0.169862 -0.215072  0.065800  0.185681
view_num      -0.003934 -0.013479 -0.059287 -0.147272  0.010612  0.094153
Waterfront    -0.002919  0.012691 -0.007301 -0.045482 -0.011317  0.007487
grade_3       -0.000241 -0.000724 -0.002197 -0.005738 -0.004252 -0.002526
grade_4        1.000000 -0.003766 -0.011421 -0.029831 -0.022108 -0.013132
grade_5       -0.003766  1.000000 -0.034363 -0.089757 -0.066521 -0.039511
grade_6       -0.011421 -0.034363  1.000000 -0.272170 -0.201711 -0.119810
grade_7       -0.029831 -0.089757 -0.272170  1.000000 -0.526882 -0.312951
grade_8       -0.022108 -0.066521 -0.201711 -0.526882  1.000000 -0.231935
grade_9       -0.013132 -0.039511 -0.119810 -0.312951 -0.231935  1.000000
grade_10      -0.008329 -0.025060 -0.075989 -0.198488 -0.147104 -0.087375
grade_11      -0.004854 -0.014605 -0.044286 -0.115678 -0.085732 -0.050922
grade_12      -0.002276 -0.006848 -0.020765 -0.054238 -0.040197 -0.023876
grade_13      -0.000868 -0.002613 -0.007922 -0.020693 -0.015336 -0.009109
long           0.012278  0.011084 -0.111258 -0.113233  0.026554  0.126992
lat           -0.016323 -0.046573 -0.062851 -0.040532  0.026330  0.042136

               grade_10  grade_11  grade_12  grade_13      long       lat
sqft_living    0.369228  0.345964  0.238136  0.144424  0.241214  0.052155
median_income  0.215769  0.134906  0.068073  0.021730  0.482594  0.375826
view_num       0.127753  0.140282  0.114607  0.051769 -0.077702  0.006321
Waterfront     0.051514  0.068410  0.082899 -0.002025 -0.037628 -0.012157
grade_3       -0.001602 -0.000934 -0.000438 -0.000167  0.010589 -0.017283
grade_4       -0.008329 -0.004854 -0.002276 -0.000868  0.012278 -0.016323
```

grade_5	-0.025060	-0.014605	-0.006848	-0.002613	0.011084	-0.046573
grade_6	-0.075989	-0.044286	-0.020765	-0.007922	-0.111258	-0.062851
grade_7	-0.198488	-0.115678	-0.054238	-0.020693	-0.113233	-0.040532
grade_8	-0.147104	-0.085732	-0.040197	-0.015336	0.026554	0.026330
grade_9	-0.087375	-0.050922	-0.023876	-0.009109	0.126992	0.042136
grade_10	1.000000	-0.032297	-0.015143	-0.005777	0.103756	0.052262
grade_11	-0.032297	1.000000	-0.008825	-0.003367	0.061840	0.039372
grade_12	-0.015143	-0.008825	1.000000	-0.001579	0.031744	0.016946
grade_13	-0.005777	-0.003367	-0.001579	1.000000	-0.008562	0.013142
long	0.103756	0.061840	0.031744	-0.008562	1.000000	-0.135371
lat	0.052262	0.039372	0.016946	0.013142	-0.135371	1.000000

This is the residual plot and qq plot

None
Figure(432x288)





Trying to understand what is happening here is difficult. It didn't make sense that houses should be more valuable moving from West to East or South to North. However, looking at the school districts within King County, most of the better schools were in the Northern part of the map. See [https://www.schooldigger.com/go/WA/county/King+County/search.aspx (https://www.schooldigger.com/go/WA/county/King+County/search.aspx)](https://www.schooldigger.com/go/WA/county/King+County/search.aspx) for more

information. We felt comfortable including Latitiude in our model for this reason but did not want to include longitude because we couldn't find a reason for this correlation.

Final model analysis

X Variables: Latitude, Square Feet Living, House Grade, Waterfront, View Grade, renovated

In [128]:

```
model_and_assess(['sqft_living', 'median_income','view_num', 'Waterfront
                     'grade_3', 'grade_4', 'grade_5', 'grade_6', 'grade_
                     'grade_8', 'grade_9', 'grade_10', 'grade_11', 'grad
                     'grade_13', 'lat', 'renovated'], df)
```

```
Train R2: 0.7059535130630036
Test R2: 0.6837844226516643
----
Train RMSE: 200071.3885125683
Test RMSE: 202919.72845490542
----
Train MAE: 123008.6987545759
Test MAE: 122693.92561063345

This is the summary of the model

                            OLS Regression Results
===========================================================================
===
Dep. Variable:                  price   R-squared:                       0.
702
Model:                            OLS   Adj. R-squared:                  0.
701
Method:                 Least Squares   F-statistic:                     31
73.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:39   Log-Likelihood:            -2.9433e
+05
No. Observations:               21597   AIC:                        5.887e
+05
Df Residuals:                   21580   BIC:                        5.888e
+05
Df Model:                          16
Covariance Type:            nonrobust
===========================================================================
======
                    coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept     -2.582e+07   4.65e+05    -55.484      0.000   -2.67e+07   -2.
49e+07
sqft_living     143.3718      2.390     59.985      0.000     138.687      1
48.057
median_income     0.9725      0.052     18.606      0.000       0.870
1.075
view_num       6.652e+04   2019.295     32.942      0.000    6.26e+04     7.
05e+04
Waterfront     5.994e+05   1.81e+04     33.144      0.000    5.64e+05     6.
35e+05
grade_3       -2.502e+06   1.88e+05    -13.292      0.000   -2.87e+06    -2.
13e+06
grade_4       -2.773e+06   5.76e+04    -48.109      0.000   -2.89e+06    -2.
66e+06
grade_5       -2.782e+06   4.72e+04    -58.886      0.000   -2.87e+06    -2.
```

```
69e+06
grade_6        -2.77e+06   4.63e+04    -59.841      0.000   -2.86e+06   -2.
68e+06
grade_7       -2.759e+06   4.62e+04    -59.690      0.000   -2.85e+06   -2.
67e+06
grade_8        -2.72e+06   4.62e+04    -58.856      0.000   -2.81e+06   -2.
63e+06
grade_9        -2.62e+06   4.63e+04    -56.585      0.000   -2.71e+06   -2.
53e+06
grade_10       -2.46e+06   4.66e+04    -52.810      0.000   -2.55e+06   -2.
37e+06
grade_11      -2.205e+06   4.75e+04    -46.409      0.000    -2.3e+06   -2.
11e+06
grade_12      -1.734e+06   5.09e+04    -34.046      0.000   -1.83e+06   -1.
63e+06
grade_13      -4.955e+05   7.04e+04     -7.037      0.000   -6.34e+05   -3.
57e+05
lat            6.007e+05   1.07e+04     56.095      0.000     5.8e+05    6.
22e+05
renovated      1.319e+05   7544.456     17.488      0.000    1.17e+05    1.
47e+05
==============================================================================
===
Omnibus:                    14534.882   Durbin-Watson:                     1.
989
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            648822.
011
Skew:                           2.674   Prob(JB):
0.00
Kurtosis:                      29.314   Cond. No.                      1.95e
+19
==============================================================================
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 7.33e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the summary of the scaled model

                            OLS Regression Results
==============================================================================
===
Dep. Variable:                  price   R-squared:                         0.
702
Model:                            OLS   Adj. R-squared:                    0.
701
Method:                 Least Squares   F-statistic:                      31
71.
Date:                Sun, 15 May 2022   Prob (F-statistic):
0.00
Time:                        20:52:39   Log-Likelihood:                 -175
86.
No. Observations:               21597   AIC:                           3.521e
+04
```

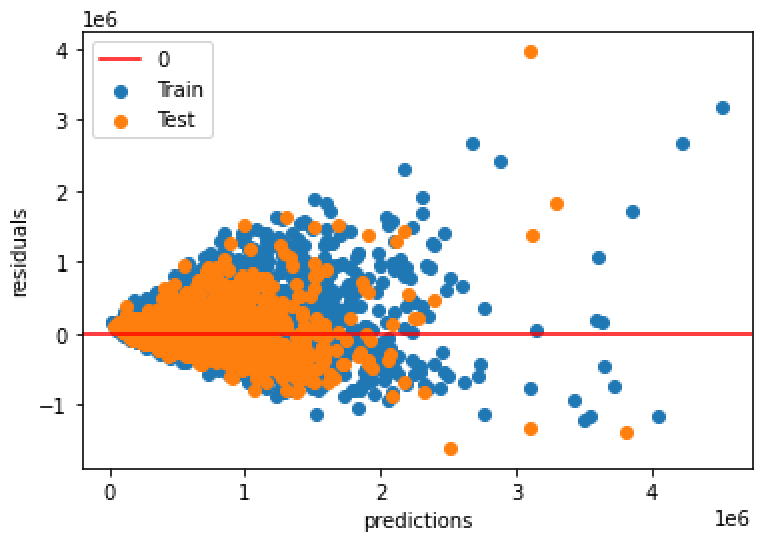
```
Df Residuals:                   21580   BIC:                             3.534e
+04
Df Model:                          16
Covariance Type:            nonrobust
===========================================================================
======
                     coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------------
------
Intercept          0.0020      0.005      0.400      0.689      -0.008
0.012
sqft_living        0.3590      0.006     58.837      0.000       0.347
0.371
median_income      0.0817      0.004     18.610      0.000       0.073
0.090
view_num           0.1385      0.004     32.941      0.000       0.130
0.147
Waterfront         0.1336      0.004     33.105      0.000       0.126
0.142
grade_3         3.993e+09   6.77e+09      0.590      0.555   -9.28e+09    1.
73e+10
grade_4         2.074e+10   3.52e+10      0.590      0.555   -4.82e+10    8.
97e+10
grade_5         6.177e+10   1.05e+11      0.590      0.555   -1.44e+11    2.
67e+11
grade_6         1.716e+11   2.91e+11      0.590      0.555   -3.99e+11    7.
42e+11
grade_7         2.892e+11    4.9e+11      0.590      0.555   -6.72e+11    1.
25e+12
grade_8         2.637e+11   4.47e+11      0.590      0.555   -6.13e+11    1.
14e+12
grade_9         1.915e+11   3.25e+11      0.590      0.555   -4.45e+11    8.
28e+11
grade_10        1.309e+11   2.22e+11      0.590      0.555   -3.04e+11    5.
66e+11
grade_11        7.903e+10   1.34e+11      0.590      0.555   -1.84e+11    3.
42e+11
grade_12         3.76e+10   6.38e+10      0.590      0.555   -8.74e+10    1.
63e+11
grade_13        1.439e+10   2.44e+10      0.590      0.555   -3.35e+10    6.
22e+10
lat                0.2265      0.004     56.075      0.000       0.219
0.234
renovated          0.0655      0.004     17.492      0.000       0.058
0.073
===========================================================================
===
Omnibus:                    14558.020   Durbin-Watson:                       1.
990
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               649892.
683
Skew:                           2.680   Prob(JB):
0.00
Kurtosis:                      29.334   Cond. No.                         3.52e
+14
===========================================================================
```

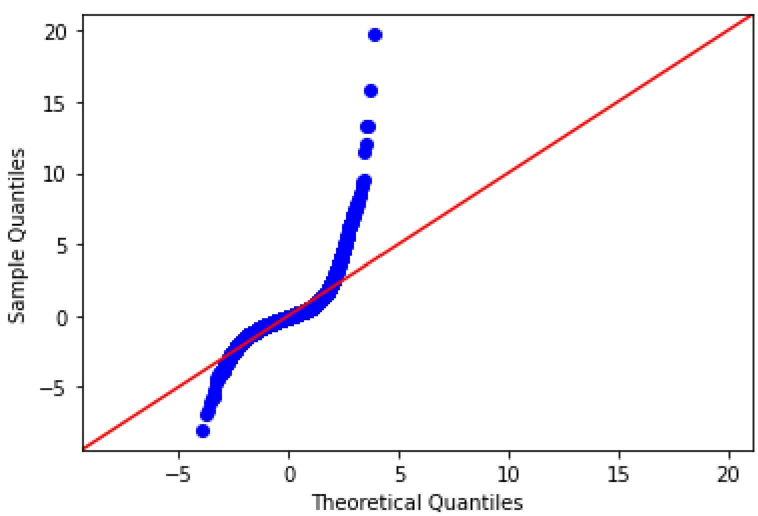
```
===

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corr
ectly specified.
[2] The smallest eigenvalue is 4.18e-25. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

This is the correlation table between variables

               sqft_living  median_income  view_num  Waterfront   grade_3
\
sqft_living       1.000000       0.337315  0.281715    0.104637 -0.011565
median_income     0.337315       1.000000  0.038370    0.002275 -0.004238
view_num          0.281715       0.038370  1.000000    0.380543 -0.002075
Waterfront        0.104637       0.002275  0.380543    1.000000 -0.000561
grade_3          -0.011565      -0.004238 -0.002075   -0.000561  1.000000
grade_4          -0.053935      -0.009219 -0.003934   -0.002919 -0.000241
grade_5          -0.127198      -0.050634 -0.013479    0.012691 -0.000724
grade_6          -0.312486      -0.169862 -0.059287   -0.007301 -0.002197
grade_7          -0.358915      -0.215072 -0.147272   -0.045482 -0.005738
grade_8           0.071115       0.065800  0.010612   -0.011317 -0.004252
grade_9           0.318499       0.185681  0.094153    0.007487 -0.002526
grade_10          0.369228       0.215769  0.127753    0.051514 -0.001602
grade_11          0.345964       0.134906  0.140282    0.068410 -0.000934
grade_12          0.238136       0.068073  0.114607    0.082899 -0.000438
grade_13          0.144424       0.021730  0.051769   -0.002025 -0.000167
lat               0.052155       0.375826  0.006321   -0.012157 -0.017283
renovated         0.050829       0.002503  0.090480    0.074267 -0.001285

                grade_4   grade_5   grade_6   grade_7   grade_8   grade_9
\
sqft_living   -0.053935 -0.127198 -0.312486 -0.358915  0.071115  0.318499
median_income -0.009219 -0.050634 -0.169862 -0.215072  0.065800  0.185681
view_num      -0.003934 -0.013479 -0.059287 -0.147272  0.010612  0.094153
Waterfront    -0.002919  0.012691 -0.007301 -0.045482 -0.011317  0.007487
grade_3       -0.000241 -0.000724 -0.002197 -0.005738 -0.004252 -0.002526
grade_4        1.000000 -0.003766 -0.011421 -0.029831 -0.022108 -0.013132
grade_5       -0.003766  1.000000 -0.034363 -0.089757 -0.066521 -0.039511
grade_6       -0.011421 -0.034363  1.000000 -0.272170 -0.201711 -0.119810
grade_7       -0.029831 -0.089757 -0.272170  1.000000 -0.526882 -0.312951
grade_8       -0.022108 -0.066521 -0.201711 -0.526882  1.000000 -0.231935
grade_9       -0.013132 -0.039511 -0.119810 -0.312951 -0.231935  1.000000
grade_10      -0.008329 -0.025060 -0.075989 -0.198488 -0.147104 -0.087375
grade_11      -0.004854 -0.014605 -0.044286 -0.115678 -0.085732 -0.050922
grade_12      -0.002276 -0.006848 -0.020765 -0.054238 -0.040197 -0.023876
grade_13      -0.000868 -0.002613 -0.007922 -0.020693 -0.015336 -0.009109
lat           -0.016323 -0.046573 -0.062851 -0.040532  0.026330  0.042136
renovated      0.000502 -0.010460  0.002425 -0.017076  0.006251  0.016277

               grade_10  grade_11  grade_12  grade_13       lat  renovated
sqft_living    0.369228  0.345964  0.238136  0.144424  0.052155   0.050829
median_income  0.215769  0.134906  0.068073  0.021730  0.375826   0.002503
view_num       0.127753  0.140282  0.114607  0.051769  0.006321   0.090480
Waterfront     0.051514  0.068410  0.082899 -0.002025 -0.012157   0.074267
grade_3       -0.001602 -0.000934 -0.000438 -0.000167 -0.017283  -0.001285
grade_4       -0.008329 -0.004854 -0.002276 -0.000868 -0.016323   0.000502
```

```
grade_5      -0.025060 -0.014605 -0.006848 -0.002613 -0.046573  -0.010460
grade_6      -0.075989 -0.044286 -0.020765 -0.007922 -0.062851   0.002425
grade_7      -0.198488 -0.115678 -0.054238 -0.020693 -0.040532  -0.017076
grade_8      -0.147104 -0.085732 -0.040197 -0.015336  0.026330   0.006251
grade_9      -0.087375 -0.050922 -0.023876 -0.009109  0.042136   0.016277
grade_10      1.000000 -0.032297 -0.015143 -0.005777  0.052262   0.002202
grade_11     -0.032297  1.000000 -0.008825 -0.003367  0.039372  -0.001405
grade_12     -0.015143 -0.008825  1.000000 -0.001579  0.016946  -0.000261
grade_13     -0.005777 -0.003367 -0.001579  1.000000  0.013142   0.016067
lat           0.052262  0.039372  0.016946  0.013142  1.000000   0.027908

renovated     0.002202 -0.001405 -0.000261  0.016067  0.027908   1.000000
```

```
This is the residual plot and qq plot
```

```
None
Figure(432x288)
```





Create a New Dataframe of the Co-Efficient Grade between grades for analysis

In [132]:

```
grade_df = pd.read_csv('data/grade_values_final_model.csv')
```

In [133]:
```
grade_df
```

Out[133]:

	grade	regression_coefficient	change_from_previous_grade
0	3	-2500000	0
1	4	-2770000	-270000
2	5	-2780000	-10000
3	6	-2770000	10000
4	7	-2760000	10000
5	8	-2720000	40000
6	9	-2620000	100000
7	10	-2460000	160000
8	11	-2210000	250000
9	12	-1730000	480000
10	13	-496000	1234000

In [134]:
```
#Make the columsn numeric values
grade_df.astype('int64')
```

Out[134]:

	grade	regression_coefficient	change_from_previous_grade
0	3	-2500000	0
1	4	-2770000	-270000
2	5	-2780000	-10000
3	6	-2770000	10000
4	7	-2760000	10000
5	8	-2720000	40000
6	9	-2620000	100000
7	10	-2460000	160000
8	11	-2210000	250000
9	12	-1730000	480000
10	13	-496000	1234000

In [135]:

```
grade_df['change_from_3'] = (grade_df['regression_coefficient'] - -250000
grade_df
```

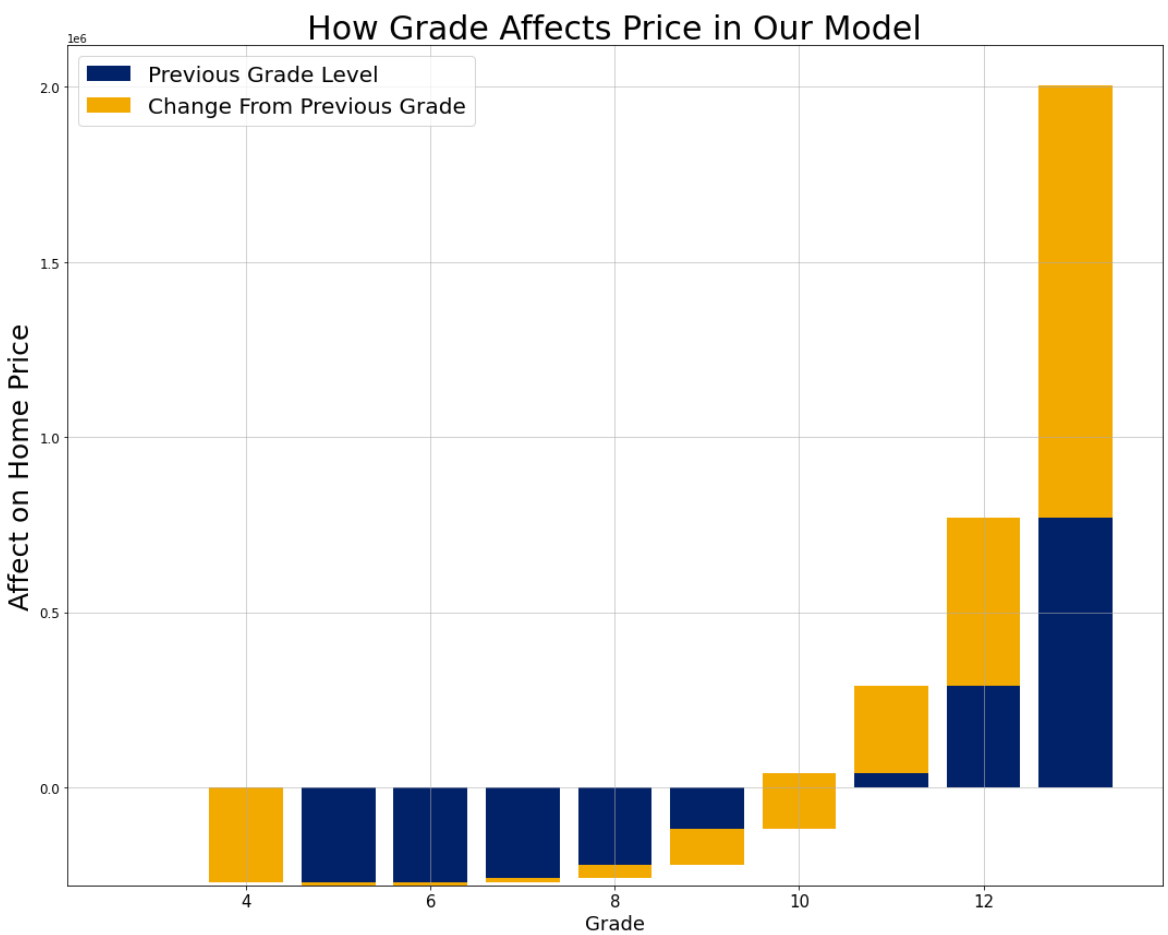
Out[135]:

	grade	regression_coefficient	change_from_previous_grade	change_from_3
0	3	-2500000	0	0
1	4	-2770000	-270000	-270000
2	5	-2780000	-10000	-280000
3	6	-2770000	10000	-270000
4	7	-2760000	10000	-260000
5	8	-2720000	40000	-220000
6	9	-2620000	100000	-120000
7	10	-2460000	160000	40000
8	11	-2210000	250000	290000
9	12	-1730000	480000	770000
10	13	-496000	1234000	2004000

Create a graph for analyses

In [137]:

```
fig, ax = plt.subplots(figsize=(15, 12))

ax.bar(grade_df['grade'], grade_df['change_from_3'], color = '#012169', 
ax.bar(grade_df['grade'], grade_df['change_from_previous_grade'],
       bottom = (grade_df['change_from_3'] -grade_df['change_from_previo
                 ,color = '#F2A900', label = 'Change From Previous Grade

plt.legend(loc="upper left", fontsize = 20)
ax.set_title('How Grade Affects Price in Our Model', fontsize = 30)
ax.set_xlabel('Grade', fontsize = 18)
ax.set_ylabel('Affect on Home Price', fontsize = 25)
ax.tick_params(axis='x', labelsize=15)
ax.tick_params(axis='y', labelsize=12)
ax.grid(which = 'major', alpha = .7)


plt.tight_layout()
```



Our final model residual graph shows it is pretty evenly distributed between over and under estimating price. It has a slight lean towards overestimating price that should be analyzed further.

Additionally, the QQ plot shows that our model fails to accurately predict houses at either end of the extreme. Beyond two standard deviations from our average price our model becomes inaccurate. This is most likely due to high/low end houses having extremely specific reasons for their price. Our model is too general to account for these factors.

Based off the coefficient for our model. We saw that with all other variables being equal a one dollar increase in median income affects the final price by $.975. Additionally, we were able to see how grade affects a house from one grade to the next.

Recommendations

Our recommendations are thusly to not focus on improving the grade of a house at the lower end. But heavily prioritize improving the grade of houses beyond 10. We saw an exponential relationship between grade and price meaning it is far more valuable to move up from 10-11 than it is to move from 5-6. This recommendation can also be used for building houses. It makes economic sense to reach a minimum grade of around 5 for new houses being built, but is likely not worth spending money to prioritize a higher grade for these types of houses. However, when building luxury homes achieving the highest grade possible will raise the price significantly.

Additionally, using the coefficient values in our model for locations can give insight into finding undervalued homes in desirable locations.

Farther Analysis

Using the data provided, a more complicated model could be made to improve its accuracy. This may lead to significant multi-collinearity complications and reduce the interpretability of any one variable.

Additionally, more data could be gathered to improve our location metrics. Although median income served as a reasonable proxy for a neighborhood's desirability it's not a perfect predictor. Researching things such as school districts, walkability, or crime would improve the model.

In [ ]:

```
 
```