Final Notebook

Please fill out:

- Student name: Andrew Levinton
- Student pace: self paced
- Scheduled project review date/time:
- Instructor name: Ahbineet Kulkarni
- Blog post URL:

Statsmodels debug

- This is because statsmodels was having version issues. this is a workaround
- The code below re-publishes the existing (but private) _centered function as a public attribute to the module already imported in RAM.

In [887…

```
import  scipy.signal.signaltools

def _centered(arr, newsize):
    # Return the center newsize portion of the array.
    newsize = np.asarray(newsize)
    currsize = np.array(arr.shape)
    startind = (currsize - newsize) // 2
    endind = startind + newsize
    myslice = [slice(startind[k], endind[k]) for k in range(len(endind))]
    return arr[tuple(myslice)]

scipy.signal.signaltools._centered = _centered
```

Import necessary libraries

In [888…

```
# raw data handling
import pandas as pd
import numpy as np
import datetime as dt

# data visualiztion
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns

# regression modeling
import scipy.stats as stats
import statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_log_error


import warnings # weird sns.distplot() warnings
warnings.filterwarnings("ignore")



plt.style.use('ggplot')
```

Define Functions

In [889…

```python
# Grabbing vifs
def get_vifs(data):
    # Get a list of the column names
    cols = data.columns

    # Create an empty DataFrame to hold the VIF results
    vif_data = pd.DataFrame()

    # Loop through each column and calculate the VIF
    for i in range(len(cols)):
        vif = variance_inflation_factor(data[cols].values, i)
        vif_data = vif_data.append({'Variable': cols[i], 'VIF': vif}, ignore_index=True)

    # Print the VIF results
    return print(vif_data)


# get ols model and plot residual distribution
def get_OLS_model(name, X, y):
    model = sm.OLS(y, sm.add_constant(X))
    results = model.fit()
    model_residual = results.resid

    return print(results.summary()), plt.suptitle(f'Residual distribution for {name} model'), sns.distplot(model_resid



#get qq and histogram plots
def plot_hist_qq(df, target_col):
    """
    Creates a histogram and QQ-plot for a given dataframe and target column.

    Args:
        df (pandas.DataFrame): The dataframe to plot.
        target_col (str): The name of the target column.

    Returns:
        None
    """
    # Create subplots with 1 row and 2 columns
    fig, axs = plt.subplots(1, 2, figsize=(10, 5))

    # Plot histogram on the first subplot
    axs[0].hist(df[target_col], bins=30)
    axs[0].set_xlabel(target_col)
    axs[0].set_ylabel('Frequency')

    # Plot QQ-plot on the second subplot
    stats.probplot(df[target_col], plot=axs[1])
    axs[1].set_xlabel('Theoretical quantiles')
    axs[1].set_ylabel('Sample quantiles')

    # Adjust the layout and display the plot
    plt.tight_layout()
    plt.show()

# getting qqplots from stats model
def get_model_qqplots(data, y):
    # Set up the plot grid
    fig, axes = plt.subplots(nrows=5, ncols=6, figsize=(25, 18))

    # Loop through each variable in the DataFrame
    for i, var in enumerate(data.columns):
        # Fit a linear regression model
        X = sm.add_constant(data[var])
        model = sm.OLS(y, X).fit()

        # Calculate the residuals
        resid = model.resid

        # Create a QQ plot
        sm.qqplot(resid, line='s', ax=axes[i//6, i%6])
```

```
        axes[i//6, i%6].set_title(var)

    plt.tight_layout()
    plt.show()

def get_log_mse(X,y):
    model = LinearRegression()
    model.fit(X, y)

    # Calculate the predicted values of the target variable using the linear model
    y_pred = model.predict(X)

    return mean_squared_log_error(y, y_pred)

def get_error_metrics(X,y):
    # Fit a linear regression model with an intercept term
    model = sm.OLS(y, sm.add_constant(X))
    results = model.fit()

    # Get the predicted values for the input data
    y_pred = results.predict(sm.add_constant(X))

    # Calculate MSE
    mse = np.mean((y - y_pred)**2)

    # Calculate RMSE
    rmse = np.sqrt(mse)

    # Calculate MAPE
    mape = np.mean(np.abs((y - y_pred) / y)) * 100

    print("Mean Squared Error: ", mse)
    print("Root Mean Squared Error: ", rmse)
    print("Mean Average Percentage Error: ", mape,'%')
```

Read in dataset, check length

In [890…

```
cd data
```

```
C:\Users\alevi\Documents\Flatiron\dsc-data-science-env-config\Course_Folder\Phase_2\Housing_Linear_Model_Project\data
```

In [891…

```
df = pd.read_csv('kc_house_data.csv')
len(df)
```

Out[891…

```
30155
```

Dataset timeline

In [892…

```
df['yr_built'].min(), df['yr_built'].max()
```

Out[892…

```
(1900, 2022)
```

Checking dtypes

In [893…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30155 entries, 0 to 30154
Data columns (total 25 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             30155 non-null  int64
 1   date           30155 non-null  object
 2   price          30155 non-null  float64
 3   bedrooms       30155 non-null  int64
 4   bathrooms      30155 non-null  float64
```

```
 5   sqft_living    30155 non-null  int64
 6   sqft_lot       30155 non-null  int64
 7   floors         30155 non-null  float64
 8   waterfront     30155 non-null  object
 9   greenbelt      30155 non-null  object
 10  nuisance       30155 non-null  object
 11  view           30155 non-null  object
 12  condition      30155 non-null  object
 13  grade          30155 non-null  object
 14  heat_source    30123 non-null  object
 15  sewer_system   30141 non-null  object
 16  sqft_above     30155 non-null  int64
 17  sqft_basement  30155 non-null  int64
 18  sqft_garage    30155 non-null  int64
 19  sqft_patio     30155 non-null  int64
 20  yr_built       30155 non-null  int64
 21  yr_renovated   30155 non-null  int64
 22  address        30155 non-null  object
 23  lat            30155 non-null  float64
 24  long           30155 non-null  float64
dtypes: float64(5), int64(10), object(10)
memory usage: 5.8+ MB
```

Linear Model must meet the following assumptions:

Simple Linear Regression on select features

Assumption check:

- Is it linear?
  - Scatter plots
- Is it normal?
  - histogram
  - QQ-plot
- Is it homoscedastic?
  - Durbin-Watson Score
- Does the model present with multicollinearity?

The process for building this linear model:

- Prep data for linear model regression: This involves dropping null values, dropping "bad data", as well as engineering features to assist in assuming linearization
- Key scores to look at:
- `R-Squared ( or the coefficient of determination)` - a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit).
- `Correlation coefficients` - check to see what variables seem relatable to the target variable (price)
- `residual plots` - check how far data compares to the mean. Data should be normally distributed to avoid skewness of the mean
- `variance inflation factor` - level of statistical skew
- `Root mean squared erro` r - how far predictions fall from measured true values using Euclidean distance.
- `pvalues of independent variables` - measures how statistically significant the independent variables are

Data Preparation

Dropping nullls

In [894…

```
df.dropna(inplace=True)
```

Recheck length

In [895…
```
len(df)
```
Out[895…
```
30111
```

Looking at Washington state

In [896…
```
df['address'] = df['address'].str.lower()
```
In [897…
```
df = df[df['address'].str.contains('washington')]
```
In [898…
```
len(df)
```
Out[898…
```
29208
```

Grabbing Zipcodes

In [899…
```
df['zipcode'] = df['address'].apply(lambda x: x.split(',')[2].split(' ')[-1])
```
In [900…
```
df['zipcode'] = df['zipcode'].astype(str)
```
In [901…
```
df['zipcode'].unique()
```
Out[901…
```
array(['98055', '98133', '98178', '98118', '98027', '98166', '98030',
       '98023', '98019', '98144', '98031', '98092', '98103', '98006',
       '98136', '98007', '98038', '98057', '98077', '98126', '98053',
       '98039', '98107', '98008', '98155', '98168', '98199', '98004',
       '98045', '98052', '98011', '98002', '98033', '98116', '98198',
       '98125', '98001', '98112', '98034', '98056', '98059', '98005',
       '98040', '98014', '98106', '98029', '98122', '98003', '98117',
       '98042', '98119', '98065', '98022', '98072', '98058', '98108',
       '98115', '98074', '98105', '98024', '98146', '98109', '98102',
       '98028', '98188', '98177', '98075', '98010', '98148', '98047',
       '98032', '98070', '98051', '98288', '98354', '98272', '98296',
       '98271', '98050', '63090', 'seattle', '98387', '15301', '98251',
       '98223', '98338', '98224', '98372', '98663', '99202', '99403',
       '98422', '99203', '99223', '98270'], dtype=object)
```

Categorizing waterfronts

In [902…
```
duwamish = ['98168']
elliot_bay_zips= ['98119','98104','98129','98132','98127','98125','98195','98101','98134','98170','98139','98131','981
puget_sound = ['98071','98083','98013','98070','98031','98131','98063','98195','98207','98190']
lake_union = ['98109']
ship_canal = ['00000']
lake_washington = ['98072','98077']
lake_sammamish = ['98074','98075','98029']
other = ['00000']
river_slough_waterfronts = ['00000']

df['waterfront_loc'] = df['zipcode'].apply(lambda x: 'Duwamish' if x=='98168'\
                                          else 'Elliot Bay' if x in elliot_bay_zips\
                                          else 'Puget Sound' if x in puget_sound\
                                          else 'Lake Union' if x in lake_union\
                                          else 'ship canal' if x in ship_canal\
                                          else 'Lake Washington' if x in lake_washington\
                                          else 'Lake Sammamish' if x in lake_sammamish\
                                          else 'other')
```

In [903...

```
df['waterfront_loc'].value_counts()
```

Out[903...

```
other              25497
Lake Sammamish      1159
Elliot Bay           730
Puget Sound          721
Lake Washington      589
Duwamish             383
Lake Union           129
Name: waterfront_loc, dtype: int64
```

Filter by state of Washington Zipcodes (assuming seattle is its own zipcode)

In [904...

```
df = df[df['zipcode'].str.startswith('98') | df['zipcode'].str.contains('seattle')]
```

One Hot Encoding Waterfronts

In [905...

```
waterfront_dummies = pd.get_dummies(df['waterfront_loc'], prefix='water', drop_first=True)
```

In [906...

```
waterfront_dummies
```

Out[906...

	water_Elliot Bay	water_Lake Sammamish	water_Lake Union	water_Lake Washington	water_Puget Sound	water_other
0	0	0	0	0	0	1
1	0	0	0	0	0	1
2	0	0	0	0	0	1
3	0	0	0	0	0	1
4	0	0	0	0	0	1
...	...	...	...	...	...	...
30150	0	0	0	0	0	1
30151	0	0	0	0	0	1
30152	0	0	0	0	0	1
30153	0	0	0	0	0	1
30154	0	0	0	0	0	1

29200 rows × 6 columns

In [907...

```
len(df)
```

Out[907...

```
29200
```

In [908...

```
len(df) == len(waterfront_dummies)
```

Out[908...

```
True
```

In [909...

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

replacing seattle with seattle zipcode

In [910...

```
df['zipcode'] = df['zipcode'].apply(lambda x: '98101' if x=='seattle' else x)
```

recheck zipcodes

In [911…

```
df['zipcode'].unique()
```

Out[911…

```
array(['98055', '98133', '98178', '98118', '98027', '98166', '98030',
       '98023', '98019', '98144', '98031', '98092', '98103', '98006',
       '98136', '98007', '98038', '98057', '98077', '98126', '98053',
       '98039', '98107', '98008', '98155', '98168', '98199', '98004',
       '98045', '98052', '98011', '98002', '98033', '98116', '98198',
       '98125', '98001', '98112', '98034', '98056', '98059', '98005',
       '98040', '98014', '98106', '98029', '98122', '98003', '98117',
       '98042', '98119', '98065', '98022', '98072', '98058', '98108',
       '98115', '98074', '98105', '98024', '98146', '98109', '98102',
       '98028', '98188', '98177', '98075', '98010', '98148', '98047',
       '98032', '98070', '98051', '98288', '98354', '98272', '98296',
       '98271', '98050', '98101', '98387', '98251', '98223', '98338',
       '98224', '98372', '98663', '98422', '98270'], dtype=object)
```

In [912…

```
len(df['zipcode'].unique())
```

Out[912…

```
89
```

Adding in Engineered Zipcode Data Generated from GreatSchools API

The csv file that is being imported was generated using an extensive process of requests and data aggregation of school ratings by zipcode. To view the process of retrieval and aggregation please visit the file Final_Exploratory_Data_Analysis.ipynb in the notebooks folder.

In [913…

```
cd ..
```

```
C:\Users\alevi\Documents\Flatiron\dsc-data-science-env-config\Course_Folder\Phase_2\Housing_Linear_Model_Project
```

In [914…

```
school_ratings = pd.read_csv('school_ratings_by_zipcode_.csv')
school_ratings['zipcode'] = school_ratings['zipcode'].astype(str)
```

In [915…

```
school_ratings
```

Out[915…

	zipcode	rating
0	98001	4.000000
1	98002	3.888889
2	98004	7.666667
3	98005	7.333333
4	98006	8.666667
...	...	...
68	98664	3.500000
69	98682	3.333333
70	98683	5.600000
71	98684	3.555556
72	98686	5.000000

73 rows × 2 columns

Assigning average school ratings to corresponding zipcodes

In [916…

```
# Create a dictionary from the zipcode dataframe
zip_dict = school_ratings.set_index('zipcode')['rating'].to_dict()

# Load your larger dataframe


# Assign the ratings from the zipcode dictionary to the large dataframe
df['school_rating'] = df['zipcode'].apply(lambda x: zip_dict.get(x, None))

# The above line applies the lambda function to each element of the 'zipcode' column of the large dataframe.
# If the zipcode is present in the zip_dict, its corresponding rating is assigned to the 'rating' column.
# If not, None is assigned.

# You can then save the updated large dataframe to a new csv file
df['school_rating'].isnull().sum()
```

Out[916…

```
8932
```

Filling nulls with mean value

In [917…

```
mean_val = df['school_rating'].mean()
med_val = df['school_rating'].median()
```

In [918…

```
mean_val, med_val
```

Out[918…

```
(5.937337959170456, 6.0)
```

In [919…

```
df['school_rating'] = df['school_rating'].fillna(mean_val)
```

In [920…

```
df['school_rating'].isnull().sum()
```

Out[920…

```
0
```

Observing correlation matrix for possible features that can be used with the price

In [921…

```
df.corr()['price'].abs().sort_values(ascending=False)
```

Out[921…

```
price                    1.000000
sqft_living              0.616741
sqft_above               0.546108
bathrooms                0.488039
school_rating            0.364442
sqft_patio               0.317623
lat                      0.296212
bedrooms                 0.290994
sqft_garage              0.267477
sqft_basement            0.246548
floors                   0.199285
water_Lake Sammamish     0.141426
yr_built                 0.105877
sqft_lot                 0.086790
yr_renovated             0.085506
long                     0.081940
water_Lake Washington    0.070383
water_Puget Sound        0.068457
water_other              0.064781
water_Lake Union         0.035352
id                       0.030237
water_Elliot Bay         0.004859
Name: price, dtype: float64
```

In [922…

```python
# Increase the size of the heatmap.
plt.figure(figsize=(16, 6))
# Store heatmap object in a variable to easily access it when you want to include more features (such as title).
# Set the range of values to be displayed on the colormap from -1 to 1, and set the annotation to True to display the
heatmap = sns.heatmap(df.corr(), vmin=-1, vmax=1, annot=True)
# Give a title to the heatmap. Pad defines the distance of the title from the top of the heatmap.
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad=12);
```

Observations

- At first glance, it appears that sqft_living, sqft_above and bathrooms are the strongest correlated features to the price.
- Further investigation is needed to measure the validity of the variables. They may be correlated with the price due to skewness or other factors that can make the correlation a deceptively "good" feature.
- To investigate further, we will monitor the Variance Inflation Factor(VIF) to address the issue of multicollinearity.

Changing categorical variables to numerical columns - this needs to be done if we want to use them in a linear model

In [923…

```python
#extracting grade as an integer
df['grade'] = df['grade'].apply(lambda x: int(str(x.split(' ')[0])))

# replacing conditions with values
cond_dict = {'Poor':1,'Fair':2,'Average':3,'Good':4,'Very Good':5}
df.condition.replace(to_replace=cond_dict,inplace=True)

#changing date to datetime object, get day and month
df['date'] = pd.to_datetime(df['date'])
df['month'] = df['date'].dt.month

df['day_of_year'] = df['date'].dt.dayofyear
```

Recheck dtypes

In [924…

```python
df.dtypes
```

Out[924…

```
id                                int64
date                     datetime64[ns]
```

```
price                        float64
bedrooms                       int64
bathrooms                    float64
sqft_living                    int64
sqft_lot                       int64
floors                       float64
waterfront                    object
greenbelt                     object
nuisance                      object
view                          object
condition                      int64
grade                          int64
heat_source                   object
sewer_system                  object
sqft_above                     int64
sqft_basement                  int64
sqft_garage                    int64
sqft_patio                     int64
yr_built                       int64
yr_renovated                   int64
address                       object
lat                          float64
long                         float64
zipcode                       object
waterfront_loc                object
water_Elliot Bay               uint8
water_Lake Sammamish           uint8
water_Lake Union               uint8
water_Lake Washington          uint8
water_Puget Sound              uint8
water_other                    uint8
school_rating                float64
month                          int64
day_of_year                    int64
dtype: object
```

Extracting Numerical Predictors by filtering dtypes

In [925…

```
df.dtypes.unique()
```

Out[925…

```
array([dtype('int64'), dtype('<M8[ns]'), dtype('float64'), dtype('O'),
       dtype('uint8')], dtype=object)
```

In [926…

```
# categorizing dtypes
numerical_types = ['int64','float64']
numerical_predictors = list(df.select_dtypes(include=numerical_types))
numerical_predictors
```

Out[926…

```
['id',
 'price',
 'bedrooms',
 'bathrooms',
 'sqft_living',
 'sqft_lot',
 'floors',
 'condition',
 'grade',
 'sqft_above',
 'sqft_basement',
 'sqft_garage',
 'sqft_patio',
 'yr_built',
 'yr_renovated',
 'lat',
 'long',
 'school_rating',
 'month',
 'day_of_year']
```

Create dataframe of numerical values

In [927…

```python
# df[numerical_predictors] selects only numerical columns
df_numerical = df[numerical_predictors]
```

In [928…

```python
df_numerical.columns
```

Out[928…

```
Index(['id', 'price', 'bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot',
       'floors', 'condition', 'grade', 'sqft_above', 'sqft_basement',
       'sqft_garage', 'sqft_patio', 'yr_built', 'yr_renovated', 'lat', 'long',
       'school_rating', 'month', 'day_of_year'],
      dtype='object')
```

In [929…

```python
len(df_numerical)
```

Out[929…

```
29200
```

In [930…

```python
len(waterfront_dummies)
```

Out[930…

```
29200
```

Dropping price to isolate predictors

In [931…

```python
df_numerical = df_numerical.drop(['id','price'],axis=1)
```

In [932…

```python
df_numerical['floors'] = df['floors'].astype(float)
```

Calculating variance inflation factor [VIF]

VIF levels:

- Good: VIF <= 5
- Moderate/Questionable: VIF >=5 and VIF <= 10
- Throw out: VIF >= 10

In [933…

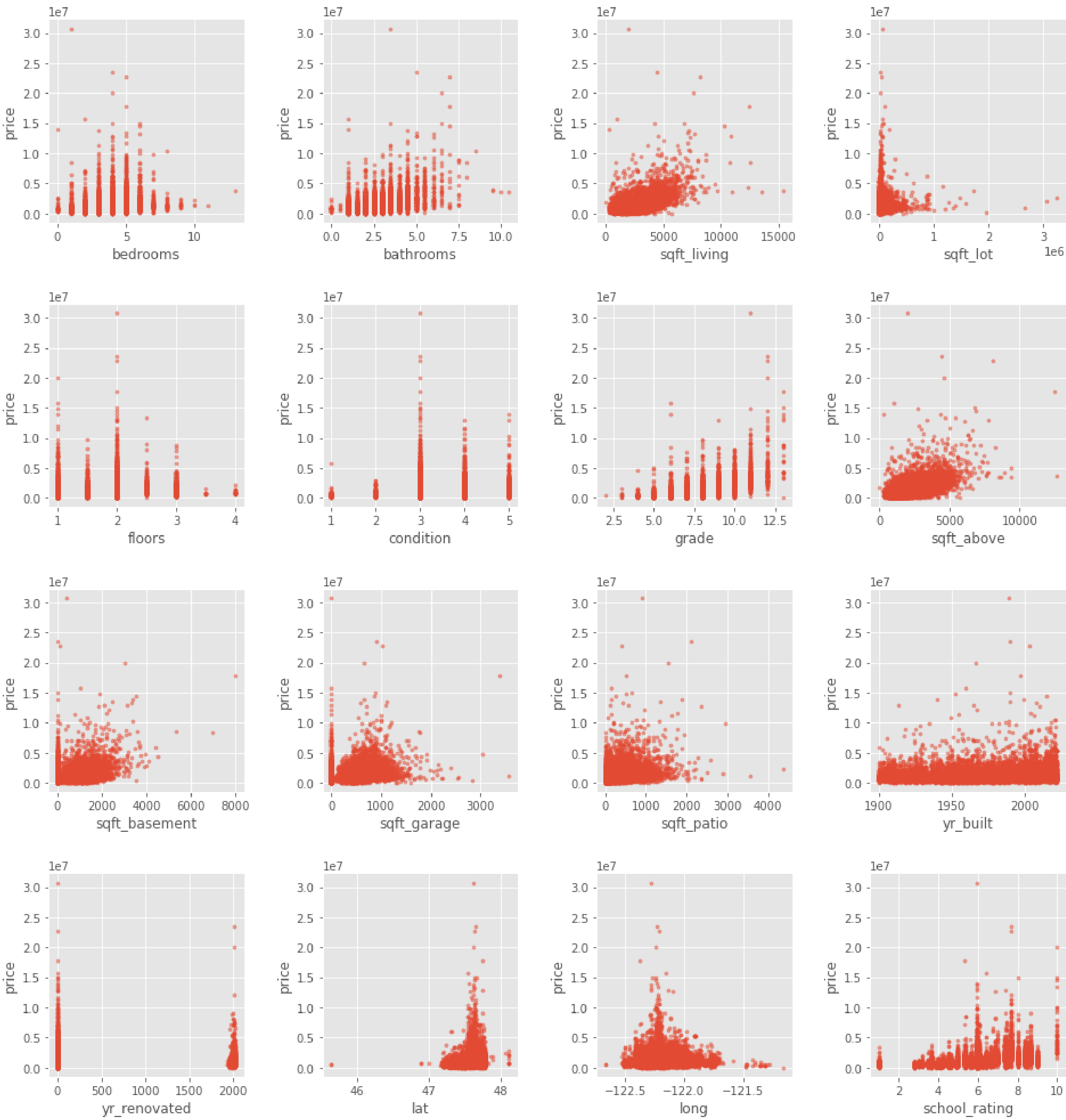
```python
print(get_vifs(df_numerical))
```

```
         Variable            VIF
0        bedrooms      24.784656
1       bathrooms      26.274063
2     sqft_living     119.813949
3        sqft_lot       1.140732
4          floors      17.198465
5       condition      31.166852
6           grade     133.745651
7      sqft_above      92.881318
8   sqft_basement       7.075247
9     sqft_garage       4.672476
10     sqft_patio       2.240448
11       yr_built    9230.201222
12   yr_renovated       1.210978
13            lat  154765.470827
14           long  165297.345977
15  school_rating      26.080014
16          month     697.154476
17    day_of_year     612.158491
None
```

It appears at first glance that the data only yields a small set of independent variables that are not highly collinear with eachother. This will be looked at again after the removal of outliers, and the transformation of data.

In [934…

```
# Specify the dependent variable and independent variables
y_col = 'price'
x_cols = [col for col in df_numerical.columns if col != y_col][:16]  # Use the first 16 independent variables

# Create scatter plot matrix
fig, axs = plt.subplots(4, 4, figsize=(16, 16))
for i, x_var in enumerate(x_cols):
    row, col = divmod(i, 4)
    axs[row, col].scatter(df_numerical[x_var], df[y_col], alpha=0.5, s=10)
    axs[row, col].set_xlabel(x_var)
    axs[row, col].set_ylabel(y_col)

# Adjust plot layout
fig.subplots_adjust(top=0.93, hspace=0.4, wspace=0.4)

# Show the plot
plt.show()
```



Extracting Categorical String Predictors

In [935…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 29200 entries, 0 to 30154
Data columns (total 36 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   id                   29200 non-null  int64
 1   date                 29200 non-null  datetime64[ns]
 2   price                29200 non-null  float64
 3   bedrooms             29200 non-null  int64
 4   bathrooms            29200 non-null  float64
 5   sqft_living          29200 non-null  int64
 6   sqft_lot             29200 non-null  int64
```

```
 7   floors                29200 non-null  float64
 8   waterfront            29200 non-null  object
 9   greenbelt             29200 non-null  object
 10  nuisance              29200 non-null  object
 11  view                  29200 non-null  object
 12  condition             29200 non-null  int64
 13  grade                 29200 non-null  int64
 14  heat_source           29200 non-null  object
 15  sewer_system          29200 non-null  object
 16  sqft_above            29200 non-null  int64
 17  sqft_basement         29200 non-null  int64
 18  sqft_garage           29200 non-null  int64
 19  sqft_patio            29200 non-null  int64
 20  yr_built              29200 non-null  int64
 21  yr_renovated          29200 non-null  int64
 22  address               29200 non-null  object
 23  lat                   29200 non-null  float64
 24  long                  29200 non-null  float64
 25  zipcode               29200 non-null  object
 26  waterfront_loc        29200 non-null  object
 27  water_Elliot Bay      29200 non-null  uint8
 28  water_Lake Sammamish  29200 non-null  uint8
 29  water_Lake Union      29200 non-null  uint8
 30  water_Lake Washington 29200 non-null  uint8
 31  water_Puget Sound     29200 non-null  uint8
 32  water_other           29200 non-null  uint8
 33  school_rating         29200 non-null  float64
 34  month                 29200 non-null  int64
 35  day_of_year           29200 non-null  int64
dtypes: datetime64[ns](1), float64(6), int64(14), object(9), uint8(6)
memory usage: 7.1+ MB
```

In [936…

```
categorical_types = ['O']
categorical_predictors = list(df.select_dtypes(include=categorical_types))
categorical_predictors
```

Out[936…

```
['waterfront',
 'greenbelt',
 'nuisance',
 'view',
 'heat_source',
 'sewer_system',
 'address',
 'zipcode',
 'waterfront_loc']
```

In [937…

```
df_categorical = df[categorical_predictors]
```

In [938…

```
df_categorical
```

Out[938…

	waterfront	greenbelt	nuisance	view	heat_source	sewer_system	address	zipcode	waterfront_loc
0	NO	NO	NO	NONE	Gas	PUBLIC	2102 southeast 21st court, renton, washington ...	98055	other
1	NO	NO	YES	AVERAGE	Oil	PUBLIC	11231 greenwood avenue north, seattle, washing...	98133	other
2	NO	NO	NO	AVERAGE	Gas	PUBLIC	8504 south 113th street, seattle, washington 9...	98178	other
3	NO	NO	NO	AVERAGE	Gas	PUBLIC	4079 letitia avenue south, seattle, washington...	98118	other
4	NO	NO	YES	NONE	Electricity	PUBLIC	2193 northwest talus drive, issaquah, washingt...	98027	other
...	...	...	...	...	...	...	...	...	...
30150	NO	NO	NO	NONE	Oil	PUBLIC	4673 eastern avenue north, seattle, washington...	98103	other

	waterfront	greenbelt	nuisance	view	heat_source	sewer_system	address	zipcode	waterfront_loc
30151	NO	NO	NO	FAIR	Gas	PUBLIC	4131 44th avenue southwest, seattle, washingto...	98116	other
30152	NO	NO	YES	NONE	Gas	PUBLIC	910 martin luther king jr way, seattle, washin...	98122	other
30153	NO	NO	NO	NONE	Gas	PUBLIC	17127 114th avenue southeast, renton, washingt...	98055	other
30154	NO	NO	NO	NONE	Oil	PUBLIC	18615 7th avenue south, burien, washington 981...	98148	other

29200 rows × 9 columns

Model #1

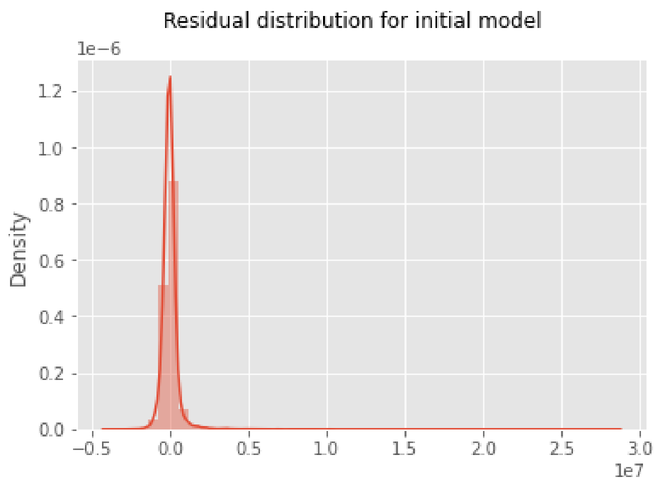
In [939…

```
model_data = df_numerical
```

In [940…

```
get_OLS_model('initial',X = model_data, y = df['price'])
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.528
Model:                            OLS   Adj. R-squared:                  0.528
Method:                 Least Squares   F-statistic:                     1814.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:25   Log-Likelihood:            -4.3066e+05
No. Observations:               29200   AIC:                         8.614e+05
Df Residuals:                   29181   BIC:                         8.615e+05
Df Model:                          18
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const         -7.813e+07   3.98e+06    -19.626      0.000   -8.59e+07   -7.03e+07
bedrooms      -1.074e+05   5020.496    -21.393      0.000   -1.17e+05   -9.76e+04
bathrooms      8.633e+04   7420.424     11.634      0.000    7.18e+04    1.01e+05
sqft_living     213.7108     16.821     12.705      0.000     180.742     246.680
sqft_lot          0.2844      0.062      4.616      0.000       0.164       0.405
floors        -1.673e+05   9450.694    -17.706      0.000   -1.86e+05   -1.49e+05
condition      5.205e+04   5692.993      9.144      0.000    4.09e+04    6.32e+04
grade           1.92e+05   5493.865     34.944      0.000    1.81e+05    2.03e+05
sqft_above      270.8766     17.168     15.778      0.000     237.227     304.526
sqft_basement    75.8317     12.704      5.969      0.000      50.930     100.733
sqft_garage    -153.9644     17.798     -8.650      0.000    -188.850    -119.079
sqft_patio      193.8916     16.438     11.795      0.000     161.672     226.111
yr_built      -2451.7504    188.002    -13.041      0.000   -2820.244   -2083.257
yr_renovated     73.8796      9.195      8.035      0.000      55.857      91.902
lat            8.987e+05   3.04e+04     29.595      0.000    8.39e+05    9.58e+05
long          -3.167e+05   3.16e+04    -10.029      0.000   -3.79e+05   -2.55e+05
school_rating  9.008e+04   3034.290     29.687      0.000    8.41e+04     9.6e+04
month          1.694e+04   1.26e+04      1.344      0.179   -7771.459    4.17e+04
day_of_year   -1129.9476    413.820     -2.731      0.006   -1941.053    -318.842
==============================================================================
Omnibus:                    47675.574   Durbin-Watson:                   1.910
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        103237212.459
Skew:                          10.409   Prob(JB):                         0.00
Kurtosis:                     293.550   Cond. No.                     6.98e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.98e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[940…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for initial model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Getting RMSE(Root Mean Squared Error), MAE(Mean Absolute Error), and MAPE(Mean Absolute Percentage Error)

In [941…

```
from sklearn.metrics import mean_squared_error

def get_rmse(X, y):
    model = sm.OLS(y, sm.add_constant(X))
    result = model.fit()

    # Calculate the predicted values of the target variable using the linear model
    y_pred = result.predict(sm.add_constant(X))

    # Calculate the RMSE
    rmse = np.sqrt(mean_squared_error(y, y_pred))

    return rmse
get_rmse(model_data,df['price'])
```

Out[941…

```
615147.2440023683
```

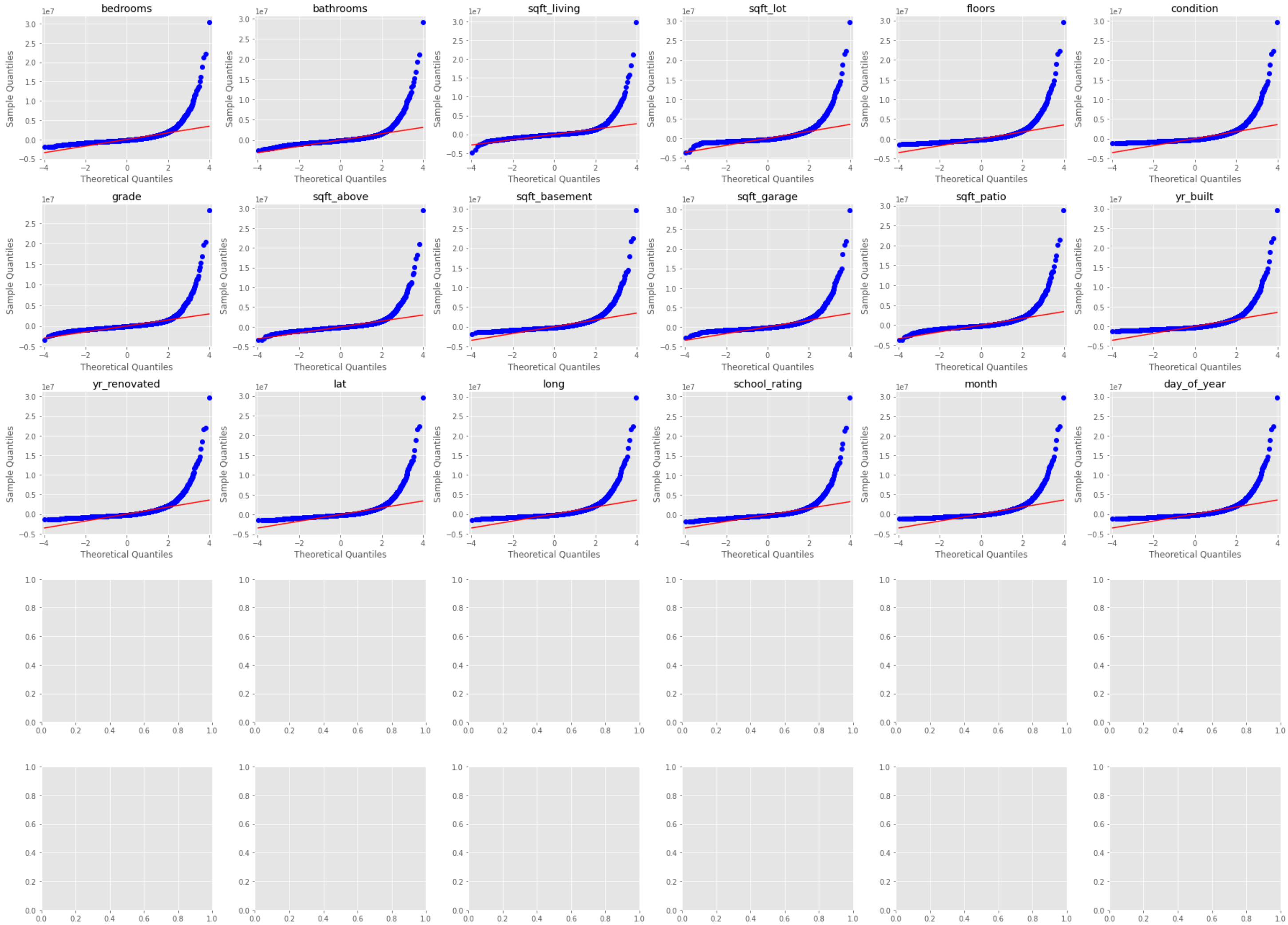
In [942…

```
get_error_metrics(model_data,df['price'])
```

```
Mean Squared Error:  378406131803.7115
Root Mean Squared Error:  615147.2440023702
Mean Average Percentage Error:  38.34330476891375 %
```

In [943…

```
get_model_qqplots(model_data, df['price'])
```



Observations

p_value > 0.05

- `longitude **`
- `month`
    - month was not anticipated as an effective predictor because it is not typical for the season to affect the sale price of a house

  Additional Observations:

    - The adjusted r-squared value is .514, indicating that his model can explain approximately 51.4% of the data.
    - Skew: A kurtosis value between -2 and +2 is good to prove normalcy. The skew score is 10.065, indicating that this model is heavily skewed. This will be addressed through transformations to normalize the data.

Possible Improvements to be made to model:

```
- dropping of variables that are not statistically significant (Pval > 0.05)
- addition of categorial variables(one hot encoded)
- location would possibly be the most interesting variable, mapped against the waterfront or view
variable
- transformation of data to satisfy normality assumption -ex: log transformation or square root
transformation
- removal of outliers: Outliers in this case will be considered to be any data falling greater than
  3 standard deviations outside the mean
```

Goals

```
- improve skewness - removal of outliers, transform data
- improve on homoscedasticity
- increase rsquared to promote higher level explanation of data from model
- remove collinearity - all VIFs less than a value of 5.
```

Categorical data Exploratory Analysis and Engineering

The goal of this section will be to add in meaningful categorical data to the model, to be OneHotEncoded once prepped. For this, we first look at the categorical data.

In [944…

```
df_categorical.columns
```

Out[944…

```
Index(['waterfront', 'greenbelt', 'nuisance', 'view', 'heat_source',
       'sewer_system', 'address', 'zipcode', 'waterfront_loc'],
      dtype='object')
```

Possible categorical variables of interest:

- `waterfront` - Whether the house is on a waterfront
    - Includes Duwamish, Elliott Bay, Puget Sound, Lake Union, Ship Canal, Lake Washington, Lake Sammamish, other lake, and river/slough waterfronts
- `greenbelt` - Whether the house is adjacent to a green belt
- `nuisance` - Whether the house has traffic noise or other recorded nuisances
- `view` - Quality of view from house
    - Includes views of Mt. Rainier, Olympics, Cascades, Territorial, Seattle Skyline, Puget Sound, Lake Washington, Lake Sammamish, small lake / river / creek, and other
- `heat_source` - Heat source for the house
- `sewer_system` - Sewer system for the house
- `address` - The street address

The grade and condition are already onehotencoded in the model and could be changed to a numerical variable, so this part of the analysis will focus on the string categorical variables.

The address appears to be the most interesting variable in the batch because it can be mapped against the waterfronts or the quality of view from the houses. For this, we will extrapolate features of the address to reduce and categorize the location.

In [945…

```
df['waterfront'].unique()
```

Out[945…

```
array(['NO', 'YES'], dtype=object)
```

Data like this will be converted to a numeric boolean, Yes as 1 and No as 0.

In [946…

```
# convert waterfront into numeric boolean
waterfront_bool_dict = {'YES':1,'NO':0,np.nan:0}
df_categorical.waterfront.replace(to_replace=waterfront_bool_dict,inplace=True)
```

In [947…

```
plt.scatter(x=df['waterfront'], y=df['price'])
```

Out[947…

```
<matplotlib.collections.PathCollection at 0x1eda8b50d90>
```

In [948...

```
df['nuisance'].unique()
```

Out[948...

```
array(['NO', 'YES'], dtype=object)
```

In [949...

```
# convert nuisance into numeric boolean
nuisance_bool_dict = {'YES':1,'NO':0,np.nan:0}
df_categorical.nuisance.replace(to_replace=nuisance_bool_dict,inplace=True)
```

In [950...

```
plt.scatter(x=df['nuisance'], y=df['price'])
```

Out[950...

```
<matplotlib.collections.PathCollection at 0x1eda97bcdf0>
```

In [951...

```
# convert nuisance into numeric boolean
greenbelt_bool_dict = {'YES':1,'NO':0,np.nan:0}
df_categorical.greenbelt.replace(to_replace=greenbelt_bool_dict,inplace=True)
```

In [952...

```
df['view'].unique()
```

Out[952...

```
array(['NONE', 'AVERAGE', 'EXCELLENT', 'FAIR', 'GOOD'], dtype=object)
```

In [953...

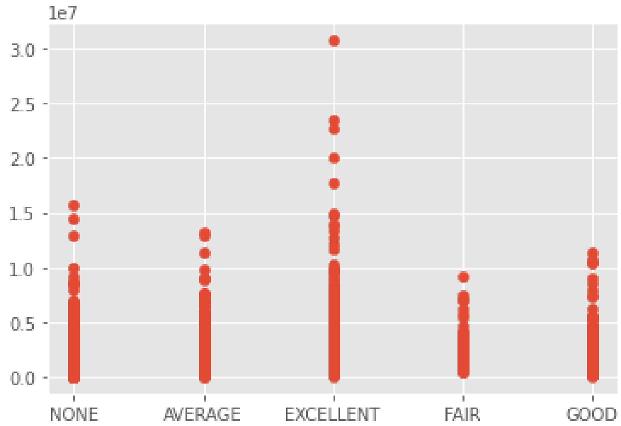
```
# convert view from string into categorical ordinal
view_dict = {'NONE':0,'FAIR':1,'AVERAGE':2,'GOOD':3,'EXCELLENT':4}
df_categorical.view.replace(to_replace=view_dict,inplace=True)
```

In [954...

```
plt.scatter(x=df['view'], y=df['price'])
```

Out[954...

```
<matplotlib.collections.PathCollection at 0x1edb2454910>
```



In [955…

```
df['heat_source'].unique()
```

Out[955…

```
array(['Gas', 'Oil', 'Electricity', 'Gas/Solar', 'Electricity/Solar',
       'Other', 'Oil/Solar'], dtype=object)
```

In [956…

```
heat_source_dummies = pd.get_dummies(df['heat_source'], prefix='heat_source',drop_first=True)
heat_source_dummies
```

Out[956…

	heat_source_Electricity/Solar	heat_source_Gas	heat_source_Gas/Solar	heat_source_Oil	heat_source_Oil/Solar	heat_source_Other
0	0	1	0	0	0	0
1	0	0	0	1	0	0
2	0	1	0	0	0	0
3	0	1	0	0	0	0
4	0	0	0	0	0	0
...	...	...	...	...	...	...
30150	0	0	0	1	0	0
30151	0	1	0	0	0	0
30152	0	1	0	0	0	0
30153	0	1	0	0	0	0
30154	0	0	0	1	0	0

29200 rows × 6 columns

In [957…

```
df['sewer_system'].unique()
```

Out[957…

```
array(['PUBLIC', 'PRIVATE', 'PRIVATE RESTRICTED', 'PUBLIC RESTRICTED'],
      dtype=object)
```

In [958…

```
sewer_dummies = pd.get_dummies(df['sewer_system'],prefix='sewer', drop_first=True)
sewer_dummies
```

Out[958…

	sewer_PRIVATE RESTRICTED	sewer_PUBLIC	sewer_PUBLIC RESTRICTED
0	0	1	0
1	0	1	0
2	0	1	0
3	0	1	0
4	0	1	0
...	...	...	...

	sewer_PRIVATE RESTRICTED	sewer_PUBLIC	sewer_PUBLIC RESTRICTED
30150	0	1	0
30151	0	1	0
30152	0	1	0
30153	0	1	0
30154	0	1	0

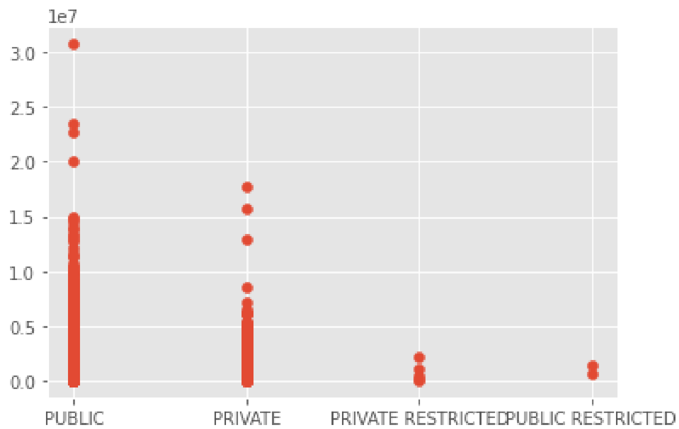
29200 rows × 3 columns

In [959…

```
plt.scatter(x=df['sewer_system'], y=df['price'])
```

Out[959…

```
<matplotlib.collections.PathCollection at 0x1edb234cdf0>
```



Developing categorical dataframe

In [960…

```
df_cat_pick = df_categorical[['waterfront','nuisance','view','greenbelt']]
```

Model #2

In [961…

```
model_2_data = pd.concat([df_numerical,sewer_dummies,heat_source_dummies, df_cat_pick], axis = 1)
```

In [962…

```
len(model_2_data) == len(waterfront_dummies)
```

Out[962…

```
True
```

In [963…

```
model_2_data.columns
```

Out[963…

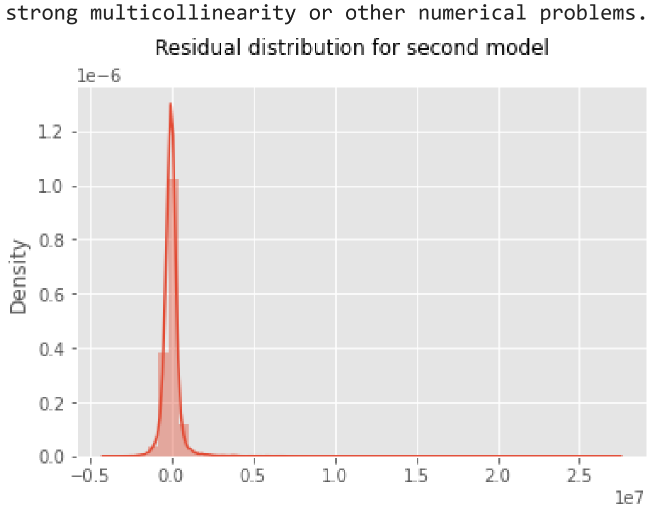
```
Index(['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',
       'condition', 'grade', 'sqft_above', 'sqft_basement', 'sqft_garage',
       'sqft_patio', 'yr_built', 'yr_renovated', 'lat', 'long',
       'school_rating', 'month', 'day_of_year', 'sewer_PRIVATE RESTRICTED',
       'sewer_PUBLIC', 'sewer_PUBLIC RESTRICTED',
       'heat_source_Electricity/Solar', 'heat_source_Gas',
       'heat_source_Gas/Solar', 'heat_source_Oil', 'heat_source_Oil/Solar',
       'heat_source_Other', 'waterfront', 'nuisance', 'view', 'greenbelt'],
      dtype='object')
```

In [964…

```
get_OLS_model('second',model_2_data, df['price'])
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.569
Model:                            OLS   Adj. R-squared:                  0.569
Method:                 Least Squares   F-statistic:                     1242.
```

```
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:38   Log-Likelihood:            -4.2933e+05
No. Observations:               29200   AIC:                         8.587e+05
Df Residuals:                   29168   BIC:                         8.590e+05
Df Model:                          31
Covariance Type:            nonrobust
===================================================================================================
                                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------------
const                         -7.214e+07   3.98e+06    -18.118      0.000   -7.99e+07   -6.43e+07
bedrooms                       -8.16e+04   4858.628    -16.794      0.000   -9.11e+04   -7.21e+04
bathrooms                       7.33e+04   7150.075     10.252      0.000    5.93e+04    8.73e+04
sqft_living                     168.4207     16.162     10.421      0.000     136.743     200.099
sqft_lot                          0.3840      0.062      6.211      0.000       0.263       0.505
floors                        -1.796e+05   9094.701    -19.743      0.000   -1.97e+05   -1.62e+05
condition                      5.751e+04   5506.503     10.445      0.000    4.67e+04    6.83e+04
grade                          1.749e+05   5312.776     32.921      0.000    1.64e+05    1.85e+05
sqft_above                      294.1133     16.507     17.818      0.000     261.759     326.468
sqft_basement                    63.3168     12.262      5.164      0.000      39.282      87.351
sqft_garage                     -90.1170     17.220     -5.233      0.000    -123.868     -56.366
sqft_patio                      129.3873     16.064      8.054      0.000      97.901     160.874
yr_built                      -1939.4403    183.414    -10.574      0.000   -2298.940   -1579.941
yr_renovated                     42.5945      8.855      4.810      0.000      25.238      59.951
lat                            9.677e+05   2.93e+04     32.985      0.000     9.1e+05    1.03e+06
long                          -2.321e+05   3.13e+04     -7.408      0.000   -2.93e+05   -1.71e+05
school_rating                  9.092e+04   2906.989     31.276      0.000    8.52e+04    9.66e+04
month                          1.939e+04   1.21e+04      1.608      0.108   -4241.247      4.3e+04
day_of_year                   -1224.3906    395.663     -3.095      0.002   -1999.908    -448.873
sewer_PRIVATE RESTRICTED      -1.274e+05   2.64e+05     -0.483      0.629   -6.44e+05    3.89e+05
sewer_PUBLIC                   1.705e+05   1.14e+04     14.905      0.000    1.48e+05    1.93e+05
sewer_PUBLIC RESTRICTED       -2.266e+04   4.16e+05     -0.054      0.957   -8.38e+05    7.93e+05
heat_source_Electricity/Solar -4.846e+04   7.84e+04     -0.618      0.536   -2.02e+05    1.05e+05
heat_source_Gas               -1606.2714   9352.052     -0.172      0.864   -1.99e+04    1.67e+04
heat_source_Gas/Solar          1.136e+05   6.17e+04      1.842      0.065   -7256.108    2.34e+05
heat_source_Oil               -2.621e+04   1.43e+04     -1.831      0.067   -5.43e+04    1840.620
heat_source_Oil/Solar         -9.824e+04   2.94e+05     -0.334      0.739   -6.75e+05    4.79e+05
heat_source_Other             -2.899e+04   1.32e+05     -0.219      0.826   -2.88e+05     2.3e+05
waterfront                     1.062e+06   2.95e+04     36.025      0.000       1e+06    1.12e+06
nuisance                       1.122e+04   9363.726      1.198      0.231   -7135.030    2.96e+04
view                           9.054e+04   4775.596     18.959      0.000    8.12e+04    9.99e+04
greenbelt                     -3.147e+04    2.2e+04     -1.429      0.153   -7.46e+04    1.17e+04
==============================================================================
Omnibus:                    46406.532   Durbin-Watson:                   1.898
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         98118370.070
Skew:                           9.814   Prob(JB):                         0.00
Kurtosis:                     286.302   Cond. No.                     7.31e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.31e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[964…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for second model'),
```

```
<AxesSubplot:ylabel='Density'>,
None)
```

In [965...

```python
get_error_metrics(model_2_data,df['price'])
```

```
Mean Squared Error:  345564825409.89886
Root Mean Squared Error:  587847.6209102992
Mean Average Percentage Error:  38.276485540425995 %
```

`heat_source`, `greenbelt` and `sewer_system` both have incredibly high p-values. These will be dropped from the final model if it holds.

Observations of Model 2

Model is still highly skewed although did present itself with some improvements. Next steps will be to normalize the data by transforming features that are skewed within the data, as well as remove outliers

- Durbin Watson score is in the acceptable range of 1.50-2.50
- Rsquared has 'improved' but only at the expense of the the continued flaws mentioned before.

Eliminating Outliers

To normalize the distribution, outlier removal will be the first step. An outlier will be defined as three standard deviations away from the mean of the target variable.

In [966...

```python
# Get the 3rd and 97th percentiles of the dependent variable column
lower_percentile = df['price'].quantile(0.03)
upper_percentile = df['price'].quantile(0.97)

# Filter the DataFrame to exclude values outside the lower and upper percentiles
df_outlier_removed = df[(df['price'] > lower_percentile) &
                 (df['price'] < upper_percentile)]

# Create a new DataFrame with only the dependent variable column
y = df_outlier_removed['price']

# Filter the model_2_data to exclude values outside the lower and upper percentiles
model_2_data_outlier_removed = model_2_data[(df['price'] > lower_percentile) &  (df['price'] < upper_percentile)]
```

In [967...

```python
df_outlier_removed
```

Out[967...

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	greenbelt	...	waterfront_loc	water_El
0	7399300360	2022-05-24	675000.0	4	1.0	1180	7140	1.0	NO	NO	...	other	
1	8910500230	2021-12-13	920000.0	5	2.5	2770	6703	1.0	NO	NO	...	other	
3	1604601802	2021-12-14	775000.0	3	3.0	2160	1400	2.0	NO	NO	...	other	
4	8562780790	2021-08-24	592500.0	2	2.0	1120	758	2.0	NO	NO	...	other	
5	2807100156	2021-07-20	625000.0	2	1.0	1190	5688	1.0	NO	NO	...	other	
...	...	...	...	...	...	...	...	...	...	...	...	...	
30150	7834800180	2021-11-30	1555000.0	5	2.0	1910	4000	1.5	NO	NO	...	other	
30151	194000695	2021-06-16	1313000.0	3	2.0	2020	5800	2.0	NO	NO	...	other	
30152	7960100080	2022-05-27	800000.0	3	2.0	1620	3600	1.0	NO	NO	...	other	

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	greenbelt	...	waterfront_loc	water_Ell
30153	2781280080	2022-02-24	775000.0	3	2.5	2570	2889	2.0	NO	NO	...	other	
30154	9557800100	2022-04-29	500000.0	3	1.5	1200	11058	1.0	NO	NO	...	other	

27446 rows × 36 columns

In [968...

```
waterfront_dummies = df_outlier_removed[['water_Elliot Bay','water_Lake Sammamish', 'water_Lake Washington','water_Pug
```

In [969...

```
df_outlier_removed.columns
```

Out[969...

```
Index(['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'greenbelt', 'nuisance', 'view',
       'condition', 'grade', 'heat_source', 'sewer_system', 'sqft_above',
       'sqft_basement', 'sqft_garage', 'sqft_patio', 'yr_built',
       'yr_renovated', 'address', 'lat', 'long', 'zipcode', 'waterfront_loc',
       'water_Elliot Bay', 'water_Lake Sammamish', 'water_Lake Union',
       'water_Lake Washington', 'water_Puget Sound', 'water_other',
       'school_rating', 'month', 'day_of_year'],
      dtype='object')
```

New look at model with removed outliers

In [970...

```
len(y)
```

Out[970...

```
27446
```

In [971...

```
outlier_data = pd.concat([y,model_2_data_outlier_removed], axis=1)
```

In [972...

```
outlier_data = outlier_data.drop('price', axis=1)
```

In [973...

```
len(outlier_data)
```

Out[973...

```
27446
```

In [974...

```
outlier_data
```

Out[974...

	bedrooms	bathrooms	sqft_living	sqft_lot	floors	condition	grade	sqft_above	sqft_basement	sqft_garage	...	heat_source_Electric
0	4	1.0	1180	7140	1.0	4	7	1180	0	0	...	
1	5	2.5	2770	6703	1.0	3	7	1570	1570	0	...	
3	3	3.0	2160	1400	2.0	3	9	1090	1070	200	...	
4	2	2.0	1120	758	2.0	3	7	1120	550	550	...	
5	2	1.0	1190	5688	1.0	3	7	1190	0	300	...	
...	...	...	...	...	...	...	...	...	...	...	...	
30150	5	2.0	1910	4000	1.5	4	8	1600	1130	0	...	
30151	3	2.0	2020	5800	2.0	3	7	2020	0	0	...	
30152	3	2.0	1620	3600	1.0	3	7	940	920	240	...	
30153	3	2.5	2570	2889	2.0	3	8	1830	740	480	...	
30154	3	1.5	1200	11058	1.0	3	7	1200	0	420	...	

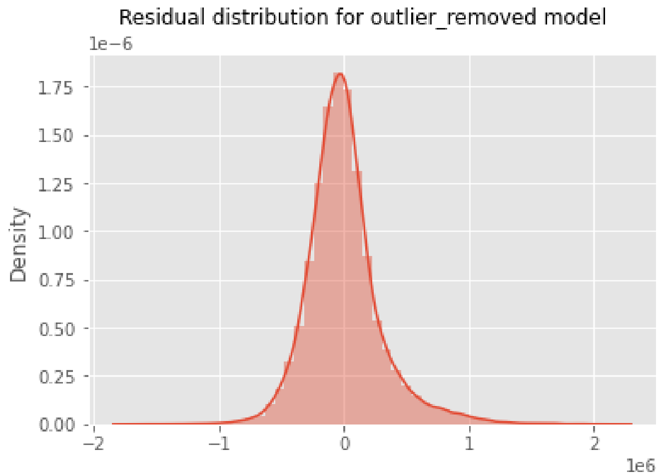
27446 rows × 31 columns

Model #3

In [975…

```
get_OLS_model('outlier_removed', outlier_data,y)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                  price   R-squared:                       0.667
Model:                            OLS   Adj. R-squared:                  0.667
Method:                 Least Squares   F-statistic:                     1774.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:39   Log-Likelihood:            -3.8514e+05
No. Observations:               27446   AIC:                         7.704e+05
Df Residuals:                   27414   BIC:                         7.706e+05
Df Model:                          31                                         
Covariance Type:            nonrobust                                         
===============================================================================================
                                  coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                       -4.382e+07   2.13e+06    -20.606      0.000    -4.8e+07   -3.97e+07
bedrooms                    -1.008e+04   2631.364     -3.830      0.000   -1.52e+04   -4920.150
bathrooms                    3.092e+04   3850.634      8.030      0.000    2.34e+04    3.85e+04
sqft_living                   144.6680      8.869     16.312      0.000     127.285     162.051
sqft_lot                        0.4440      0.034     12.967      0.000       0.377       0.511
floors                       -5.77e+04   4878.898    -11.827      0.000   -6.73e+04   -4.81e+04
condition                    5.669e+04   2922.313     19.398      0.000     5.1e+04    6.24e+04
grade                        1.406e+05   2883.414     48.755      0.000    1.35e+05    1.46e+05
sqft_above                    120.7283      9.133     13.218      0.000     102.826     138.630
sqft_basement                   8.7164      6.709      1.299      0.194      -4.434      21.867
sqft_garage                   -13.2045      9.255     -1.427      0.154     -31.345       4.936
sqft_patio                     49.1902      8.745      5.625      0.000      32.050      66.331
yr_built                    -1906.5707     97.896    -19.475      0.000   -2098.452   -1714.690
yr_renovated                   39.8656      4.777      8.345      0.000      30.502      49.229
lat                          9.899e+05   1.54e+04     64.206      0.000     9.6e+05    1.02e+06
long                         5020.0745   1.67e+04      0.300      0.764   -2.78e+04    3.78e+04
school_rating                 6.86e+04   1547.135     44.341      0.000    6.56e+04    7.16e+04
month                        1.622e+04   6361.851      2.550      0.011    3752.313    2.87e+04
day_of_year                 -1047.0839    208.798     -5.015      0.000   -1456.339    -637.829
sewer_PRIVATE RESTRICTED     2.742e+05   1.74e+05      1.576      0.115   -6.68e+04    6.15e+05
sewer_PUBLIC                 6.929e+04   6105.815     11.348      0.000    5.73e+04    8.13e+04
sewer_PUBLIC RESTRICTED     -1.611e+04   2.13e+05     -0.076      0.940   -4.33e+05    4.01e+05
heat_source_Electricity/Solar -5145.4791 4.56e+04     -0.113      0.910   -9.45e+04    8.42e+04
heat_source_Gas              2.939e+04   4944.984      5.943      0.000    1.97e+04    3.91e+04
heat_source_Gas/Solar        1.667e+05   3.34e+04      4.995      0.000    1.01e+05    2.32e+05
heat_source_Oil             -5327.3783   7576.169     -0.703      0.482   -2.02e+04    9522.295
heat_source_Oil/Solar       -2.859e+04   1.51e+05     -0.190      0.849   -3.24e+05    2.67e+05
heat_source_Other            1.934e+05   6.93e+04      2.790      0.005    5.75e+04    3.29e+05
waterfront                   1.779e+05   1.76e+04     10.095      0.000    1.43e+05    2.12e+05
nuisance                    -2.132e+04   4985.687     -4.275      0.000   -3.11e+04   -1.15e+04
view                         6.825e+04   2587.286     26.380      0.000    6.32e+04    7.33e+04
greenbelt                    5.804e+04   1.16e+04      5.011      0.000    3.53e+04    8.07e+04
==============================================================================
Omnibus:                     5752.577   Durbin-Watson:                   1.990
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            22199.937
Skew:                           1.007   Prob(JB):                         0.00
Kurtosis:                       6.918   Cond. No.                     7.06e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.06e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[975…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for outlier_removed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Looking at RMSE, MAE, MAPE

In [976…

```
get_error_metrics(outlier_data,y)
```

```
Mean Squared Error:  90418051636.01196
Root Mean Squared Error:  300695.9454931376
Mean Average Percentage Error:  22.330377104885844 %
```

In [977…

```
outlier_data.columns
```

Out[977…

```
Index(['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',
       'condition', 'grade', 'sqft_above', 'sqft_basement', 'sqft_garage',
       'sqft_patio', 'yr_built', 'yr_renovated', 'lat', 'long',
       'school_rating', 'month', 'day_of_year', 'sewer_PRIVATE RESTRICTED',
       'sewer_PUBLIC', 'sewer_PUBLIC RESTRICTED',
       'heat_source_Electricity/Solar', 'heat_source_Gas',
       'heat_source_Gas/Solar', 'heat_source_Oil', 'heat_source_Oil/Solar',
       'heat_source_Other', 'waterfront', 'nuisance', 'view', 'greenbelt'],
      dtype='object')
```

Observations of model 3

pvalue > 0.05

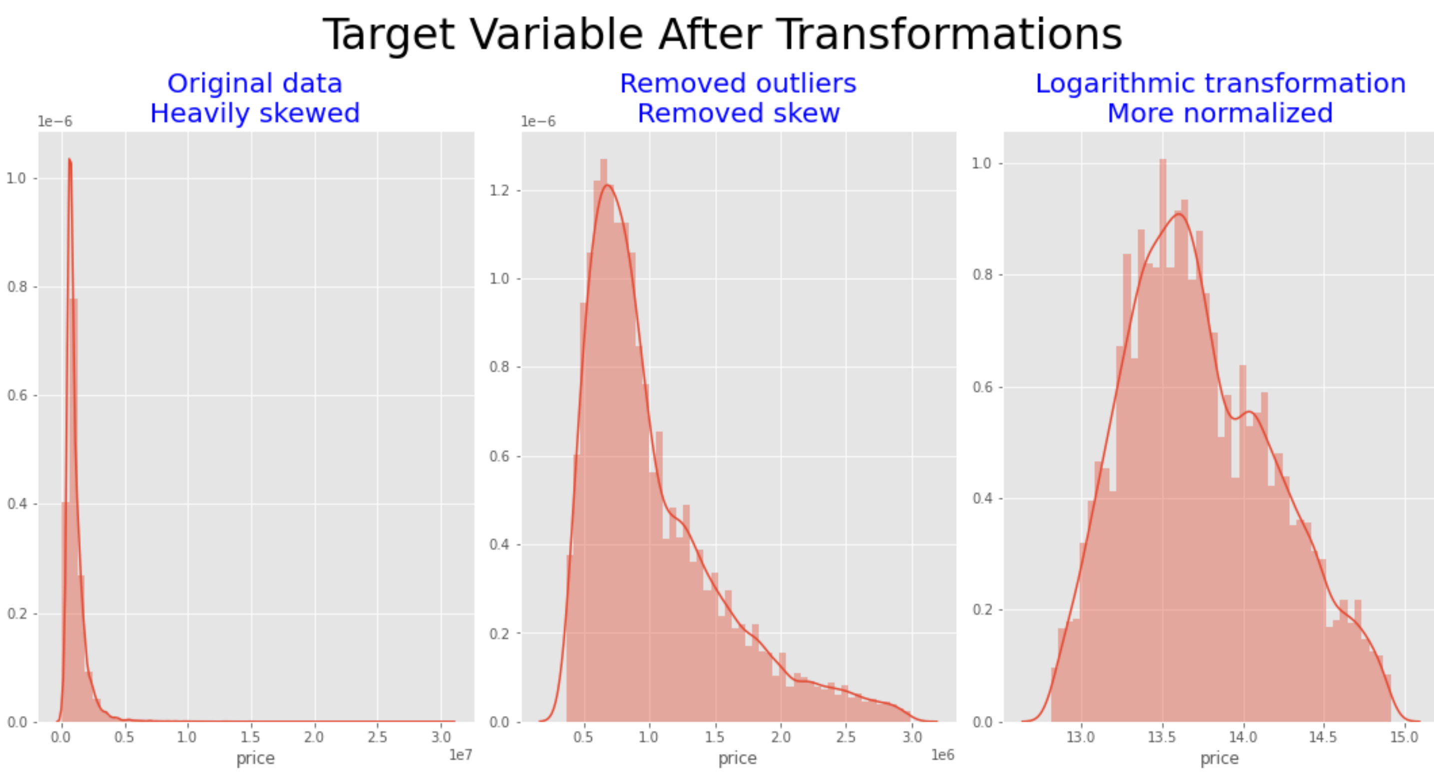
- `sqft_basement`
- `sqft_garage`
- `sewer_PRIVATE RESTRICTED`
- `sewer_PUBLIC RESTRICTED`
- `heat_source_Electricity/Solar`
- `heat_source_Oil/Solar`
- `heat_source_Other`

- Adjusted rsquared indicates that the model explains 62.2% of the data.
- Skewness has improved dramatically to an acceptable range between -2 and 2. The removal of outliers has made this possible.
- Durbin-Watson score is still in the acceptable ranges of 1.5-2.5
- Jarque-Bera score is still very high but has been brought down by a significant factor. Still not perfect but trending in the right direction.
- Multicollinearity is possibly present in the model and likely so given the initial VIFs before the first model was built. VIFS should be revisited again to see if those variables are worth keeping.

Looking at transformations for the price.

In [978…

```python
def plot_dist(ax, data, title):
    sns.distplot(data, ax=ax)
    ax.set_title(title, fontsize=20, color='b')
    ax.set_ylabel("")

# Create subplots
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(15, 8))

# Plot the original data
plot_dist(ax1, df['price'], "Original data\nHeavily skewed")


# Plot the data with outliers removed
plot_dist(ax2, y, "Removed outliers\nRemoved skew")

# Apply square root transformation to the data
y_log = np.log(y)

# Plot the transformed data
plot_dist(ax3, y_log, "Logarithmic transformation\nMore normalized")

# Set the overall title of the figure
fig.suptitle("Target Variable After Transformations", fontsize=32)

# Adjust the layout of the subplots
fig.tight_layout()

# Show the figure
plt.show()
```



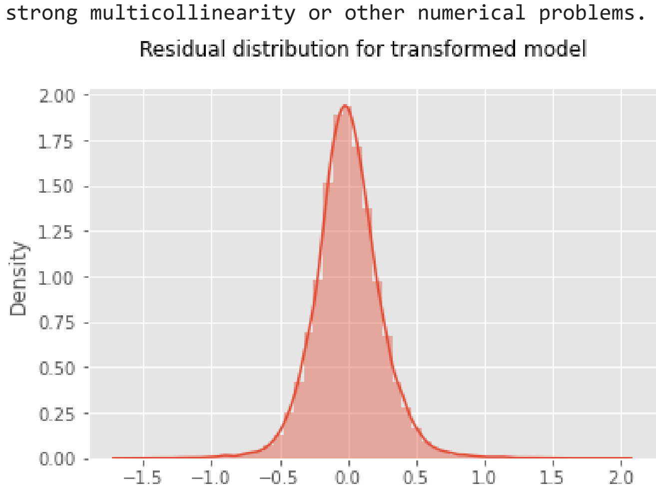
Checking model with transformed target variable - log transformation

In [979…

```python
get_OLS_model('transformed', outlier_data, y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.714
Model:                            OLS   Adj. R-squared:                  0.714
Method:                 Least Squares   F-statistic:                     2209.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:44   Log-Likelihood:                -345.54
```

```
No. Observations:               27446   AIC:                             755.1
Df Residuals:                   27414   BIC:                             1018.
Df Model:                          31
Covariance Type:            nonrobust
=================================================================================================
                                    coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------------
const                           -28.2127      1.733    -16.280      0.000     -31.609     -24.816
bedrooms                          0.0024      0.002      1.111      0.266      -0.002       0.007
bathrooms                         0.0371      0.003     11.814      0.000       0.031       0.043
sqft_living                    9.452e-05   7.23e-06     13.079      0.000    8.04e-05       0.000
sqft_lot                       4.482e-07   2.79e-08     16.062      0.000    3.94e-07    5.03e-07
floors                           -0.0238      0.004     -5.974      0.000      -0.032      -0.016
condition                         0.0536      0.002     22.529      0.000       0.049       0.058
grade                             0.1159      0.002     49.340      0.000       0.111       0.121
sqft_above                        0.0001   7.44e-06     15.062      0.000    9.75e-05       0.000
sqft_basement                  3.251e-05   5.47e-06      5.947      0.000    2.18e-05    4.32e-05
sqft_garage                   -7.241e-06   7.54e-06     -0.960      0.337    -2.2e-05    7.54e-06
sqft_patio                     5.187e-05   7.13e-06      7.279      0.000    3.79e-05    6.58e-05
yr_built                         -0.0015   7.98e-05    -19.285      0.000      -0.002      -0.001
yr_renovated                   3.309e-05   3.89e-06      8.502      0.000    2.55e-05    4.07e-05
lat                               1.0775      0.013     85.760      0.000       1.053       1.102
long                              0.0668      0.014      4.893      0.000       0.040       0.094
school_rating                     0.0604      0.001     47.910      0.000       0.058       0.063
month                             0.0161      0.005      3.099      0.002       0.006       0.026
day_of_year                      -0.0010      0.000     -5.934      0.000      -0.001      -0.001
sewer_PRIVATE RESTRICTED          0.1713      0.142      1.208      0.227      -0.107       0.449
sewer_PUBLIC                      0.0549      0.005     11.032      0.000       0.045       0.065
sewer_PUBLIC RESTRICTED           0.0328      0.173      0.189      0.850      -0.307       0.373
heat_source_Electricity/Solar     0.0069      0.037      0.186      0.852      -0.066       0.080
heat_source_Gas                   0.0392      0.004      9.727      0.000       0.031       0.047
heat_source_Gas/Solar             0.1338      0.027      4.921      0.000       0.081       0.187
heat_source_Oil                   0.0166      0.006      2.696      0.007       0.005       0.029
heat_source_Oil/Solar             0.0230      0.123      0.188      0.851      -0.217       0.264
heat_source_Other                 0.1724      0.057      3.052      0.002       0.062       0.283
waterfront                        0.1562      0.014     10.873      0.000       0.128       0.184
nuisance                         -0.0245      0.004     -6.034      0.000      -0.032      -0.017
view                              0.0562      0.002     26.667      0.000       0.052       0.060
greenbelt                         0.0476      0.009      5.040      0.000       0.029       0.066
==============================================================================
Omnibus:                     1920.408   Durbin-Watson:                   1.989
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             8710.668
Skew:                           0.183   Prob(JB):                         0.00
Kurtosis:                       5.736   Cond. No.                     7.06e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.06e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[979…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [980…

```
def get_log_mse(X, y):
    model = sm.OLS(y, sm.add_constant(X))
    result = model.fit()

    # Calculate the predicted values of the target variable using the linear model
    y_pred = result.predict(sm.add_constant(X))

    return mean_squared_log_error(y, y_pred)
get_log_mse(outlier_data, y_log)
```

Out[980…

```
0.00027353634108759735
```

In [981…

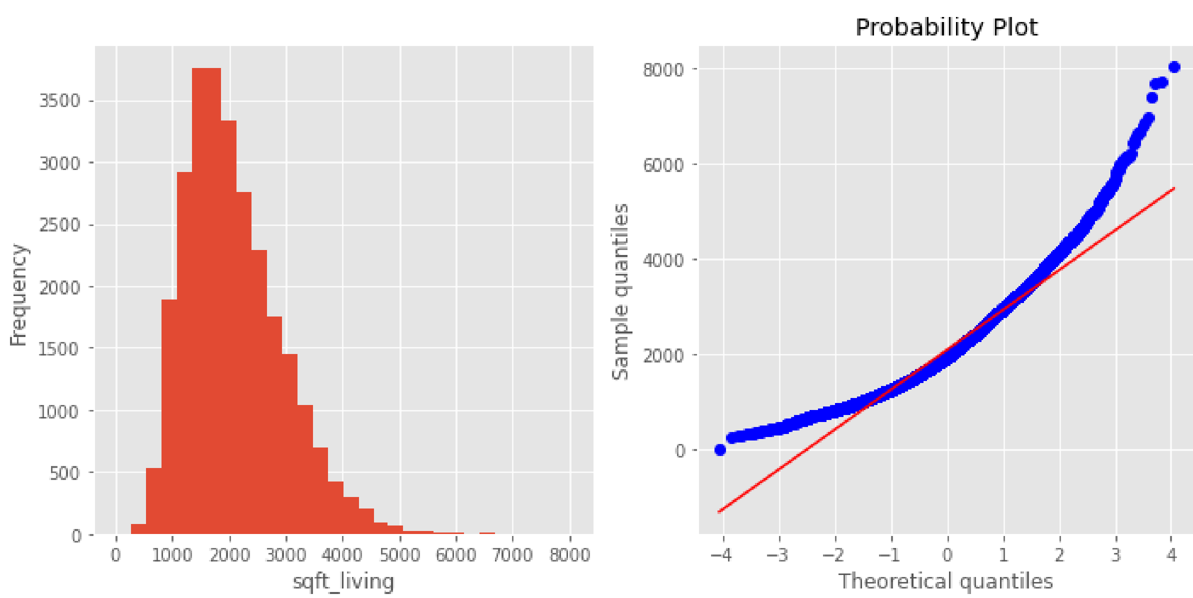
```
get_error_metrics(outlier_data, y)
```

```
Mean Squared Error:  90418051636.01196
Root Mean Squared Error:  300695.9454931376
Mean Average Percentage Error:  22.330377104885844 %
```

Checking distribution of predictor

In [982…

```
plot_hist_qq(outlier_data, 'sqft_living')
```



Data is clearly skewed right and follows an exponential pattern similar to price. For this, we will use a logarithmic transformation.

In [983…

```
outlier_data['sqft_garage']
```

Out[983…

```
0          0
1          0
3        200
4        550
5        300
        ...
30150      0
30151      0
30152    240
30153    480
30154    420
Name: sqft_garage, Length: 27446, dtype: int64
```

In [984…

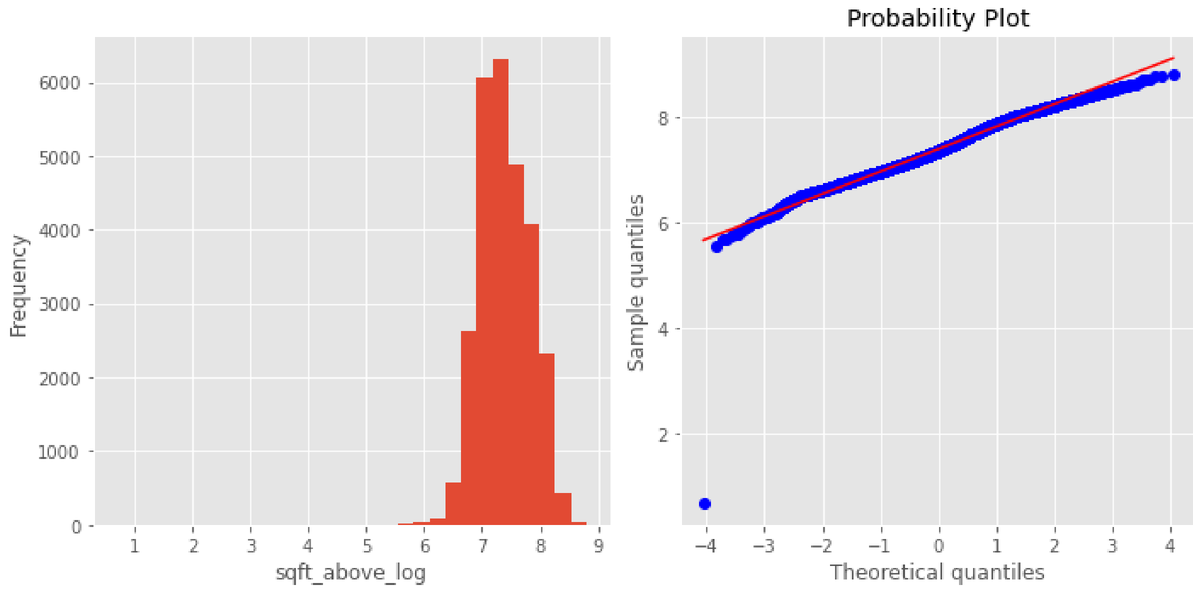
```
outlier_data['sqft_living_log'] = np.log(outlier_data['sqft_living'])
```

In [985…

```python
outlier_data['sqft_above_log'] = np.log(outlier_data['sqft_above'])
outlier_data['sqft_basement_log'] = np.log(outlier_data['sqft_basement'])
```

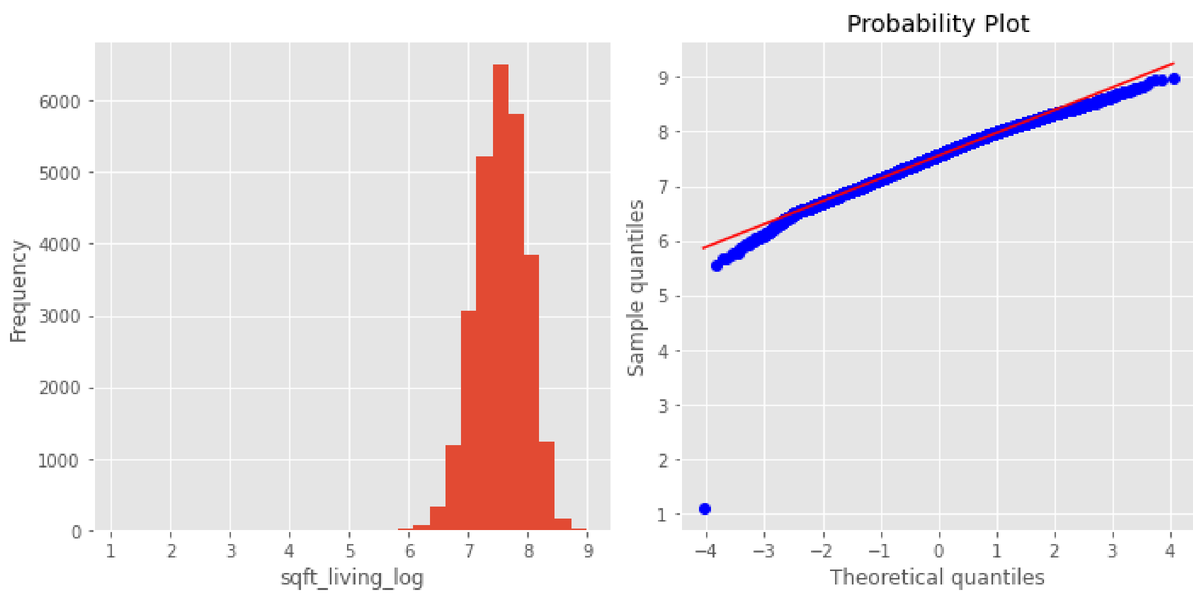
In [986…

```python
plot_hist_qq(outlier_data, 'sqft_above_log')
```



In [987…

```python
plot_hist_qq(outlier_data, 'sqft_living_log')
```



In [988…

```python
outlier_data = outlier_data.drop('sqft_living', axis=1)
```

In [989…

```python
outlier_data = outlier_data.drop(['sqft_basement_log','sqft_above'], axis=1)
```

In [990…

```python
outlier_data
```

Out[990…

	bedrooms	bathrooms	sqft_lot	floors	condition	grade	sqft_basement	sqft_garage	sqft_patio	yr_built	...	heat_source_Gas/Solar
0	4	1.0	7140	1.0	4	7	0	0	40	1969	...	0
1	5	2.5	6703	1.0	3	7	1570	0	240	1950	...	0
3	3	3.0	1400	2.0	3	9	1070	200	270	2010	...	0

	bedrooms	bathrooms	sqft_lot	floors	condition	grade	sqft_basement	sqft_garage	sqft_patio	yr_built	...	heat_source_Gas/Solar
4	2	2.0	758	2.0	3	7	550	550	30	2012	...	0
5	2	1.0	5688	1.0	3	7	0	300	0	1948	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
30150	5	2.0	4000	1.5	4	8	1130	0	210	1921	...	0
30151	3	2.0	5800	2.0	3	7	0	0	520	2011	...	0
30152	3	2.0	3600	1.0	3	7	920	240	110	1995	...	0
30153	3	2.5	2889	2.0	3	8	740	480	100	2006	...	0
30154	3	1.5	11058	1.0	3	7	0	420	0	1965	...	0

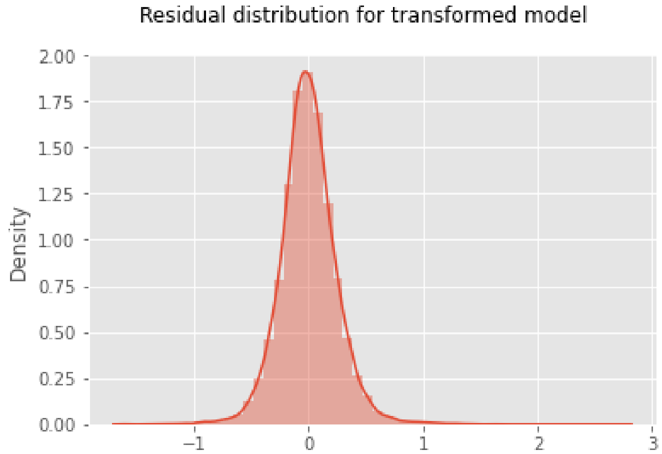
27446 rows × 31 columns

In [991…

```
get_OLS_model('transformed', outlier_data, y_log)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                  price   R-squared:                       0.709
Model:                            OLS   Adj. R-squared:                  0.709
Method:                 Least Squares   F-statistic:                     2157.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:48   Log-Likelihood:                -578.42
No. Observations:               27446   AIC:                             1221.
Df Residuals:                   27414   BIC:                             1484.
Df Model:                          31                                         
Covariance Type:            nonrobust                                         
===================================================================================================
                                      coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------------
const                             -30.3704      1.766    -17.199      0.000     -33.831     -26.909
bedrooms                           -0.0003      0.002     -0.135      0.893      -0.005       0.004
bathrooms                           0.0458      0.003     14.611      0.000       0.040       0.052
sqft_lot                         4.991e-07   2.81e-08     17.768      0.000    4.44e-07    5.54e-07
floors                             -0.0272      0.004     -6.727      0.000      -0.035      -0.019
condition                           0.0477      0.002     19.965      0.000       0.043       0.052
grade                               0.1229      0.002     52.294      0.000       0.118       0.127
sqft_basement                    2.667e-05   4.84e-06      5.511      0.000    1.72e-05    3.62e-05
sqft_garage                      6.813e-07   7.53e-06      0.090      0.928   -1.41e-05    1.54e-05
sqft_patio                       5.931e-05   7.17e-06      8.268      0.000    4.52e-05    7.34e-05
yr_built                           -0.0017      8e-05    -21.642      0.000      -0.002      -0.002
yr_renovated                     2.789e-05   3.92e-06      7.109      0.000    2.02e-05    3.56e-05
lat                                 1.0840      0.013     85.499      0.000       1.059       1.109
long                                0.0685      0.014      4.960      0.000       0.041       0.096
school_rating                       0.0614      0.001     48.211      0.000       0.059       0.064
month                               0.0170      0.005      3.245      0.001       0.007       0.027
day_of_year                        -0.0010      0.000     -6.021      0.000      -0.001      -0.001
sewer_PRIVATE RESTRICTED            0.2474      0.143      1.730      0.084      -0.033       0.528
sewer_PUBLIC                        0.0532      0.005     10.609      0.000       0.043       0.063
sewer_PUBLIC RESTRICTED             0.0119      0.175      0.068      0.946      -0.331       0.355
heat_source_Electricity/Solar       0.0009      0.037      0.023      0.982      -0.073       0.074
heat_source_Gas                     0.0325      0.004      7.959      0.000       0.024       0.040
heat_source_Gas/Solar               0.1343      0.027      4.898      0.000       0.081       0.188
heat_source_Oil                     0.0121      0.006      1.946      0.052   -8.58e-05       0.024
heat_source_Oil/Solar               0.0295      0.124      0.238      0.812      -0.213       0.272
heat_source_Other                   0.1744      0.057      3.061      0.002       0.063       0.286
waterfront                          0.1617      0.014     11.167      0.000       0.133       0.190
nuisance                           -0.0231      0.004     -5.634      0.000      -0.031      -0.015
view                                0.0579      0.002     27.293      0.000       0.054       0.062
greenbelt                           0.0555      0.010      5.839      0.000       0.037       0.074
sqft_living_log                     0.1676      0.013     12.999      0.000       0.142       0.193
sqft_above_log                      0.2059      0.012     17.334      0.000       0.183       0.229
==============================================================================
Omnibus:                     2607.514   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14094.894
Skew:                           0.301   Prob(JB):                         0.00
```

```
Kurtosis:                       6.459   Cond. No.                       7.13e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.13e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[991…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [992…

```python
get_error_metrics(outlier_data, y_log)
```

```
Mean Squared Error:  0.0610704553800042
Root Mean Squared Error:  0.2471243722905618
Mean Average Percentage Error:  1.329227711650429 %
```

In [993…

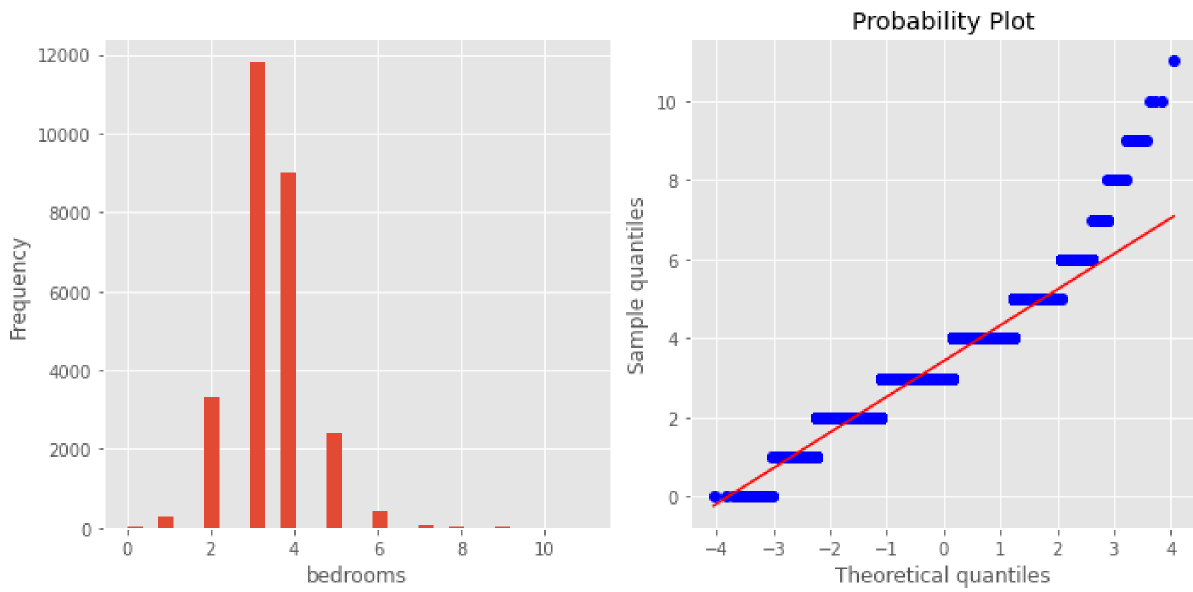
```python
get_log_mse(outlier_data, y_log)
```

Out[993…

```
0.00027919630760690826
```

In [994…

```python
plot_hist_qq(outlier_data, 'bedrooms')
```



pval > 0.05

- bedrooms - will be dropped from the current model

In [995…

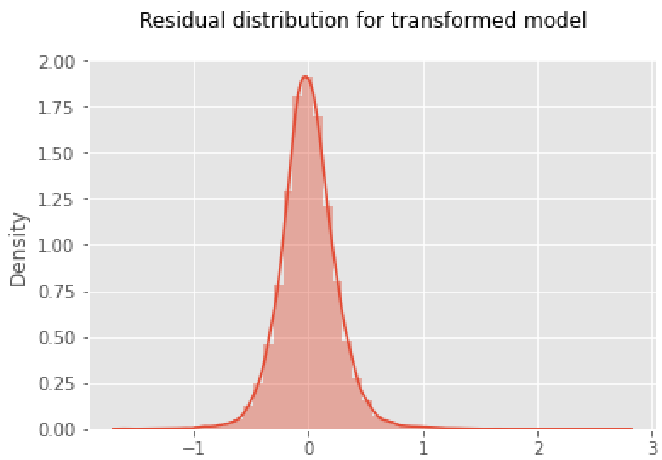
```python
outlier_data = outlier_data.drop(['bedrooms'], axis=1)
```

Rerun model

In [996…

```
get_OLS_model('transformed', outlier_data, y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.709
Model:                            OLS   Adj. R-squared:                  0.709
Method:                 Least Squares   F-statistic:                     2229.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:50   Log-Likelihood:                -578.43
No. Observations:               27446   AIC:                             1219.
Df Residuals:                   27415   BIC:                             1474.
Df Model:                          30
Covariance Type:            nonrobust
===================================================================================================
                                      coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------------
const                             -30.3715      1.766    -17.200      0.000     -33.832     -26.910
bathrooms                           0.0457      0.003     14.973      0.000       0.040       0.052
sqft_lot                         4.992e-07   2.81e-08     17.787      0.000    4.44e-07    5.54e-07
floors                             -0.0271      0.004     -6.733      0.000      -0.035      -0.019
condition                           0.0477      0.002     19.966      0.000       0.043       0.052
grade                               0.1229      0.002     52.807      0.000       0.118       0.127
sqft_basement                    2.667e-05   4.84e-06      5.511      0.000    1.72e-05    3.62e-05
sqft_garage                       7.06e-07   7.53e-06      0.094      0.925   -1.41e-05    1.55e-05
sqft_patio                       5.935e-05   7.16e-06      8.286      0.000    4.53e-05    7.34e-05
yr_built                           -0.0017   7.98e-05    -21.684      0.000      -0.002      -0.002
yr_renovated                     2.791e-05   3.92e-06      7.119      0.000    2.02e-05    3.56e-05
lat                                 1.0841      0.013     85.549      0.000       1.059       1.109
long                                0.0685      0.014      4.960      0.000       0.041       0.096
school_rating                       0.0614      0.001     48.237      0.000       0.059       0.064
month                               0.0170      0.005      3.244      0.001       0.007       0.027
day_of_year                        -0.0010      0.000     -6.021      0.000      -0.001      -0.001
sewer_PRIVATE RESTRICTED            0.2474      0.143      1.730      0.084      -0.033       0.528
sewer_PUBLIC                        0.0532      0.005     10.622      0.000       0.043       0.063
sewer_PUBLIC RESTRICTED             0.0117      0.175      0.067      0.947      -0.331       0.355
heat_source_Electricity/Solar       0.0008      0.037      0.022      0.982      -0.073       0.074
heat_source_Gas                     0.0325      0.004      7.959      0.000       0.024       0.040
heat_source_Gas/Solar               0.1344      0.027      4.900      0.000       0.081       0.188
heat_source_Oil                     0.0121      0.006      1.945      0.052   -9.56e-05       0.024
heat_source_Oil/Solar               0.0297      0.124      0.240      0.810      -0.213       0.272
heat_source_Other                   0.1745      0.057      3.062      0.002       0.063       0.286
waterfront                          0.1618      0.014     11.180      0.000       0.133       0.190
nuisance                           -0.0231      0.004     -5.635      0.000      -0.031      -0.015
view                                0.0580      0.002     27.392      0.000       0.054       0.062
greenbelt                           0.0556      0.010      5.840      0.000       0.037       0.074
sqft_living_log                     0.1672      0.013     13.322      0.000       0.143       0.192
sqft_above_log                      0.2058      0.012     17.342      0.000       0.183       0.229
==============================================================================
Omnibus:                     2606.012   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14078.132
Skew:                           0.301   Prob(JB):                         0.00
Kurtosis:                       6.457   Cond. No.                     7.13e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.13e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[996…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Dropping sewer/heat source data

In [997…

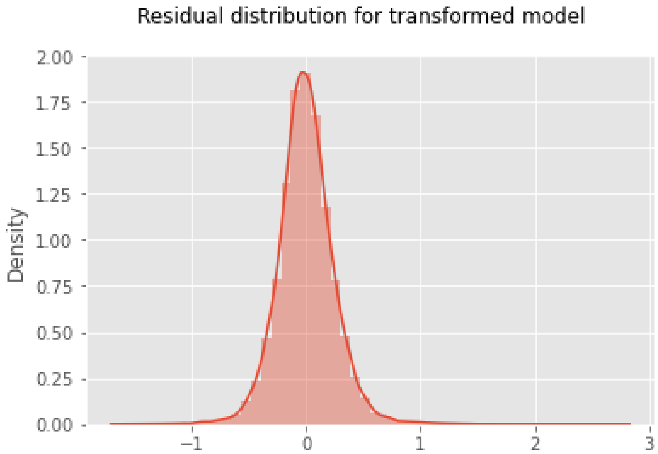
```python
new_outlier_data = outlier_data.drop(['sewer_PRIVATE RESTRICTED','sewer_PUBLIC RESTRICTED', 'heat_source_Oil', 'heat_s
```

In [998…

```python
get_OLS_model('transformed', new_outlier_data, y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.709
Model:                            OLS   Adj. R-squared:                  0.709
Method:                 Least Squares   F-statistic:                     2784.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:52   Log-Likelihood:                -586.36
No. Observations:               27446   AIC:                             1223.
Df Residuals:                   27421   BIC:                             1428.
Df Model:                          24
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                    -30.3368      1.766    -17.180      0.000     -33.798     -26.876
bathrooms                  0.0451      0.003     14.866      0.000       0.039       0.051
sqft_lot                5.018e-07    2.8e-08     17.900      0.000    4.47e-07    5.57e-07
floors                    -0.0275      0.004     -6.842      0.000      -0.035      -0.020
condition                  0.0470      0.002     19.866      0.000       0.042       0.052
grade                      0.1230      0.002     52.869      0.000       0.118       0.128
sqft_basement           2.744e-05   4.83e-06      5.688      0.000     1.8e-05    3.69e-05
sqft_garage             7.238e-07   7.53e-06      0.096      0.923    -1.4e-05    1.55e-05
sqft_patio              5.842e-05   7.15e-06      8.173      0.000    4.44e-05    7.24e-05
yr_built                  -0.0018   7.86e-05    -22.339      0.000      -0.002      -0.002
yr_renovated             2.71e-05    3.9e-06      6.945      0.000    1.94e-05    3.47e-05
lat                        1.0845      0.013     85.585      0.000       1.060       1.109
long                       0.0685      0.014      4.963      0.000       0.041       0.096
school_rating              0.0613      0.001     48.223      0.000       0.059       0.064
month                      0.0170      0.005      3.254      0.001       0.007       0.027
day_of_year               -0.0010      0.000     -6.034      0.000      -0.001      -0.001
sewer_PUBLIC               0.0533      0.005     10.662      0.000       0.043       0.063
heat_source_Gas            0.0285      0.004      7.931      0.000       0.021       0.035
heat_source_Gas/Solar      0.1304      0.027      4.765      0.000       0.077       0.184
waterfront                 0.1623      0.014     11.217      0.000       0.134       0.191
nuisance                  -0.0231      0.004     -5.636      0.000      -0.031      -0.015
view                       0.0582      0.002     27.501      0.000       0.054       0.062
greenbelt                  0.0557      0.010      5.854      0.000       0.037       0.074
sqft_living_log            0.1669      0.013     13.299      0.000       0.142       0.191
sqft_above_log             0.2074      0.012     17.530      0.000       0.184       0.231
==============================================================================
Omnibus:                     2616.004   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14156.302
```

```
Skew:                           0.302   Prob(JB):                         0.00
Kurtosis:                       6.466   Cond. No.                     7.13e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.13e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```



Out[998…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [999…

```
get_error_metrics(outlier_data, y_log)
```

```
Mean Squared Error:  0.06107049591260175
Root Mean Squared Error:  0.24712445429904695
Mean Average Percentage Error:  1.329220342637079 %
```

In [100…

```
get_error_metrics(outlier_data, y)
```

```
Mean Squared Error:  95220005558.32008
Root Mean Squared Error:  308577.3899013343
Mean Average Percentage Error:  23.22399188172752 %
```

Observations

- pval > 0.05

bedrooms - dropped from the current model

- all variables are statistically significant (pvalue < 0.05)
- Durbin-Watson Score continues to be "fine" but not improve a whole lot.
- Jarque-Bera Score continues to improve but still must come down
- skewness is now an afterthought as its at a very low -0.347 Overall no real improvement of the model happens here, we will try adding in new variables to improve as well as revisit VIFs to likely drop all that were originally at extremely high levels.

Next steps to improve the model:

1. revisit VIFs to see if any variables(now that outliers are removed and data has been transformed) should now be dropped from the model.
2. New predictors will be engineered to be added to the model. The next focus will be on the zipcodes in an attempt to narrow down the data with location-dependent price points. Possible data to be looked at are:

- waterfronts
- views
- school districts: rating, and school taxes
- tax brackets

Jarque-Beras score and skew level continue to improve but there is still some work to do.

Rechecking VIFs

In [100…

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Load your data into a pandas DataFrame
data = new_outlier_data

# Get a list of the column names
cols = data.columns

# Create an empty DataFrame to hold the VIF results
vif_data = pd.DataFrame()

# Loop through each column and calculate the VIF
for i in range(len(cols)):
    vif = variance_inflation_factor(data[cols].values, i)
    vif_data = vif_data.append({'Variable': cols[i], 'VIF': vif}, ignore_index=True)

# Print the VIF results
print(vif_data)
```

```
                Variable            VIF
0              bathrooms      24.585430
1               sqft_lot       1.272768
2                 floors      17.963674
3              condition      31.752875
4                  grade     136.876374
5          sqft_basement       5.547915
6            sqft_garage       4.712920
7             sqft_patio       2.268028
8               yr_built    9395.645709
9           yr_renovated       1.205844
10                   lat  155544.546934
11                  long  167482.287336
12         school_rating      26.465964
13                 month     698.007551
14           day_of_year     612.956164
15          sewer_PUBLIC       8.923106
16       heat_source_Gas       3.953563
17   heat_source_Gas/Solar     1.015903
18            waterfront       1.203274
19              nuisance       1.265885
20                  view       1.445364
21             greenbelt       1.069552
22       sqft_living_log    4042.140518
23        sqft_above_log    3389.641186
```

Scaling data

In [100…

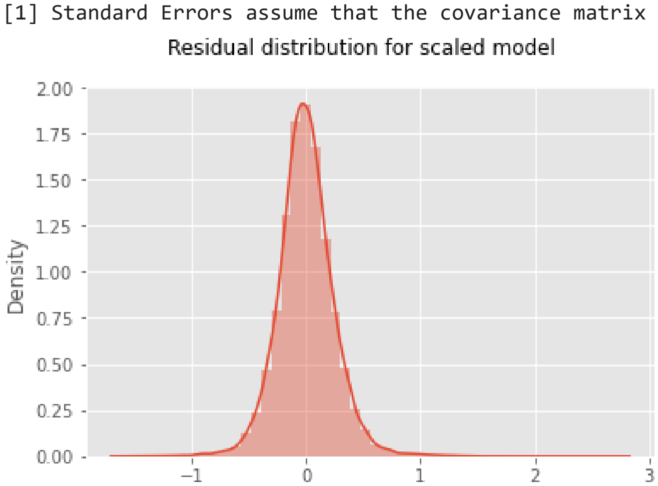
```
scaledX = (new_outlier_data - np.mean(new_outlier_data)) / np.std(new_outlier_data)
```

In [100…

```
get_OLS_model('scaled',scaledX, y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.709
Model:                            OLS   Adj. R-squared:                  0.709
Method:                 Least Squares   F-statistic:                     2784.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:30:56   Log-Likelihood:                -586.36
No. Observations:               27446   AIC:                             1223.
Df Residuals:                   27421   BIC:                             1428.
Df Model:                          24
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
```

```
==============================================================================
const                    13.7366      0.001   9201.923      0.000      13.734      13.739
bathrooms                 0.0372      0.003     14.866      0.000       0.032       0.042
sqft_lot                  0.0291      0.002     17.900      0.000       0.026       0.032
floors                   -0.0151      0.002     -6.842      0.000      -0.019      -0.011
condition                 0.0333      0.002     19.866      0.000       0.030       0.037
grade                     0.1295      0.002     52.869      0.000       0.125       0.134
sqft_basement             0.0154      0.003      5.688      0.000       0.010       0.021
sqft_garage               0.0002      0.002      0.096      0.923      -0.004       0.004
sqft_patio                0.0137      0.002      8.173      0.000       0.010       0.017
yr_built                 -0.0554      0.002    -22.339      0.000      -0.060      -0.051
yr_renovated              0.0112      0.002      6.945      0.000       0.008       0.014
lat                       0.1608      0.002     85.585      0.000       0.157       0.164
long                      0.0099      0.002      4.963      0.000       0.006       0.014
school_rating             0.0901      0.002     48.223      0.000       0.086       0.094
month                     0.0525      0.016      3.254      0.001       0.021       0.084
day_of_year              -0.0974      0.016     -6.034      0.000      -0.129      -0.066
sewer_PUBLIC              0.0189      0.002     10.662      0.000       0.015       0.022
heat_source_Gas           0.0132      0.002      7.931      0.000       0.010       0.016
heat_source_Gas/Solar     0.0072      0.002      4.765      0.000       0.004       0.010
waterfront                0.0183      0.002     11.217      0.000       0.015       0.021
nuisance                 -0.0086      0.002     -5.636      0.000      -0.012      -0.006
view                      0.0468      0.002     27.501      0.000       0.043       0.050
greenbelt                 0.0090      0.002      5.854      0.000       0.006       0.012
sqft_living_log           0.0700      0.005     13.299      0.000       0.060       0.080
sqft_above_log            0.0888      0.005     17.530      0.000       0.079       0.099
==============================================================================
Omnibus:                     2616.004   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14156.302
Skew:                           0.302   Prob(JB):                         0.00
Kurtosis:                       6.466   Cond. No.                         33.1
==============================================================================

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



Out[100…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for scaled model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [100…

```python
get_vifs(scaledX)
```

```
             Variable       VIF
0           bathrooms  2.813452
1            sqft_lot  1.185815
2              floors  2.180485
3           condition  1.264250
4               grade  2.691863
5       sqft_basement  3.294265
6         sqft_garage  1.953769
7          sqft_patio  1.260659
8            yr_built  2.758570
9        yr_renovated  1.157342
10                lat  1.583700
11               long  1.772648
```

```
12          school_rating    1.566587
13                  month  116.889211
14            day_of_year  116.882368
15           sewer_PUBLIC    1.405877
16        heat_source_Gas    1.233890
17  heat_source_Gas/Solar    1.012836
18             waterfront    1.188791
19               nuisance    1.055259
20                   view    1.300162
21              greenbelt    1.050283
22        sqft_living_log   12.424688
23         sqft_above_log   11.527536
```

Adding waterfront dummies to the model

In [100…

```
water_data = pd.concat([scaledX,waterfront_dummies], axis=1)
```

In [100…

```
water_data.columns
```

Out[100…

```
Index(['bathrooms', 'sqft_lot', 'floors', 'condition', 'grade',
       'sqft_basement', 'sqft_garage', 'sqft_patio', 'yr_built',
       'yr_renovated', 'lat', 'long', 'school_rating', 'month', 'day_of_year',
       'sewer_PUBLIC', 'heat_source_Gas', 'heat_source_Gas/Solar',
       'waterfront', 'nuisance', 'view', 'greenbelt', 'sqft_living_log',
       'sqft_above_log', 'water_Elliot Bay', 'water_Lake Sammamish',
       'water_Lake Washington', 'water_Puget Sound', 'water_other'],
      dtype='object')
```

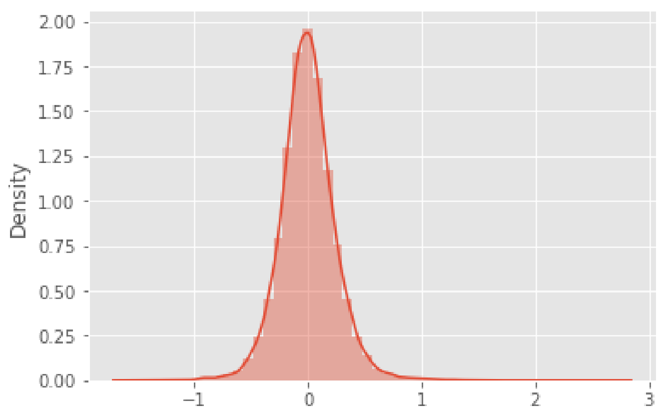
In [100…

```
get_OLS_model('waterfront',water_data,y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.712
Model:                            OLS   Adj. R-squared:                  0.712
Method:                 Least Squares   F-statistic:                     2340.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:31:00   Log-Likelihood:                -436.14
No. Observations:               27446   AIC:                             932.3
Df Residuals:                   27416   BIC:                             1179.
Df Model:                          29
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                    13.7420      0.012   1172.926      0.000      13.719      13.765
bathrooms                 0.0366      0.002     14.684      0.000       0.032       0.041
sqft_lot                  0.0295      0.002     18.231      0.000       0.026       0.033
floors                   -0.0156      0.002     -7.116      0.000      -0.020      -0.011
condition                 0.0341      0.002     20.437      0.000       0.031       0.037
grade                     0.1264      0.002     51.627      0.000       0.122       0.131
sqft_basement             0.0155      0.003      5.759      0.000       0.010       0.021
sqft_garage               0.0014      0.002      0.664      0.507      -0.003       0.005
sqft_patio                0.0146      0.002      8.775      0.000       0.011       0.018
yr_built                 -0.0533      0.002    -21.536      0.000      -0.058      -0.048
yr_renovated              0.0116      0.002      7.250      0.000       0.008       0.015
lat                       0.1665      0.002     84.883      0.000       0.163       0.170
long                      0.0055      0.002      2.721      0.007       0.002       0.009
school_rating             0.0853      0.002     43.651      0.000       0.081       0.089
month                     0.0529      0.016      3.296      0.001       0.021       0.084
day_of_year              -0.0977      0.016     -6.086      0.000      -0.129      -0.066
sewer_PUBLIC              0.0127      0.002      6.959      0.000       0.009       0.016
heat_source_Gas           0.0138      0.002      8.390      0.000       0.011       0.017
heat_source_Gas/Solar     0.0071      0.001      4.745      0.000       0.004       0.010
waterfront                0.0184      0.002     11.312      0.000       0.015       0.022
nuisance                 -0.0086      0.002     -5.639      0.000      -0.012      -0.006
view                      0.0463      0.002     27.320      0.000       0.043       0.050
greenbelt                 0.0091      0.002      5.950      0.000       0.006       0.012
sqft_living_log           0.0704      0.005     13.449      0.000       0.060       0.081
sqft_above_log            0.0899      0.005     17.826      0.000       0.080       0.100
water_Elliot Bay         -0.0203      0.015     -1.368      0.171      -0.049       0.009
```

```
water_Lake Sammamish      0.0862      0.015      5.843      0.000       0.057       0.115
water_Lake Washington    -0.1410      0.016     -8.555      0.000      -0.173      -0.109
water_Puget Sound        -0.0406      0.015     -2.692      0.007      -0.070      -0.011
water_other              -0.0052      0.012     -0.439      0.660      -0.028       0.018
==============================================================================
Omnibus:                     2744.128   Durbin-Watson:                   1.989
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            15638.027
Skew:                           0.313   Prob(JB):                         0.00
Kurtosis:                       6.645   Cond. No.                         43.8
==============================================================================

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

Residual distribution for waterfront model



Out[100…
```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Elliot Bay and Puget Sound present high pvalues indicating a lack of statistical significance. These will be dropped from the model.

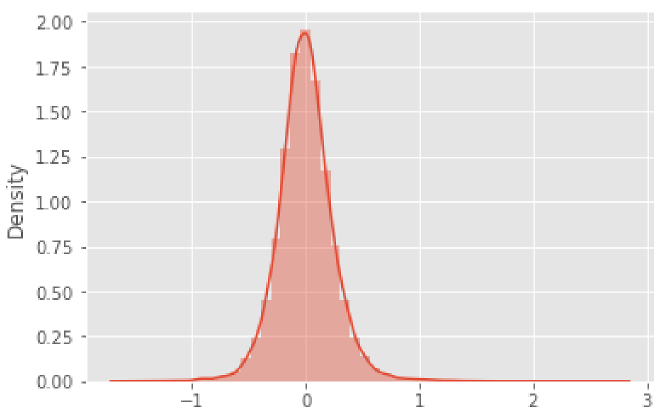
In [100…
```
water_data = water_data.drop(['water_other'], axis=1)
```

In [100…
```
get_OLS_model('waterfront',water_data,y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.712
Model:                            OLS   Adj. R-squared:                  0.712
Method:                 Least Squares   F-statistic:                     2423.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:31:01   Log-Likelihood:                -436.24
No. Observations:               27446   AIC:                             930.5
Df Residuals:                   27417   BIC:                             1169.
Df Model:                          28
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                    13.7369      0.002   8678.460      0.000      13.734      13.740
bathrooms                 0.0366      0.002     14.688      0.000       0.032       0.041
sqft_lot                  0.0295      0.002     18.227      0.000       0.026       0.033
floors                   -0.0156      0.002     -7.106      0.000      -0.020      -0.011
condition                 0.0341      0.002     20.433      0.000       0.031       0.037
grade                     0.1264      0.002     51.628      0.000       0.122       0.131
sqft_basement             0.0156      0.003      5.763      0.000       0.010       0.021
sqft_garage               0.0014      0.002      0.656      0.512      -0.003       0.005
sqft_patio                0.0146      0.002      8.770      0.000       0.011       0.018
yr_built                 -0.0534      0.002    -21.584      0.000      -0.058      -0.049
yr_renovated              0.0116      0.002      7.250      0.000       0.008       0.015
lat                       0.1666      0.002     84.998      0.000       0.163       0.170
long                      0.0055      0.002      2.712      0.007       0.002       0.009
school_rating             0.0851      0.002     44.149      0.000       0.081       0.089
month                     0.0529      0.016      3.296      0.001       0.021       0.084
day_of_year              -0.0977      0.016     -6.086      0.000      -0.129      -0.066
```

```
sewer_PUBLIC             0.0126      0.002      6.946      0.000       0.009       0.016
heat_source_Gas          0.0138      0.002      8.389      0.000       0.011       0.017
heat_source_Gas/Solar    0.0071      0.001      4.749      0.000       0.004       0.010
waterfront               0.0183      0.002     11.305      0.000       0.015       0.022
nuisance                -0.0086      0.002     -5.629      0.000      -0.012      -0.006
view                     0.0463      0.002     27.326      0.000       0.043       0.050
greenbelt                0.0091      0.002      5.956      0.000       0.006       0.012
sqft_living_log          0.0704      0.005     13.447      0.000       0.060       0.081
sqft_above_log           0.0899      0.005     17.826      0.000       0.080       0.100
water_Elliot Bay        -0.0154      0.010     -1.588      0.112      -0.034       0.004
water_Lake Sammamish     0.0916      0.008     11.018      0.000       0.075       0.108
water_Lake Washington   -0.1358      0.012    -11.806      0.000      -0.158      -0.113
water_Puget Sound       -0.0356      0.010     -3.631      0.000      -0.055      -0.016
==============================================================================
Omnibus:                     2742.889   Durbin-Watson:                   1.989
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            15630.594
Skew:                           0.312   Prob(JB):                         0.00
Kurtosis:                       6.644   Cond. No.                         33.2
==============================================================================

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

Residual distribution for waterfront model



Out[100…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Recheck VIFs

In [101…

```
get_vifs(water_data)
```

	Variable	VIF
0	bathrooms	2.815292
1	sqft_lot	1.189346
2	floors	2.184048
3	condition	1.265736
4	grade	2.718295
5	sqft_basement	3.304182
6	sqft_garage	1.966614
7	sqft_patio	1.262357
8	yr_built	2.772976
9	yr_renovated	1.158320
10	lat	1.737909
11	long	1.838453
12	school_rating	1.686288
13	month	116.907328
14	day_of_year	116.899793
15	sewer_PUBLIC	1.497687
16	heat_source_Gas	1.235765
17	heat_source_Gas/Solar	1.013087
18	waterfront	1.193535
19	nuisance	1.056759
20	view	1.302360
21	greenbelt	1.051093

```
22        sqft_living_log   12.432154
23         sqft_above_log   11.534209
24       water_Elliot Bay    1.067688
25   water_Lake Sammamish    1.197236
26  water_Lake Washington    1.163606
27      water_Puget Sound    1.048373
```

Month and day_of_year present with high variance inflation factors indicating possible collinearity. These will be dropped.

In [101…

```
water_data = water_data.drop(['month','day_of_year','sqft_living_log'], axis =1)
```

In [101…

```
get_vifs(water_data)
```

```
                 Variable       VIF
0               bathrooms  2.578357
1                sqft_lot  1.189172
2                  floors  2.127025
3               condition  1.245391
4                   grade  2.682408
5           sqft_basement  1.739542
6             sqft_garage  1.931111
7              sqft_patio  1.260893
8                yr_built  2.742945
9            yr_renovated  1.156881
10                    lat  1.737904
11                   long  1.836777
12          school_rating  1.685785
13           sewer_PUBLIC  1.497604
14        heat_source_Gas  1.231800
15  heat_source_Gas/Solar  1.013061
16             waterfront  1.193478
17               nuisance  1.056396
18                   view  1.302207
19              greenbelt  1.051015
20         sqft_above_log  3.224589
21       water_Elliot Bay  1.067596
22   water_Lake Sammamish  1.196869
23  water_Lake Washington  1.163047
24      water_Puget Sound  1.048124
```

All VIFs are now below 3 aside from sqft_above, meaning the issue of collinearity is now for the most part solved.

Final model

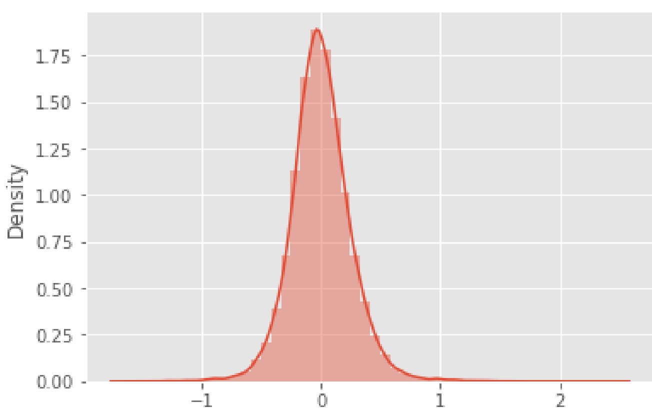
In [101…

```
get_OLS_model('waterfront',water_data,y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.701
Model:                            OLS   Adj. R-squared:                  0.700
Method:                 Least Squares   F-statistic:                     2566.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:31:10   Log-Likelihood:                -982.25
No. Observations:               27446   AIC:                             2017.
Df Residuals:                   27420   BIC:                             2230.
Df Model:                          25
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                    13.7370      0.002   8508.481      0.000      13.734      13.740
bathrooms                 0.0447      0.002     18.386      0.000       0.040       0.049
sqft_lot                  0.0289      0.002     17.493      0.000       0.026       0.032
floors                   -0.0196      0.002     -8.886      0.000      -0.024      -0.015
condition                 0.0368      0.002     21.792      0.000       0.034       0.040
grade                     0.1305      0.002     52.623      0.000       0.126       0.135
sqft_basement             0.0404      0.002     20.237      0.000       0.037       0.044
sqft_garage              -0.0018      0.002     -0.856      0.392      -0.006       0.002
sqft_patio                0.0152      0.002      8.915      0.000       0.012       0.018
```

```
yr_built                 -0.0505      0.003    -20.125      0.000      -0.055      -0.046
yr_renovated              0.0123      0.002      7.546      0.000       0.009       0.015
lat                       0.1666      0.002     83.362      0.000       0.163       0.171
long                      0.0068      0.002      3.336      0.001       0.003       0.011
school_rating             0.0847      0.002     43.044      0.000       0.081       0.089
sewer_PUBLIC              0.0130      0.002      6.983      0.000       0.009       0.017
heat_source_Gas           0.0150      0.002      8.941      0.000       0.012       0.018
heat_source_Gas/Solar     0.0071      0.002      4.643      0.000       0.004       0.010
waterfront                0.0182      0.002     10.994      0.000       0.015       0.021
nuisance                 -0.0095      0.002     -6.108      0.000      -0.013      -0.006
view                      0.0458      0.002     26.524      0.000       0.042       0.049
greenbelt                 0.0093      0.002      5.973      0.000       0.006       0.012
sqft_above_log            0.1471      0.003     54.081      0.000       0.142       0.152
water_Elliot Bay         -0.0143      0.010     -1.445      0.149      -0.034       0.005
water_Lake Sammamish      0.0912      0.008     10.754      0.000       0.075       0.108
water_Lake Washington    -0.1357      0.012    -11.564      0.000      -0.159      -0.113
water_Puget Sound        -0.0396      0.010     -3.964      0.000      -0.059      -0.020
==============================================================================
Omnibus:                     2322.832   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            10761.426
Skew:                           0.294   Prob(JB):                         0.00
Kurtosis:                       6.011   Cond. No.                         15.8
==============================================================================

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

Residual distribution for waterfront model




```
Out[101…  (None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

P-values of sqft_garage and Elliot bay are too high, to be dropped

In [101…

```
water_data = water_data.drop(['sqft_garage','water_Elliot Bay'], axis=1)
```

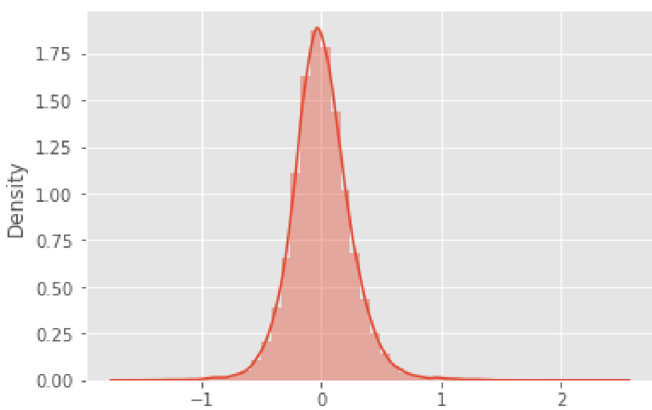
In [101…

```
get_OLS_model('waterfront',water_data,y_log)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.701
Model:                            OLS   Adj. R-squared:                  0.700
Method:                 Least Squares   F-statistic:                     2789.
Date:                Sat, 18 Mar 2023   Prob (F-statistic):               0.00
Time:                        23:31:11   Log-Likelihood:                -983.64
No. Observations:               27446   AIC:                             2015.
Df Residuals:                   27422   BIC:                             2213.
Df Model:                          23
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                    13.7366      0.002   8623.351      0.000      13.733      13.740
bathrooms                 0.0446      0.002     18.348      0.000       0.040       0.049
```

```
sqft_lot                 0.0289      0.002     17.486      0.000       0.026       0.032
floors                  -0.0193      0.002     -9.018      0.000      -0.023      -0.015
condition                0.0368      0.002     21.778      0.000       0.033       0.040
grade                    0.1302      0.002     53.138      0.000       0.125       0.135
sqft_basement            0.0404      0.002     20.243      0.000       0.036       0.044
sqft_patio               0.0151      0.002      8.896      0.000       0.012       0.018
yr_built                -0.0509      0.002    -21.078      0.000      -0.056      -0.046
yr_renovated             0.0123      0.002      7.568      0.000       0.009       0.016
lat                      0.1662      0.002     84.831      0.000       0.162       0.170
long                     0.0067      0.002      3.271      0.001       0.003       0.011
school_rating            0.0851      0.002     43.804      0.000       0.081       0.089
sewer_PUBLIC             0.0130      0.002      6.997      0.000       0.009       0.017
heat_source_Gas          0.0149      0.002      8.925      0.000       0.012       0.018
heat_source_Gas/Solar    0.0071      0.002      4.665      0.000       0.004       0.010
waterfront               0.0183      0.002     11.037      0.000       0.015       0.021
nuisance                -0.0094      0.002     -6.045      0.000      -0.012      -0.006
view                     0.0458      0.002     26.541      0.000       0.042       0.049
greenbelt                0.0092      0.002      5.956      0.000       0.006       0.012
sqft_above_log           0.1466      0.003     55.556      0.000       0.141       0.152
water_Lake Sammamish     0.0908      0.008     10.714      0.000       0.074       0.107
water_Lake Washington   -0.1358      0.012    -11.604      0.000      -0.159      -0.113
water_Puget Sound       -0.0397      0.010     -3.977      0.000      -0.059      -0.020
==============================================================================
Omnibus:                     2319.874   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            10720.794
Skew:                           0.294   Prob(JB):                         0.00
Kurtosis:                       6.005   Cond. No.                         14.9
==============================================================================

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

Residual distribution for waterfront model



Out[101…

```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [101…

```python
model = sm.OLS(y_log, sm.add_constant(water_data))
results = model.fit()
model_residual = results.resid
model_params = results.params

print(results.params)
```

```
const                    13.736608
bathrooms                 0.044602
sqft_lot                  0.028880
floors                   -0.019251
condition                 0.036807
grade                     0.130172
sqft_basement             0.040365
sqft_patio                0.015128
yr_built                 -0.050884
yr_renovated              0.012325
lat                       0.166207
long                      0.006676
```

```
school_rating            0.085132
sewer_PUBLIC             0.012971
heat_source_Gas          0.014910
heat_source_Gas/Solar    0.007111
waterfront               0.018251
nuisance                -0.009396
view                     0.045831
greenbelt                0.009246
sqft_above_log           0.146602
water_Lake Sammamish     0.090771
water_Lake Washington   -0.135775
water_Puget Sound       -0.039698
dtype: float64
```

Raw Predictor vs. Log Transformed Target

We are modeling the relationship as:

$$log(y) = \beta x \ldots$$

For **small** values of β, we can interpret β as:

> For each increase of 1 unit in x, we see an associated change of $(\beta * 100)\%$ in y

Since the coefficients seem **small** due to them being scaled as z-scores, we will just multiply by 100 to convert to percentages.

In [101…

```python
# Initializing Pandas DataFrame of params
params = pd.DataFrame(results.params)
params = params.reset_index()

#renaming columnss
params.columns = ['variable', 'coefficient']

#Labeling params as positive or negative
params['correlation'] = params['coefficient'].apply(lambda x: 'Positive' if x > 0 else 'Negative' if x<0 else x)

#Converting to absolute values
params['coefficient'] = params['coefficient'].apply(lambda x: abs(x))

#Calculating Percent Change
params['percent_change'] = round(params["coefficient"] * 100, 2)

# Sorting the 'params' in order of highest correlations, negative or positive.
params = params.sort_values('percent_change', ascending=False)
params[1:]
```

Out[101…

	variable	coefficient	correlation	percent_change
10	lat	0.166207	Positive	16.62
20	sqft_above_log	0.146602	Positive	14.66
22	water_Lake Washington	0.135775	Negative	13.58
5	grade	0.130172	Positive	13.02
21	water_Lake Sammamish	0.090771	Positive	9.08
12	school_rating	0.085132	Positive	8.51
8	yr_built	0.050884	Negative	5.09
18	view	0.045831	Positive	4.58
1	bathrooms	0.044602	Positive	4.46
6	sqft_basement	0.040365	Positive	4.04
23	water_Puget Sound	0.039698	Negative	3.97
4	condition	0.036807	Positive	3.68
2	sqft_lot	0.028880	Positive	2.89

	variable	coefficient	correlation	percent_change
3	floors	0.019251	Negative	1.93
16	waterfront	0.018251	Positive	1.83
7	sqft_patio	0.015128	Positive	1.51
14	heat_source_Gas	0.014910	Positive	1.49
13	sewer_PUBLIC	0.012971	Positive	1.30
9	yr_renovated	0.012325	Positive	1.23
17	nuisance	0.009396	Negative	0.94
19	greenbelt	0.009246	Positive	0.92
15	heat_source_Gas/Solar	0.007111	Positive	0.71
11	long	0.006676	Positive	0.67

Iterating through the dataframe to write out interpretations

In [101…

```
for i, row in params.iterrows():
    if i == 0:
        pass
    elif row.correlation =='Positive':
        print(f'As {params.variable[i]} increases by 1 standard deviation, the price of a home increases by {round(par
    else:
        print(f'As {params.variable[i]} increases by 1 standard deviation, the price of a home decreases by {round(par
```

```
As lat increases by 1 standard deviation, the price of a home increases by 16.62%

As sqft_above_log increases by 1 standard deviation, the price of a home increases by 14.66%

As water_Lake Washington increases by 1 standard deviation, the price of a home decreases by 13.58%

As grade increases by 1 standard deviation, the price of a home increases by 13.02%

As water_Lake Sammamish increases by 1 standard deviation, the price of a home increases by 9.08%

As school_rating increases by 1 standard deviation, the price of a home increases by 8.51%

As yr_built increases by 1 standard deviation, the price of a home decreases by 5.09%

As view increases by 1 standard deviation, the price of a home increases by 4.58%

As bathrooms increases by 1 standard deviation, the price of a home increases by 4.46%

As sqft_basement increases by 1 standard deviation, the price of a home increases by 4.04%

As water_Puget Sound increases by 1 standard deviation, the price of a home decreases by 3.97%

As condition increases by 1 standard deviation, the price of a home increases by 3.68%

As sqft_lot increases by 1 standard deviation, the price of a home increases by 2.89%

As floors increases by 1 standard deviation, the price of a home decreases by 1.93%

As waterfront increases by 1 standard deviation, the price of a home increases by 1.83%

As sqft_patio increases by 1 standard deviation, the price of a home increases by 1.51%

As heat_source_Gas increases by 1 standard deviation, the price of a home increases by 1.49%

As sewer_PUBLIC increases by 1 standard deviation, the price of a home increases by 1.3%

As yr_renovated increases by 1 standard deviation, the price of a home increases by 1.23%

As nuisance increases by 1 standard deviation, the price of a home decreases by 0.94%

As greenbelt increases by 1 standard deviation, the price of a home increases by 0.92%
```

`As heat_source_Gas/Solar increases by 1 standard deviation, the price of a home increases by 0.71%`

`As long increases by 1 standard deviation, the price of a home increases by 0.67%`

Final Observations

The results suggest that the most influential predictors on home price are:

`Latitude (lat)` : As latitude increases by 1 standard deviation, the price of a home increases by 16.62%. `Grade` : As grade increases by 1 standard deviation, the price of a home increases by 13.02%. `Square footage of aside from basement (sqft_above_log)` : As sqft_above_log increases by 1 standard deviation, the price of a home increases by 14.66%. Other predictors that have a significant effect on home price include:

`Water features` : Lake Sammamish increases home price by 9.08%, while Lake Washington decreases it by 13.58% and Puget Sound decreases it by 3.97%. `School rating` : As school rating increases by 1 standard deviation, the price of a home increases by 8.51%. `Bathrooms` : As the number of bathrooms increases by 1 standard deviation, the price of a home increases by 4.46%. `View` : As the view score increases by 1 standard deviation, the price of a home increases by 4.58%. `Square footage of basement area (sqft_basement)` : As sqft_basement increases by 1 standard deviation, the price of a home increases by 4.04%. `Condition` : As condition increases by 1 standard deviation, the price of a home increases by 3.68%. `Square footage of the lot (sqft_lot)` : As sqft_lot increases by 1 standard deviation, the price of a home increases by 2.89%. `Longitude (long)` : As `longitude` increases by 1 standard deviation, the price of a home increases by .67%. Some predictors have a smaller effect on home price, such as `waterfront` increasing the price by 1.83%, while `floors` decreasing it by 1.93%.

Conclusion

This entire process included the above described data engineering techniques, as well as an extensive look at transforming variables, feature selection and elimination through trial and error. Different transformations on the price for example were attempted to normalize the distribution, but the decision was made to use the square root transformation as it lended itself to dealing with the upper and lower tails of the distribution of the price more efficiently.

The rest of the data cleaning process also included dropping or filling in of missing values, removal of outliers, one hot encoding categorical variables as well as dropping all variables that presented themselves with a high variance inflation factor. The use of QQplots and histograms were used to check the distribution of residuals.

Four assumptions

The decisions of the creation of this model were based on the four assumptions with the method of justification, which are:

1. Linearity assumption: Scatterplots of the variables should represent some level of linearity aka correlation to the target variable. The use of scatterplots and correlation matrices were the method of justification for this assumption.

2. Normality Assumption: Use of QQplots and Residual histograms were the method for justifying normality. The normality assumption in linear regression models refers to the assumption that the residuals (i.e., the difference between the observed and predicted values) are normally distributed. This assumption is important because many statistical tests and procedures used in linear regression models rely on the assumption of normality, including hypothesis testing, confidence intervals, and prediction intervals.

One way to check for normality of the residuals is by using a Q-Q plot (quantile-quantile plot). A Q-Q plot is a graphical tool that compares the distribution of the residuals to a normal distribution. If the residuals are normally distributed, the points in the Q-Q plot should follow a straight line. Any deviation from the straight line indicates that the residuals are not normally distributed. Specifically, if the points in the Q-Q plot deviate from the straight line in the tails of the distribution, it suggests that the residuals have heavier tails than a normal distribution. On the other hand, if the points in the Q-Q plot deviate from the straight line in the center of the distribution, it suggests that the residuals have a skewed distribution.

Another way to check for normality of the residuals is by using a histogram. A histogram is a graph that shows the frequency distribution of the residuals. If the residuals are normally distributed, the histogram should have a bell-shaped curve, with the majority of the residuals near the mean and fewer residuals towards the tails of the distribution. Any deviation from the bell-shaped curve indicates that the residuals are not normally distributed.

1. Homoscedasticity: Durbin-Watson score close to 2 indicates that the errors are approximately normally distributed with constant variance.

A Durbin-Watson score close to 2 generally indicates that the model's errors are approximately homoscedastic (i.e., the variance of the errors is constant across all levels of the predictor variables).

This is because a score close to 2 suggests that the errors are approximately normally distributed with constant variance, which is a key assumption of many regression models. When the errors are homoscedastic, it means that the variability of the dependent variable is relatively constant across different levels of the independent variables, which allows for more accurate estimation of the coefficients and predictions of the model.

1. Multicollinearity: VIF scores for independent variables.

Variance Inflation Factors (VIFs) are a measure used to assess the degree of multicollinearity in a multiple regression model. Multicollinearity occurs when two or more independent variables in a regression model are highly correlated with each other, which can make it difficult to interpret the individual effects of each variable on the dependent variable.

Final assessment and thoughts of the model

- When plotted the model residuals show a normalized distribution, satisfying the normality assumption although with fairly long tails. This is something to be added into the future work.
- The Durbin-Watson score is almost exactly 2, satisfying the assumption of homoscedasticity.
- The linearity assumption was satisfied through a rigorous look at the plots and in general can be a difficult metric to validate.
- The skew level of the data is well within the acceptable range of -2 to 2 which is a vast improvement from the original model which was originally above 10.
- The kurtosis level is inside the acceptable range of -7 to 7 at a score of ~6.
- The Jarque Beras score is a massive 10720 which will require further investigation for future work. My instincts tell me there still may be some major outliers that may be affecting this score as the QQplots appear to be for the most part okay.
- The assumption of no multicollinearity is satisfied due to the VIFs all remaining under 3 and the OLS model shows no signs as well.

Recommendations

For the purposes of Zillows ability to choose inventory in the King County Real Estate Market, I recommend looking at properties that are near Lake Samammish or that are further north that also is accompanied with a waterfront. Since the grade, condition, and number of bathrooms appear positively correlated to the price it would make sense to try and buy older homes in the aforementioned areas as older homes tend to be cheaper in terms of price. Taking these homes and ensuring the grade and condition are of high quality through either pre-assessed purchases or renovations, along with possibly adding bathrooms can raise the price for resell value. Picking houses near school districts of high rating can have an impact as well.

Houses towards the west as well as ones that present nuisances clearly result in lower prices, so my recommendation would be to avoid buying houses that fit these parameters as it may result in "holding the bag" scenarios which could lead to longer times held with inventory.

The only invalid metric that should probably be ignored for now(but explored further) is the size of the garage negatively affecting the price. I would not consider this to be an accurate assessment, but I do not want to exclude it from the model.

Questions model can answer:

- `Should the house be on a waterfront?`
- `How far north should the houses be?`

- What age should the house have?
- What level of renovations need to be performed on the houses, and when?
- Will the house price be affected by common nuisances? (eg. noise, construction, bugs)
- How far west should the house be before one should lose interest of the purchase?
- What average quality of schools in the surrounding area?

Future Work

In the future work, it is worth revisiting the value of the homes on the remaining waterfronts and seeing if there is any statistical significance. More exploration is needed but was not ready to be presented at this time.

The views that are highlighted in the column_names.md documentation can be explored and onehotencoded and could be a potential candidate feature.

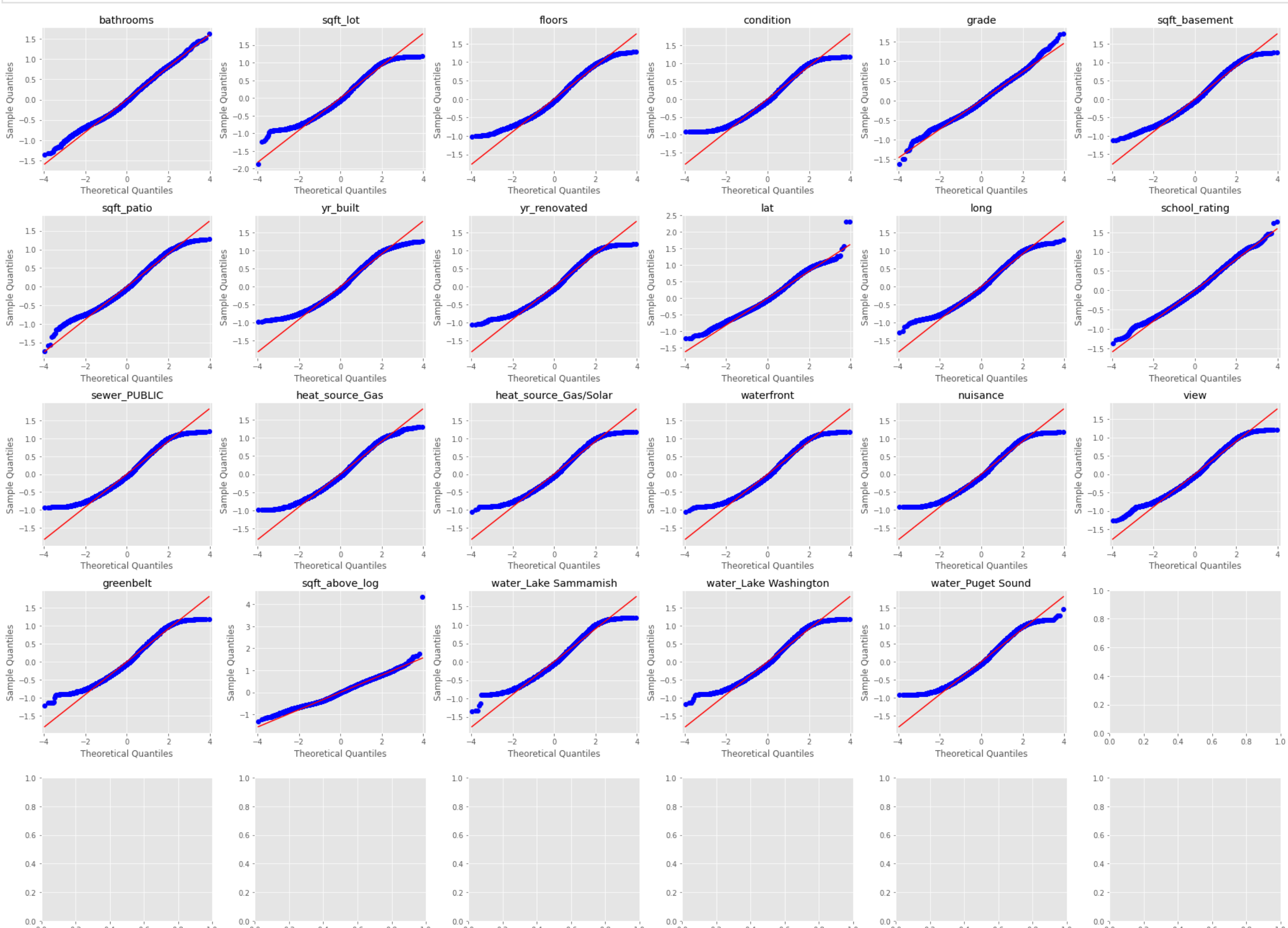
Jarque Beras score and outliers of the dataset should be further explored. The use of 3 standard deviations from the mean being the metric for outliers could be expanded slightly as it appears this was still affected in a major way.

Any independent variables that presented with a Variance Inflation factor above 5 should be looked at again to see if multicollinearity is an issue with these particular variables.

Getting final model error metrics

In [101…

```
get_model_qqplots(water_data, y_log)
```



In [102…

```
len(water_data)
```

Out[102… 27446