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 [293]:

```
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 [294]:
```
# 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 statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.outliers_influence import variance_inflation_factor


# model validation
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_absolute_error, mean_squared_error

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



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

Define Functions

In [295]:

```
# 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}, ignor

    # 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 f



#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=4, 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()
```

Read in dataset, check length

In [296]:

```
cd data
```

```
[WinError 2] The system cannot find the file specified: 'data'
C:\Users\alevi\Documents\Flatiron\dsc-data-science-env-config\Course_Fold
er\Phase_2\Housing_Linear_Model_Project\data
```

In [297]:

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

Out[297]:

```
30155
```

Dataset timeline

In [298]:

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

Out[298]:

```
(1900, 2022)
```

Checking dtypes

In [299]:
```
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?
- Is it normal?
  - histogram
  - QQ-plot
- Is it homoscedastic?

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 [300]:
```
df.dropna(inplace=True)
```

Recheck length

In [301]:
```
len(df)
```
Out[301]:
```
30111
```

Looking at Washington state

In [302]:
```
df['address'] = df['address'].str.lower()
```

In [303]:
```
df = df[df['address'].str.contains('washington')]
```

In [304]:
```
len(df)
```
Out[304]:
```
29208
```

Grabbing Zipcodes

In [305]:
```
df['zipcode'] = df['address'].apply(lambda x: x.split(',')[2].split(' ')[-1
```

In [306]:
```
df['zipcode'] = df['zipcode'].astype(str)
```

In [307]:
```
df['zipcode'].unique()
```

Out[307]:
```
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 [308]:
```
duwamish = ['98168']
elliot_bay_zips= ['98119','98104','98129','98132','98127','98125','98195','
puget_sound = ['98071','98083','98013','98070','98031','98131','98063','981
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=='9816
                                    else 'Elliot Bay' if x in elliot_
                                    else 'Puget Sound' if x in puget_
                                    else 'Lake Union' if x in lake_ur
                                    else 'ship canal' if x in ship_ca
                                    else 'Lake Washington' if x in la
                                    else 'Lake Sammamish' if x in lak
                                    else 'other')
```

In [309]:
```
df['waterfront_loc'].value_counts()
```

Out[309]:
```
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 [310]:
```
df = df[df['zipcode'].str.startswith('98') | df['zipcode'].str.contains('se
```

One Hot Encoding Waterfronts

In [311]:
```
waterfront_dummies = pd.get_dummies(df['waterfront_loc'], prefix='water', c
```

In [312]:
```
waterfront_dummies
```

Out[312]:

	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 [313]:
```
len(df)
```

Out[313]:
```
29200
```

In [314]:
```
len(df) == len(waterfront_dummies)
```
Out[314]:
```
True
```

In [315]:
```
df = pd.concat([df,waterfront_dummies], axis=1)
```

replacing seattle with seattle zipcode

In [316]:
```
df['zipcode'] = df['zipcode'].apply(lambda x: '98101' if x=='seattle' else
```

recheck zipcodes

In [317]:
```
df['zipcode'].unique()
```
Out[317]:
```
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 [318]:
```
len(df['zipcode'].unique())
```
Out[318]:
```
89
```

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

In [319]:
```
df.corr()['price'].abs().sort_values(ascending=False)
```

Out[319]:
```
price                    1.000000
sqft_living              0.616741
sqft_above               0.546108
bathrooms                0.488039
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
```

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.

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

In [320]:

```
#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 [321]:
```
df.dtypes
```

Out[321]:
```
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
month                             int64
day_of_year                       int64
dtype: object
```

Extracting Numerical Predictors by filtering dtypes

In [322]:
```
df.dtypes.unique()
```

Out[322]:
```
array([dtype('int64'), dtype('<M8[ns]'), dtype('float64'), dtype('O'),
       dtype('uint8')], dtype=object)
```

In [323]:
```
# categorizing dtypes
numerical_types = ['int64','float64']
numerical_predictors = list(df.select_dtypes(include=numerical_types))
numerical_predictors
```

Out[323]:
```
['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',
 'month',
 'day_of_year']
```

Create dataframe of numerical values

In [324]:
```
# df[numerical_predictors] selects only numerical columns
df_numerical = df[numerical_predictors]
```

In [325]:
```
df_numerical.columns
```

Out[325]:
```
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', 'l
ong',
       'month', 'day_of_year'],
      dtype='object')
```

In [326]:
```
len(df_numerical)
```

Out[326]:
```
29200
```

In [327]:
```
len(waterfront_dummies)
```

Out[327]:
```
29200
```

Dropping price to isolate predictors

In [328]:
```
df_numerical = df_numerical.drop(['id','price'],axis=1)
```

In [329]:
```
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

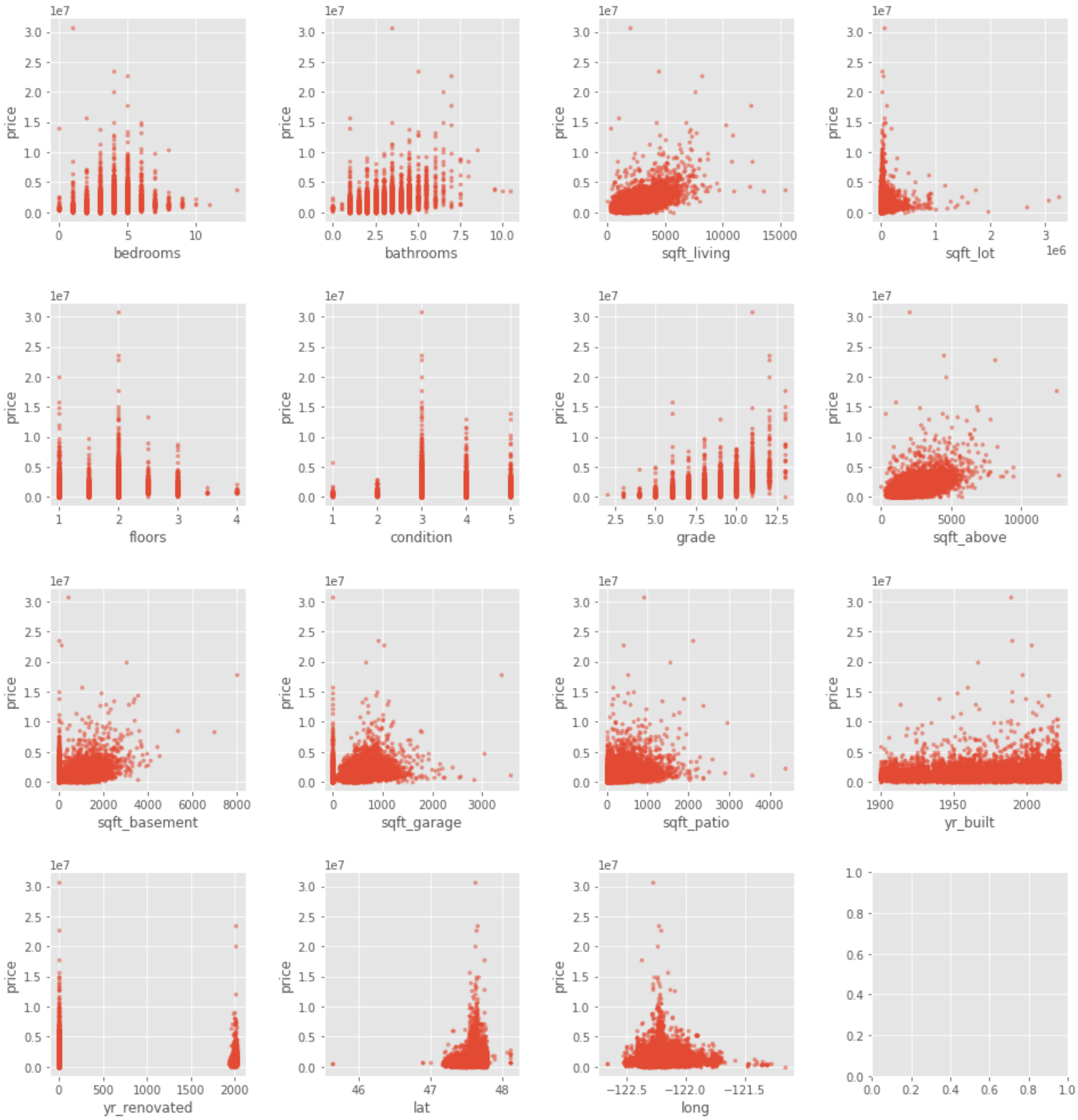
In [330]:
```
get_vifs(df_numerical)
```

	Variable	VIF
0	bedrooms	24.749584
1	bathrooms	26.262056
2	sqft_living	119.807162
3	sqft_lot	1.140572
4	floors	17.165769
5	condition	31.148954
6	grade	131.929086
7	sqft_above	92.872396
8	sqft_basement	7.075224
9	sqft_garage	4.672471
10	sqft_patio	2.240387
11	yr_built	9220.748738
12	yr_renovated	1.210932
13	lat	110243.802192
14	long	123455.262112
15	month	697.120476
16	day_of_year	612.128232

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 [331]:

```
# Specify the dependent variable and independent variables
y_col = 'price'
x_cols = [col for col in df_numerical.columns if col != y_col][:15]  # Use

# 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()
```


price
bedrooms
bathrooms
sqft_living
sqft_lot
floors
condition
grade
sqft_above
sqft_basement
sqft_garage
sqft_patio
yr_built
yr_renovated
lat
long

Extracting Categorical String Predictors

In [332]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 29200 entries, 0 to 30154
Data columns (total 35 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  month                  29200 non-null  int64
 34  day_of_year            29200 non-null  int64
dtypes: datetime64[ns](1), float64(5), int64(14), object(9), uint8(6)
memory usage: 6.9+ MB
```

In [333]:
```
categorical_types = ['O']
categorical_predictors = list(df.select_dtypes(include=categorical_types))
categorical_predictors
```

Out[333]:
```
['waterfront',
 'greenbelt',
 'nuisance',
 'view',
 'heat_source',
 'sewer_system',
 'address',
 'zipcode',
 'waterfront_loc']
```

In [334]:
```
df_categorical = df[categorical_predictors]
```

In [335]:
```
df_categorical
```

Out[335]:

	waterfront	greenbelt	nuisance	view	heat_source	sewer_system	address
0	NO	NO	NO	NONE	Gas	PUBLIC	2102 southeast 21st court, renton, washington ...
1	NO	NO	YES	AVERAGE	Oil	PUBLIC	11231 greenwood avenue north, seattle, washing...
2	NO	NO	NO	AVERAGE	Gas	PUBLIC	8504 south 113th street, seattle, washington 9...
3	NO	NO	NO	AVERAGE	Gas	PUBLIC	4079 letitia avenue south, seattle, washington...
4	NO	NO	YES	NONE	Electricity	PUBLIC	2193 northwest talus drive, issaquah, washingt...
...	...	...	...	...	...	...	...
30150	NO	NO	NO	NONE	Oil	PUBLIC	4673 eastern avenue north, seattle, washington...
30151	NO	NO	NO	FAIR	Gas	PUBLIC	4131 44th avenue southwest, seattle, washingto...
30152	NO	NO	YES	NONE	Gas	PUBLIC	910 martin luther king jr way, seattle, washin...
30153	NO	NO	NO	NONE	Gas	PUBLIC	17127 114th avenue southeast, renton, washingt...
30154	NO	NO	NO	NONE	Oil	PUBLIC	18615 7th avenue south, burien, washington 981...

29200 rows × 9 columns

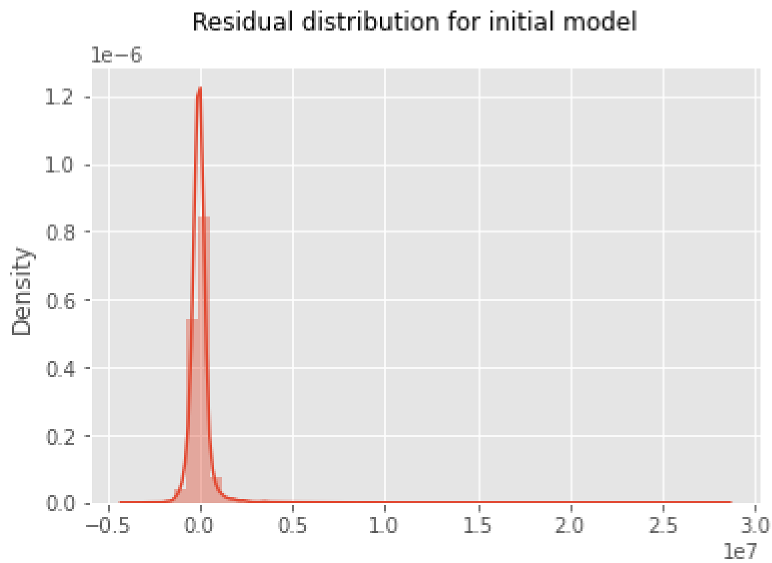
Model #1

In [336]:
```
model_data = df_numerical
```

In [337]:
```
get_OLS_model('initial',X = model_data, y = df['price'])
```

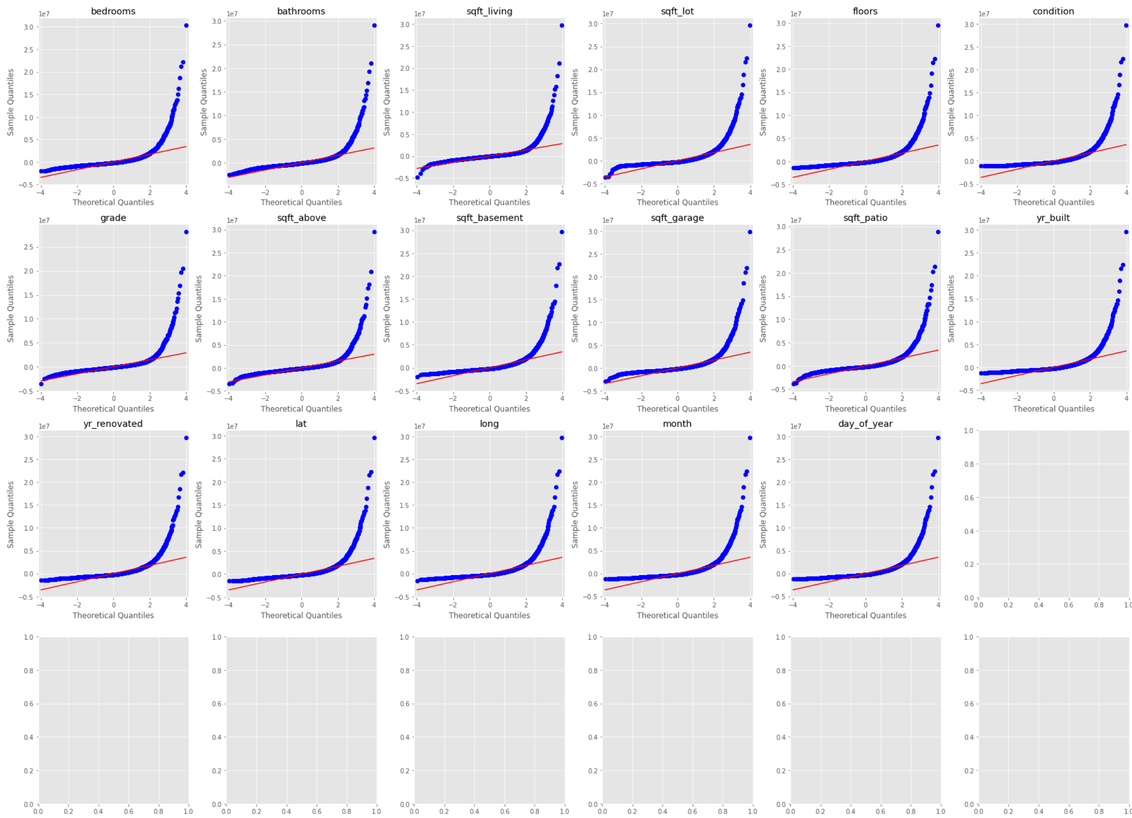
```
                            OLS Regression Results
==========================================================================
=====
Dep. Variable:                  price   R-squared:
0.514
Model:                            OLS   Adj. R-squared:
0.514
Method:                 Least Squares   F-statistic:
1814.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:30:28   Log-Likelihood:                -4.310
9e+05
No. Observations:               29200   AIC:                             8.62
2e+05
Df Residuals:                   29182   BIC:                             8.62
4e+05
Df Model:                          17
Covariance Type:            nonrobust
==========================================================================
========
                   coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------
--------
const         -6.16e+07      4e+06    -15.396      0.000   -6.94e+07    -
5.38e+07
bedrooms     -1.136e+05   5091.194    -22.322      0.000   -1.24e+05    -
1.04e+05
bathrooms     9.389e+04   7527.079     12.474      0.000    7.91e+04
1.09e+05
sqft_living    207.5950     17.071     12.161      0.000     174.135
241.055
sqft_lot         0.2667      0.063      4.265      0.000       0.144
0.389
floors       -1.476e+05   9568.312    -15.421      0.000   -1.66e+05    -
1.29e+05
condition     5.315e+04   5778.105      9.198      0.000    4.18e+04
6.45e+04
grade         2.149e+05   5521.008     38.916      0.000    2.04e+05
2.26e+05
sqft_above     270.4146     17.425     15.519      0.000     236.262
304.568
sqft_basement   80.8679     12.893      6.272      0.000      55.596
106.140
sqft_garage   -164.9199     18.061     -9.131      0.000    -200.320
-129.520
sqft_patio     193.5427     16.684     11.600      0.000     160.841
226.244
yr_built     -2899.2445    190.203    -15.243      0.000   -3272.051    -
2526.438
yr_renovated    68.9239      9.331      7.386      0.000      50.634
87.214
lat           1.344e+06   2.68e+04     50.165      0.000    1.29e+06
1.4e+06
long         -1.822e+04   3.04e+04     -0.599      0.549   -7.78e+04
4.13e+04
```

```
month            1.957e+04   1.28e+04      1.529      0.126   -5515.883
4.47e+04
day_of_year    -1215.8907    420.005     -2.895      0.004   -2039.120
-392.662
==========================================================================
=====
Omnibus:                    46855.092   Durbin-Watson:
1.915
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         9155904
1.827
Skew:                          10.060   Prob(JB):
0.00
Kurtosis:                     276.586   Cond. No.                     6.9
2e+07
==========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.92e+07. This might indicate that the
re are
strong multicollinearity or other numerical problems.
```



Out[337]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for initial model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [338]:
```
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
- reduce homoscedacity - reduce value of VIFs
- increase rsquared to promote higher level explanation of data from model
```

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 [339]:
```
df_categorical.columns
```

Out[339]:
```
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 [340]:
```
df['waterfront'].unique()
```
Out[340]:
```
array(['NO', 'YES'], dtype=object)
```

In [341]:
```
# 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=T
```

In [342]:
```
plt.scatter(x=df['waterfront'], y=df['price'])
```
Out[342]:
```
<matplotlib.collections.PathCollection at 0x19ff0695130>
```

In [343]:
```
df['nuisance'].unique()
```
Out[343]:
```
array(['NO', 'YES'], dtype=object)
```

In [344]:
```
# 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 [345]:
```
plt.scatter(x=df['nuisance'], y=df['price'])
```

Out[345]: <matplotlib.collections.PathCollection at 0x19f897a2490>

In [346]:
```
# 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=Tru
```

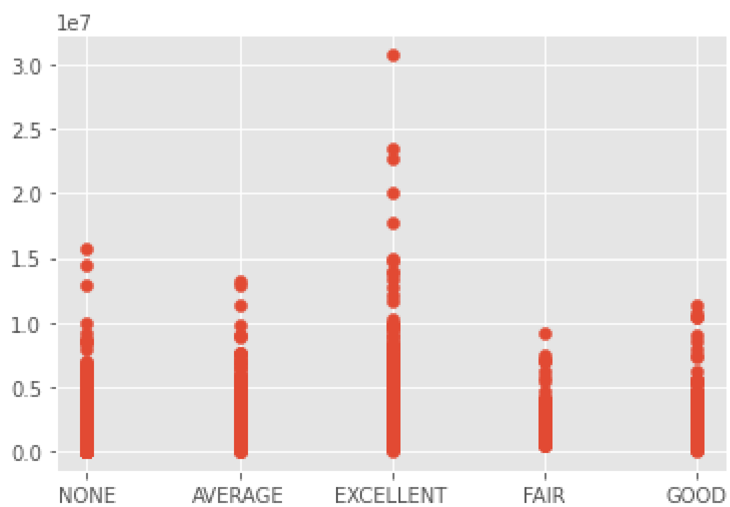
In [347]:
```
df['view'].unique()
```

Out[347]: array(['NONE', 'AVERAGE', 'EXCELLENT', 'FAIR', 'GOOD'], dtype=object)

In [348]:
```
# 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 [349]:
```
plt.scatter(x=df['view'], y=df['price'])
```

Out[349]: <matplotlib.collections.PathCollection at 0x19f89a918b0>



In [350]:
```
df['heat_source'].unique()
```

Out[350]:
```
array(['Gas', 'Oil', 'Electricity', 'Gas/Solar', 'Electricity/Solar',
       'Other', 'Oil/Solar'], dtype=object)
```

In [351]:
```
heat_source_dummies = pd.get_dummies(df['heat_source'], prefix='heat_source
heat_source_dummies
```

Out[351]:

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

29200 rows × 6 columns

In [352]:
```
df['sewer_system'].unique()
```

Out[352]:
```
array(['PUBLIC', 'PRIVATE', 'PRIVATE RESTRICTED', 'PUBLIC RESTRICTED'],
      dtype=object)
```

In [353]:
```
sewer_dummies = pd.get_dummies(df['sewer_system'],prefix='sewer', drop_firs
sewer_dummies
```

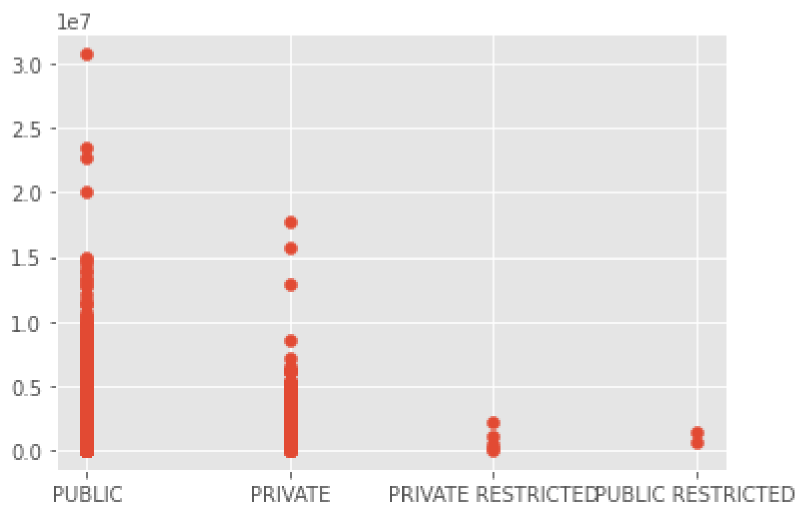
Out[353]:

	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
...	...	...	...
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 [354]:
```
plt.scatter(x=df['sewer_system'], y=df['price'])
```

Out[354]: <matplotlib.collections.PathCollection at 0x19f89be6ac0>



Developing categorical dataframe

In [355]:
```
df_cat_pick = df_categorical[['waterfront','nuisance','view','greenbelt']]
```

Model #2

In [356]:
```
model_2_data = pd.concat([df_numerical,sewer_dummies,heat_source_dummies, c
```

In [357]:
```
len(model_2_data) == len(waterfront_dummies)
```

Out[357]:
```
True
```

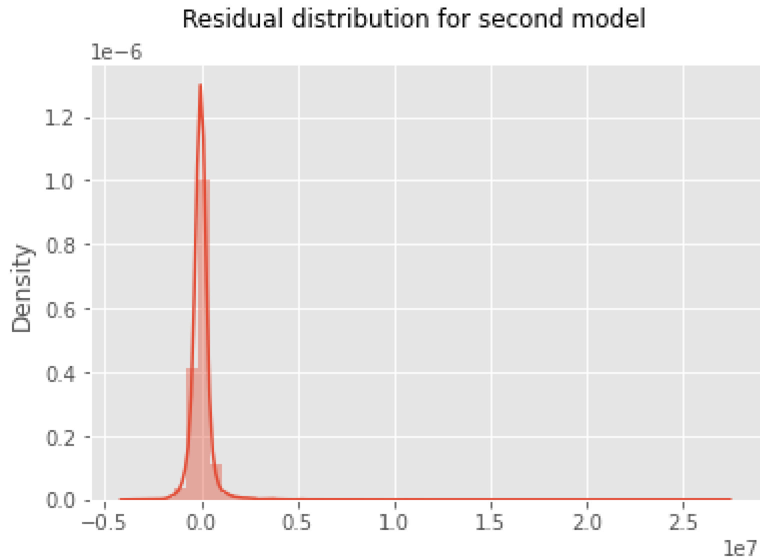
In [358]:
```
model_2_data.columns
```

Out[358]:
```
Index(['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',
       'condition', 'grade', 'sqft_above', 'sqft_basement', 'sqft_garag
e',
       'sqft_patio', 'yr_built', 'yr_renovated', 'lat', 'long', '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', 'nuisa
nce',
       'view', 'greenbelt'],
      dtype='object')
```

In [359]:
```
get_OLS_model('second',model_2_data, df['price'])
```

```
                            OLS Regression Results
==============================================================================
=====
Dep. Variable:                  price   R-squared:
0.555
Model:                            OLS   Adj. R-squared:
0.554
Method:                 Least Squares   F-statistic:
1211.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:30:50   Log-Likelihood:            -4.298
1e+05
No. Observations:               29200   AIC:                         8.59
7e+05
Df Residuals:                   29169   BIC:                         8.59
9e+05
Df Model:                          30
Covariance Type:            nonrobust
==============================================================================
========================
                                   coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
------------------------
const                        -5.628e+07   4.01e+06    -14.019      0.000
-6.42e+07   -4.84e+07
bedrooms                     -8.788e+04   4935.110    -17.808      0.000
-9.76e+04   -7.82e+04
bathrooms                     8.013e+04   7265.462     11.030      0.000
6.59e+04    9.44e+04
sqft_living                    160.8497     16.429      9.791      0.000
128.649     193.050
sqft_lot                         0.3726      0.063      5.929      0.000
0.249       0.496
floors                       -1.604e+05   9224.844    -17.391      0.000
-1.79e+05   -1.42e+05
condition                        5.8e+04   5597.956     10.361      0.000
4.7e+04     6.9e+04
grade                         1.979e+05   5348.962     37.001      0.000
1.87e+05    2.08e+05
sqft_above                     295.4203     16.781     17.604      0.000
262.529     328.312
sqft_basement                   70.3779     12.464      5.647      0.000
45.948      94.807
sqft_garage                   -103.2025     17.501     -5.897      0.000
-137.504     -68.901
sqft_patio                     129.3402     16.331      7.920      0.000
97.331     161.350
yr_built                     -2419.1189    185.808    -13.019      0.000
-2783.310   -2054.927
yr_renovated                    37.1529      9.001      4.128      0.000
19.511      54.795
lat                           1.416e+06    2.6e+04     54.431      0.000
1.37e+06    1.47e+06
long                          6.158e+04   3.04e+04      2.027      0.043
2024.286    1.21e+05
```

```
month                            2.198e+04   1.23e+04      1.793      0.073
-2046.961     4.6e+04
day_of_year                     -1309.3687    402.226     -3.255      0.001
-2097.751    -520.987
sewer_PRIVATE RESTRICTED         -5.43e+04   2.68e+05     -0.203      0.839
-5.8e+05    4.71e+05
sewer_PUBLIC                     1.703e+05   1.16e+04     14.642      0.000
1.48e+05    1.93e+05
sewer_PUBLIC RESTRICTED         -6.324e+04   4.23e+05     -0.149      0.881
-8.92e+05    7.66e+05
heat_source_Electricity/Solar -3.918e+04   7.97e+04     -0.492      0.623
-1.95e+05    1.17e+05
heat_source_Gas                  -515.2456   9507.344     -0.054      0.957
-1.92e+04    1.81e+04
heat_source_Gas/Solar            1.191e+05   6.27e+04      1.900      0.057
-3762.578    2.42e+05
heat_source_Oil                 -3.863e+04   1.45e+04     -2.656      0.008
-6.71e+04   -1.01e+04
heat_source_Oil/Solar           -1.541e+05   2.99e+05     -0.515      0.607
-7.4e+05    4.32e+05
heat_source_Other               -1.646e+04   1.34e+05     -0.123      0.902
-2.8e+05    2.47e+05
waterfront                       1.063e+06      3e+04     35.457      0.000
1e+06    1.12e+06
nuisance                         1.347e+04   9518.997      1.415      0.157
-5188.619    3.21e+04
view                             8.936e+04   4854.779     18.408      0.000
7.98e+04    9.89e+04
greenbelt                        7463.3683   2.24e+04      0.334      0.738
-3.63e+04    5.13e+04
==========================================================================
=====
Omnibus:                     45511.144   Durbin-Watson:
1.904
Prob(Omnibus):                   0.000   Jarque-Bera (JB):         8590179
7.035
Skew:                            9.450   Prob(JB):
0.00
Kurtosis:                      268.042   Cond. No.                     7.2
5e+07
==========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 7.25e+07. This might indicate that the
re are
strong multicollinearity or other numerical problems.
```



Out[359]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for second model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

`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

- Jarque-Bera score is sky high and must come down for the model to hold any validity.
- 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 [360]:
```
outlier_thresh = df['price'].std()*3 # value of the prices at the third sta
df_outlier_removed = df.loc[abs(df['price']) <= outlier_thresh] # slicing d

# assign y as the target variable
y = df_outlier_removed['price']
```

In [361]:
```
model_2_data_outlier_removed = model_2_data.loc[abs(df['price']) <= outlier
```

In [362]:
```
df_outlier_removed
```

Out[362]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	wa
0	7399300360	2022-05-24	675000.0	4	1.0	1180	7140	1.0	
1	8910500230	2021-12-13	920000.0	5	2.5	2770	6703	1.0	
2	1180000275	2021-09-29	311000.0	6	2.0	2880	6156	1.0	
3	1604601802	2021-12-14	775000.0	3	3.0	2160	1400	2.0	
4	8562780790	2021-08-24	592500.0	2	2.0	1120	758	2.0	
...	...	...	...	...	...	...	...	...	
30150	7834800180	2021-11-30	1555000.0	5	2.0	1910	4000	1.5	
30151	194000695	2021-06-16	1313000.0	3	2.0	2020	5800	2.0	
30152	7960100080	2022-05-27	800000.0	3	2.0	1620	3600	1.0	
30153	2781280080	2022-02-24	775000.0	3	2.5	2570	2889	2.0	
30154	9557800100	2022-04-29	500000.0	3	1.5	1200	11058	1.0	

28004 rows × 35 columns

In [363]:
```
waterfront_dummies = df_outlier_removed[['water_Elliot Bay','water_Lake Sar
```

In [364]:
```
df_outlier_removed.columns
```

Out[364]:
```
Index(['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'greenbelt', 'nuisance', 'vie
w',
       'condition', 'grade', 'heat_source', 'sewer_system', 'sqft_above',
       'sqft_basement', 'sqft_garage', 'sqft_patio', 'yr_built',
       'yr_renovated', 'address', 'lat', 'long', 'zipcode', 'waterfront_l
oc',
       'water_Elliot Bay', 'water_Lake Sammamish', 'water_Lake Union',
       'water_Lake Washington', 'water_Puget Sound', 'water_other', 'mont
h',
       'day_of_year'],
      dtype='object')
```

New look at model with removed outliers

In [365]:
```
outlier_data = pd.concat([y,model_2_data_outlier_removed], axis=1)
```

In [366]:
```
outlier_data = outlier_data.drop('price', axis=1)
```

In [367]:
```
len(outlier_data)
```

Out[367]: 28004

In [368]:
```
outlier_data
```

Out[368]:

	bedrooms	bathrooms	sqft_living	sqft_lot	floors	condition	grade	sqft_above	sqft
0	4	1.0	1180	7140	1.0	4	7	1180	
1	5	2.5	2770	6703	1.0	3	7	1570	
2	6	2.0	2880	6156	1.0	3	7	1580	
3	3	3.0	2160	1400	2.0	3	9	1090	
4	2	2.0	1120	758	2.0	3	7	1120	
...	...	...	...	...	...	...	...	...	
30150	5	2.0	1910	4000	1.5	4	8	1600	
30151	3	2.0	2020	5800	2.0	3	7	2020	
30152	3	2.0	1620	3600	1.0	3	7	940	
30153	3	2.5	2570	2889	2.0	3	8	1830	
30154	3	1.5	1200	11058	1.0	3	7	1200	

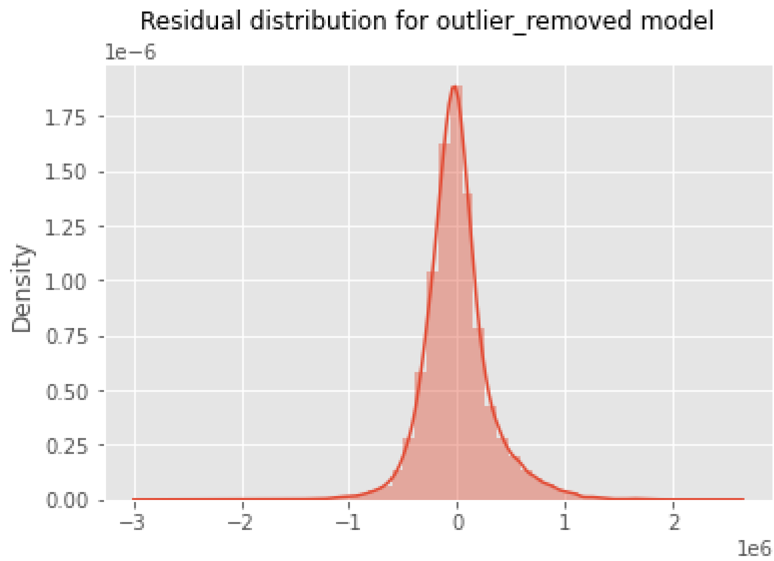
28004 rows × 30 columns

Model #3

In [369]:
```
get_OLS_model('outlier_removed', outlier_data,y)
```

```
                            OLS Regression Results
==============================================================================
=====
Dep. Variable:                  price   R-squared:
0.622
Model:                            OLS   Adj. R-squared:
0.622
Method:                 Least Squares   F-statistic:
1534.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:30:59   Log-Likelihood:            -3.934
7e+05
No. Observations:               28004   AIC:                         7.87
0e+05
Df Residuals:                   27973   BIC:                         7.87
3e+05
Df Model:                          30
Covariance Type:            nonrobust
==============================================================================
========================
                            coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
------------------------
const                -2.891e+07    2.1e+06    -13.763      0.000
-3.3e+07   -2.48e+07
bedrooms             -1.261e+04   2647.400     -4.764      0.000
-1.78e+04   -7423.552
bathrooms             3.438e+04   3907.140      8.800      0.000
2.67e+04    4.2e+04
sqft_living            137.7492      8.974     15.350      0.000
120.160     155.338
sqft_lot                 0.3540      0.036      9.708      0.000
0.283       0.426
floors               -2.695e+04   4927.866     -5.468      0.000
-3.66e+04   -1.73e+04
condition              5.94e+04   2922.086     20.326      0.000
5.37e+04    6.51e+04
grade                 1.469e+05   2888.537     50.859      0.000
1.41e+05    1.53e+05
sqft_above              98.6873      9.222     10.701      0.000
80.611     116.763
sqft_basement            9.0654      6.729      1.347      0.178
-4.123      22.254
sqft_garage            -14.9218      9.345     -1.597      0.110
-33.238       3.394
sqft_patio              52.5853      8.886      5.918      0.000
35.168      70.002
yr_built             -2128.0746     98.282    -21.653      0.000
-2320.712   -1935.437
yr_renovated            29.5335      4.828      6.117      0.000
20.070      38.997
lat                   1.305e+06   1.35e+04     96.806      0.000
1.28e+06    1.33e+06
long                  2.434e+05   1.59e+04     15.356      0.000
2.12e+05    2.74e+05
```

```
month                             1.988e+04   6406.701      3.104      0.002
7326.324    3.24e+04
day_of_year                      -1128.1162    210.292     -5.365      0.000
-1540.298    -715.934
sewer_PRIVATE RESTRICTED          1.742e+05   1.37e+05      1.268      0.205
-9.5e+04    4.43e+05
sewer_PUBLIC                      5.498e+04   6113.519      8.993      0.000
4.3e+04     6.7e+04
sewer_PUBLIC RESTRICTED          -2.283e+04   2.17e+05     -0.105      0.916
-4.48e+05    4.02e+05
heat_source_Electricity/Solar -3.437e+04   4.12e+04     -0.834      0.404
-1.15e+05    4.64e+04
heat_source_Gas                   3.284e+04   4947.829      6.638      0.000
2.31e+04    4.25e+04
heat_source_Gas/Solar             1.564e+05   3.38e+04      4.634      0.000
9.03e+04    2.23e+05
heat_source_Oil                  -1.553e+04   7536.189     -2.060      0.039
-3.03e+04    -756.860
heat_source_Oil/Solar            -4.439e+04   1.53e+05     -0.290      0.772
-3.45e+05    2.56e+05
heat_source_Other                 9.011e+04   7.06e+04      1.276      0.202
-4.83e+04    2.29e+05
waterfront                        1.227e+05   1.82e+04      6.751      0.000
8.71e+04    1.58e+05
nuisance                         -2.687e+04   5007.274     -5.366      0.000
-3.67e+04   -1.71e+04
view                              6.205e+04   2654.342     23.375      0.000
5.68e+04    6.72e+04
greenbelt                         9.809e+04   1.19e+04      8.255      0.000
7.48e+04    1.21e+05
==========================================================================
=====
Omnibus:                       3918.983   Durbin-Watson:
2.002
Prob(Omnibus):                    0.000   Jarque-Bera (JB):            2035
2.924
Skew:                             0.577   Prob(JB):
0.00
Kurtosis:                         7.014   Cond. No.                     6.6
0e+07
==========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.6e+07. This might indicate that ther
e are
strong multicollinearity or other numerical problems.
```



Out[369]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for outlier_removed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [370]:
```
outlier_data.columns
```

Out[370]:
```
Index(['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',
       'condition', 'grade', 'sqft_above', 'sqft_basement', 'sqft_garag
e',
       'sqft_patio', 'yr_built', 'yr_renovated', 'lat', 'long', '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', 'nuisa
nce',
       '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`

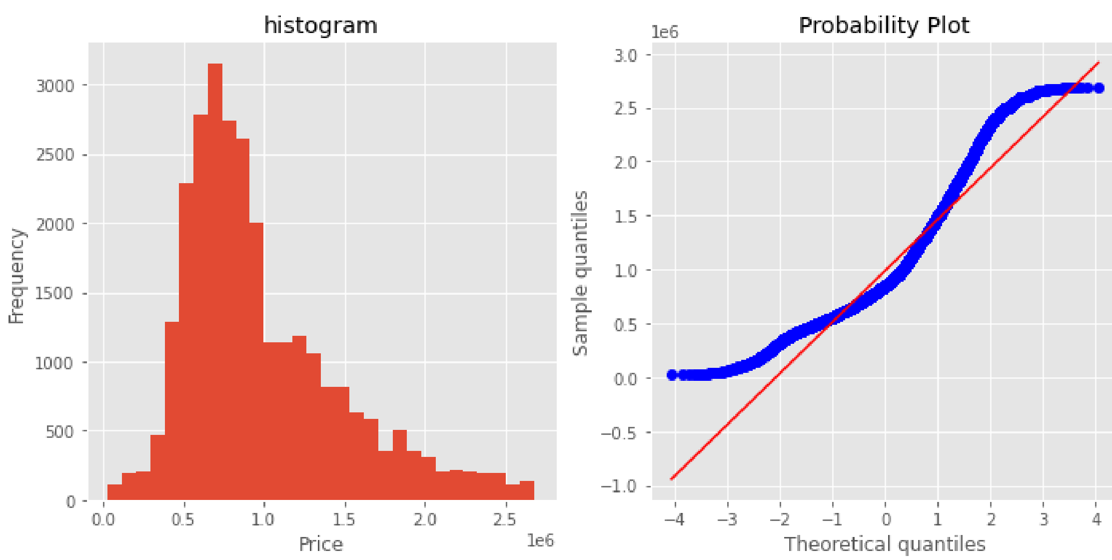
- Adjusted rsquared indicates that the model explains 62.2% of the data.
- Skewness has improved dramatically to an acceptable range between -2 and 2. The removal of outliers has made this possible.
- Durbin-Watson score is still in the acceptable ranges of 1.5-2.5

- 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 [371]:

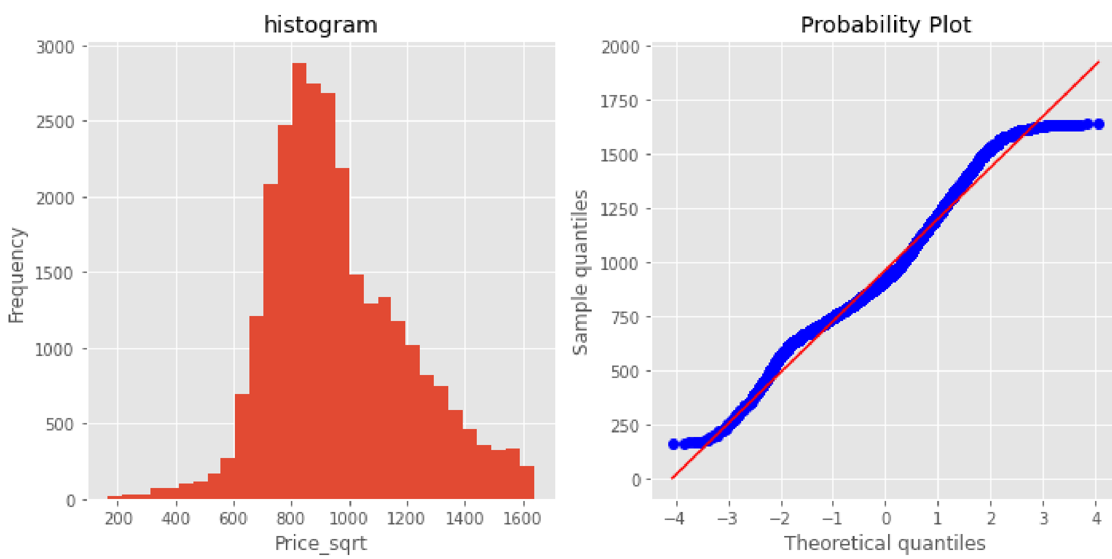
```
import scipy.stats as stats
fig, axs = plt.subplots(1, 2, figsize=(10, 5))

# Plot histogram on the first subplot
axs[0].hist(y, bins=30)
axs[0].set_xlabel('Price')
axs[0].set_ylabel('Frequency')
axs[0].set_title('histogram')
# Plot QQ-plot on the second subplot
stats.probplot(y, 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()
```



Issue above is the data shows linearization everywhere but both tails of the data. Catching the lower tail will be the goal for the next test of transformation. For this, we will try a root transformation.

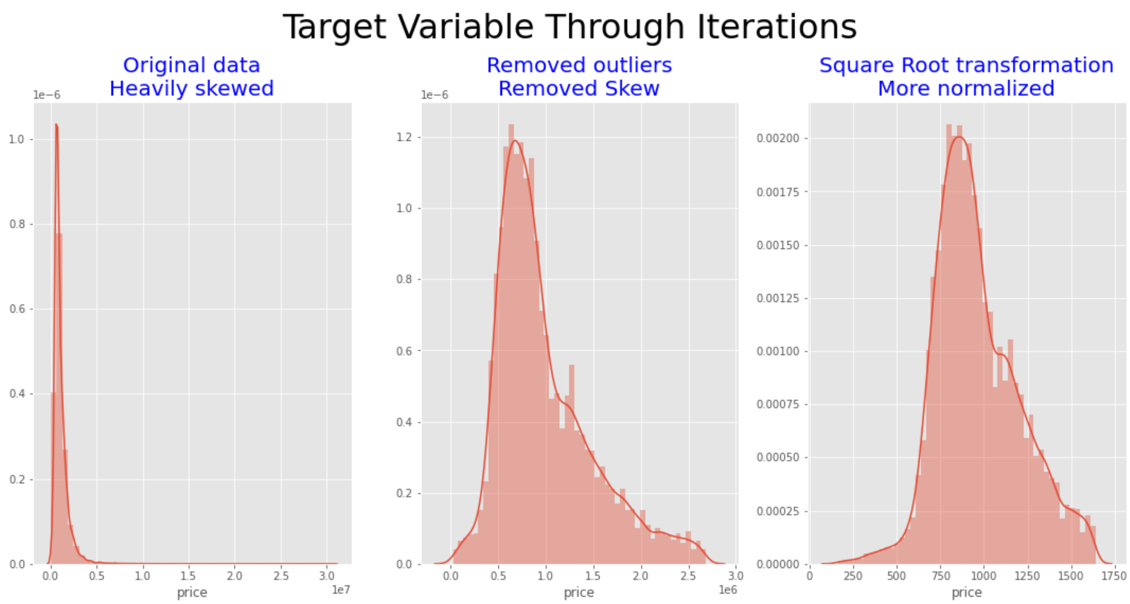
In [372]:

```
import matplotlib.pyplot as plt
import scipy.stats as stats

# Create subplots with 1 row and 2 columns
fig, axs = plt.subplots(1, 2, figsize=(10, 5))
y_sqrt = y**0.5
# Plot histogram on the first subplot
axs[0].hist(y_sqrt, bins=30)
axs[0].set_xlabel('Price_sqrt')
axs[0].set_ylabel('Frequency')
axs[0].set_title('histogram')

# Plot QQ-plot on the second subplot
stats.probplot(y_sqrt, 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()
```



In [373]:

```
fig, (ax1,ax2,ax3) = plt.subplots(1,3)

og = sns.distplot(df['price'],ax=ax1).set_title('Original data\nHeavily ske
ot = sns.distplot(y,ax=ax2).set_title('Removed outliers\nRemoved Skew',font
lo = sns.distplot(y_sqrt,ax=ax3).set_title('Square Root transformation\nMor

ax1.set_ylabel("")
ax2.set_ylabel("")
ax3.set_ylabel("")


plt.gcf().set_size_inches(15, 8)
plt.suptitle("Target Variable Through Iterations",fontsize=32)
fig.tight_layout()
plt.show()
```



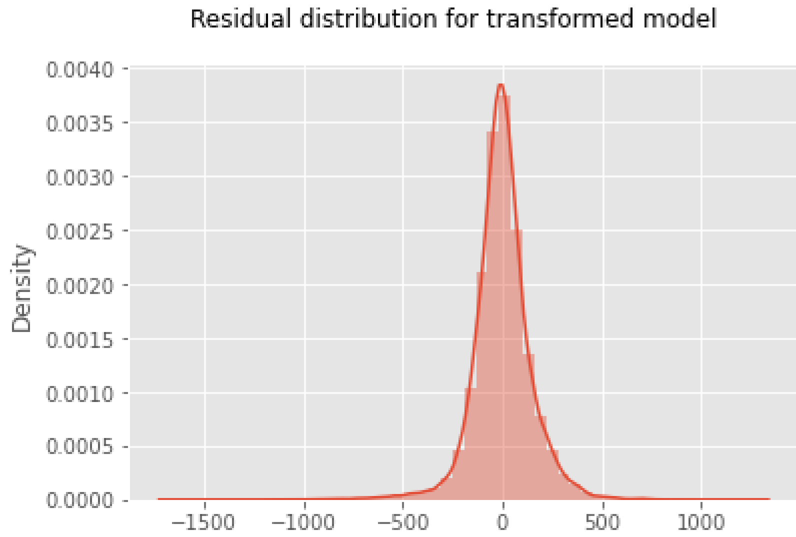
Checking model with transformed target variable - square root transformation

In [374]:

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

```
                            OLS Regression Results
==============================================================================
=====
Dep. Variable:                  price   R-squared:
0.628
Model:                            OLS   Adj. R-squared:
0.627
Method:                 Least Squares   F-statistic:
1571.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:31:08   Log-Likelihood:             -1.794
0e+05
No. Observations:               28004   AIC:                          3.58
9e+05
Df Residuals:                   27973   BIC:                          3.59
1e+05
Df Model:                          30
Covariance Type:            nonrobust
==============================================================================
========================
                                 coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
------------------------
const                      -1.45e+04   1005.553    -14.415      0.000
-1.65e+04   -1.25e+04
bedrooms                     -3.1025      1.267     -2.448      0.014
-5.587      -0.618
bathrooms                    19.6412      1.871     10.500      0.000
15.975      23.308
sqft_living                   0.0571      0.004     13.296      0.000
0.049       0.066
sqft_lot                      0.0002   1.75e-05     10.315      0.000
0.000       0.000
floors                       -6.1373      2.359     -2.601      0.009
-10.762      -1.513
condition                    31.1453      1.399     22.262      0.000
28.403      33.887
grade                        69.4559      1.383     50.223      0.000
66.745      72.167
sqft_above                    0.0462      0.004     10.458      0.000
0.038       0.055
sqft_basement                 0.0095      0.003      2.946      0.003
0.003       0.016
sqft_garage                  -0.0052      0.004     -1.158      0.247
-0.014       0.004
sqft_patio                    0.0268      0.004      6.293      0.000
0.018       0.035
yr_built                     -0.9368      0.047    -19.909      0.000
-1.029      -0.845
yr_renovated                  0.0138      0.002      5.970      0.000
0.009       0.018
lat                         669.3887      6.456    103.687      0.000
656.735     682.042
long                        125.8855      7.590     16.587      0.000
111.010     140.761
```

```
month                              10.4758      3.067      3.415      0.001
4.464      16.488
day_of_year                        -0.5743      0.101     -5.704      0.000
-0.772      -0.377
sewer_PRIVATE RESTRICTED          -12.8877     65.764     -0.196      0.845
-141.788     116.013
sewer_PUBLIC                       25.4531      2.927      8.696      0.000
19.716      31.190
sewer_PUBLIC RESTRICTED             2.5827    103.756      0.025      0.980
-200.783     205.949
heat_source_Electricity/Solar     -37.6734     19.718     -1.911      0.056
-76.322       0.975
heat_source_Gas                    19.1408      2.369      8.080      0.000
14.498      23.784
heat_source_Gas/Solar              63.9797     16.160      3.959      0.000
32.305      95.654
heat_source_Oil                    -1.4670      3.608     -0.407      0.684
-8.539       5.605
heat_source_Oil/Solar              -3.7398     73.373     -0.051      0.959
-147.554     140.074
heat_source_Other                  36.1906     33.808      1.070      0.284
-30.076     102.457
waterfront                         62.8417      8.704      7.220      0.000
45.781      79.902
nuisance                          -15.3349      2.397     -6.397      0.000
-20.034     -10.636
view                               28.3706      1.271     22.325      0.000
25.880      30.861
greenbelt                          45.6410      5.689      8.022      0.000
34.490      56.792
=========================================================================
=====
Omnibus:                     3917.403   Durbin-Watson:
2.008
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            3720
1.626
Skew:                          -0.361   Prob(JB):
0.00
Kurtosis:                       8.600   Cond. No.                     6.6
0e+07
=========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.6e+07. This might indicate that ther
e are
strong multicollinearity or other numerical problems.
```



Out[374]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

y_log vs y_sqrt

The model with the square root transformation appears to be less skewed and possesses a higher rsquared value, lending the ability of the model to explain more of the data. For these reasons we will use y_sqrt as our dependent variable for now until y_log appears to outweight the benefit of y_sqrt.

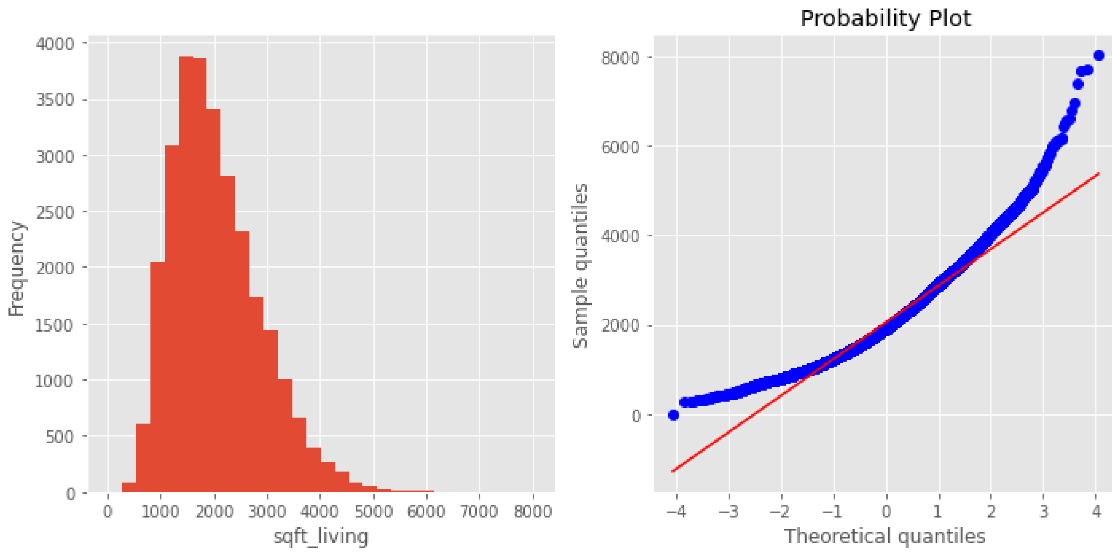
Jarque-Beras score is significantly better as well with the y_sqrt variable so I'll go with it for now.

In [375]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error

X = outlier_data
# Fit a linear regression model
reg = LinearRegression().fit(X, y_sqrt)

# Predict the target values
y_pred = reg.predict(X)

# Calculate the mean absolute error
rmse = mean_squared_error(y_sqrt, y_pred)

print("Root mean squared error: ", rmse)
```

```
Root mean squared error:  21478.232538371158
```

Checking distribution of predictor

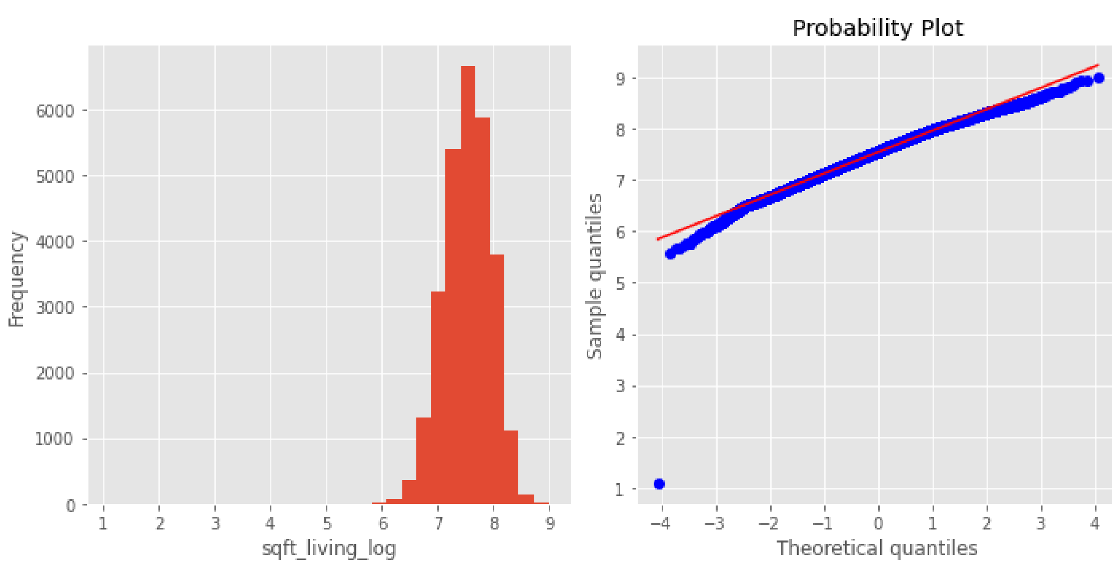
In [376]:
```
plot_hist_qq(outlier_data, 'sqft_living')
```



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

In [377]:
```
outlier_data['sqft_living_log'] = np.log(outlier_data['sqft_living'])
```

In [378]:
```
plot_hist_qq(outlier_data, 'sqft_living_log')
```



In [379]:
```
outlier_data = outlier_data.drop('sqft_living', axis=1)
```

In [380]:

```
outlier_data
```

Out[380]:

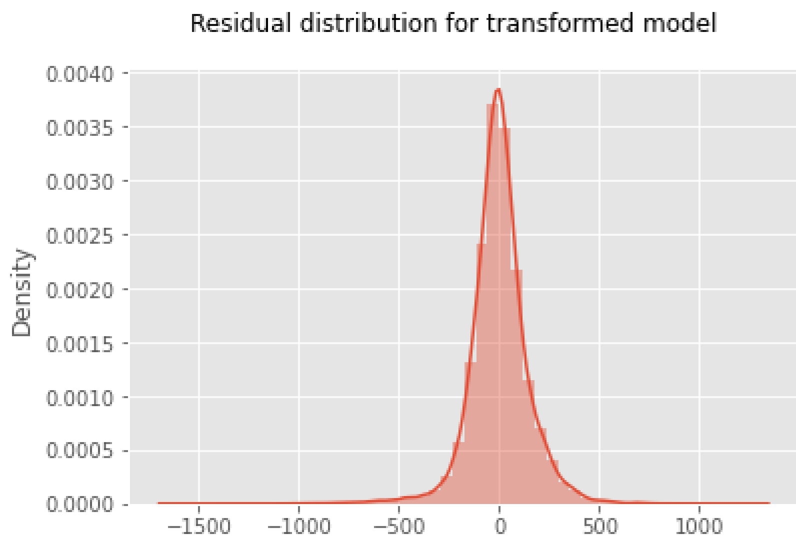
	bedrooms	bathrooms	sqft_lot	floors	condition	grade	sqft_above	sqft_basement
0	4	1.0	7140	1.0	4	7	1180	0
1	5	2.5	6703	1.0	3	7	1570	1570
2	6	2.0	6156	1.0	3	7	1580	1580
3	3	3.0	1400	2.0	3	9	1090	1070
4	2	2.0	758	2.0	3	7	1120	550
...	...	...	...	...	...	...	...	...
30150	5	2.0	4000	1.5	4	8	1600	1130
30151	3	2.0	5800	2.0	3	7	2020	0
30152	3	2.0	3600	1.0	3	7	940	920
30153	3	2.5	2889	2.0	3	8	1830	740
30154	3	1.5	11058	1.0	3	7	1200	0

28004 rows × 30 columns

In [381]:
```
get_OLS_model('transformed', outlier_data, y_sqrt)
```

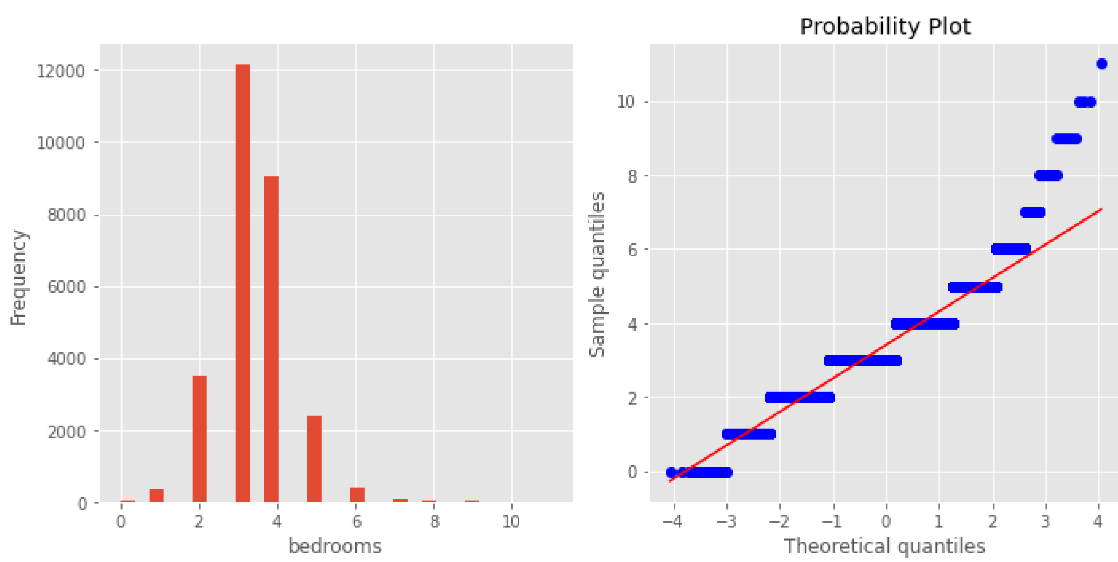
```
                            OLS Regression Results                            
==============================================================================
=====
Dep. Variable:                  price   R-squared:                       
0.626
Model:                            OLS   Adj. R-squared:                  
0.625
Method:                 Least Squares   F-statistic:                     
1558.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):              
0.00
Time:                        20:31:17   Log-Likelihood:            -1.794
8e+05
No. Observations:               28004   AIC:                         3.59
0e+05
Df Residuals:                   27973   BIC:                         3.59
3e+05
Df Model:                          30                                         
Covariance Type:            nonrobust                                         
==============================================================================
========================
                           coef    std err          t      P>|t|      
[0.025      0.975]
------------------------------------------------------------------------------
------------------------
const                -1.483e+04   1012.321    -14.651      0.000   
-1.68e+04   -1.28e+04
bedrooms                -1.9239      1.305     -1.474      0.141      
-4.482       0.634
bathrooms               23.4883      1.859     12.634      0.000      
19.844      27.132
sqft_lot                 0.0002   1.75e-05     10.510      0.000       
0.000       0.000
floors                 -10.7543      2.335     -4.605      0.000     
-15.331      -6.177
condition               32.3383      1.405     23.010      0.000      
29.584      35.093
grade                   70.5903      1.392     50.714      0.000      
67.862      73.319
sqft_above               0.0848      0.003     26.520      0.000       
0.078       0.091
sqft_basement            0.0324      0.003     11.901      0.000       
0.027       0.038
sqft_garage             -0.0133      0.004     -2.997      0.003      
-0.022      -0.005
sqft_patio               0.0299      0.004      7.026      0.000       
0.022       0.038
yr_built                -0.8908      0.047    -18.911      0.000      
-0.983      -0.798
yr_renovated             0.0144      0.002      6.188      0.000       
0.010       0.019
lat                    670.4926      6.476    103.542      0.000     
657.800     683.185
long                   126.1891      7.620     16.561      0.000     
111.254     141.124
month                   10.3472      3.075      3.364      0.001       
4.319      16.375
```

```
day_of_year                       -0.5697      0.101     -5.644      0.000
-0.768      -0.372
sewer_PRIVATE RESTRICTED          -1.5546     65.945     -0.024      0.981
-130.810     127.701
sewer_PUBLIC                      24.6168      2.934      8.390      0.000
18.866      30.368
sewer_PUBLIC RESTRICTED           -7.0466    104.027     -0.068      0.946
-210.945     196.852
heat_source_Electricity/Solar   -37.8148     19.770     -1.913      0.056
-76.566       0.936
heat_source_Gas                   18.2644      2.384      7.663      0.000
13.593      22.936
heat_source_Gas/Solar             65.0420     16.203      4.014      0.000
33.283      96.800
heat_source_Oil                   -4.6489      3.613     -1.287      0.198
-11.731       2.433
heat_source_Oil/Solar             -1.8443     73.570     -0.025      0.980
-146.046     142.357
heat_source_Other                 38.0187     33.898      1.122      0.262
-28.423     104.460
waterfront                        63.0894      8.727      7.229      0.000
45.984      80.195
nuisance                         -15.0994      2.404     -6.281      0.000
-19.811     -10.387
view                              29.2177      1.272     22.964      0.000
26.724      31.712
greenbelt                         47.5776      5.703      8.343      0.000
36.400      58.755
sqft_living_log                   33.6705      6.332      5.318      0.000
21.260      46.081
==========================================================================
=====
Omnibus:                     3862.249   Durbin-Watson:
2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           3674
8.345
Skew:                          -0.346   Prob(JB):
0.00
Kurtosis:                       8.569   Cond. No.                    6.6
3e+07
==========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.63e+07. This might indicate that the
re are
strong multicollinearity or other numerical problems.
```



Out[381]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [382]:
```
plot_hist_qq(outlier_data, 'bedrooms')
```



pval > 0.05

- `bedrooms` - will be dropped from the current model

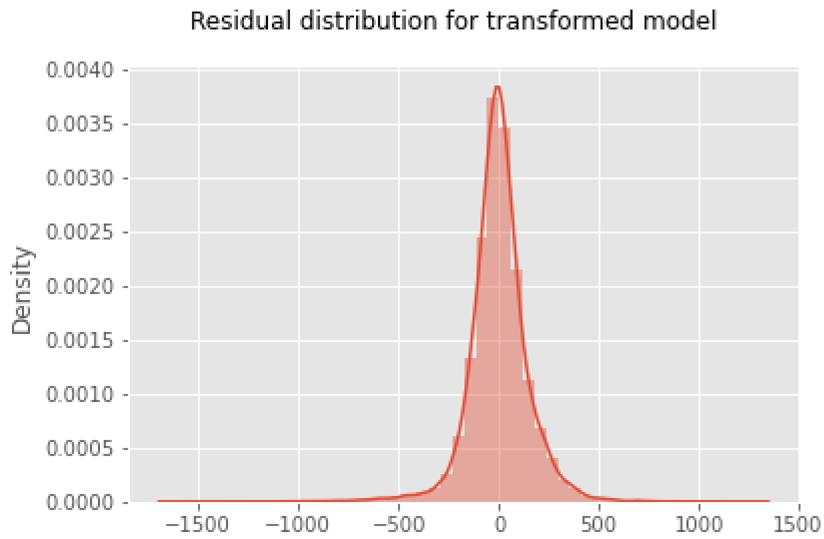
In [383]:
```
outlier_data = outlier_data.drop(['bedrooms'], axis=1)
```

Rerun model

In [384]:
```
get_OLS_model('transformed', outlier_data, y_sqrt)
```

```
                            OLS Regression Results
==============================================================================
=====
Dep. Variable:                  price   R-squared:
0.626
Model:                            OLS   Adj. R-squared:
0.625
Method:                 Least Squares   F-statistic:
1612.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:31:25   Log-Likelihood:                -1.794
8e+05
No. Observations:               28004   AIC:                             3.59
0e+05
Df Residuals:                   27974   BIC:                             3.59
3e+05
Df Model:                          29
Covariance Type:            nonrobust
==============================================================================
========================
                                   coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
------------------------
const                        -1.484e+04   1012.340    -14.654      0.000
-1.68e+04   -1.29e+04
bathrooms                       22.8849      1.814     12.619      0.000
19.330      26.439
sqft_lot                         0.0002   1.75e-05     10.583      0.000
0.000       0.000
floors                         -10.5323      2.330     -4.520      0.000
-15.100      -5.965
condition                       32.2949      1.405     22.984      0.000
29.541      35.049
grade                           70.8647      1.379     51.372      0.000
68.161      73.568
sqft_above                       0.0846      0.003     26.489      0.000
0.078       0.091
sqft_basement                    0.0325      0.003     11.922      0.000
0.027       0.038
sqft_garage                     -0.0132      0.004     -2.968      0.003
-0.022      -0.004
sqft_patio                       0.0302      0.004      7.113      0.000
0.022       0.039
yr_built                        -0.8860      0.047    -18.854      0.000
-0.978      -0.794
yr_renovated                     0.0145      0.002      6.250      0.000
0.010       0.019
lat                            671.0171      6.466    103.778      0.000
658.344     683.691
long                           126.3329      7.619     16.581      0.000
111.399     141.267
month                           10.3404      3.075      3.362      0.001
4.312      16.369
day_of_year                     -0.5697      0.101     -5.643      0.000
-0.768      -0.372
```

```
sewer_PRIVATE RESTRICTED         -1.6498     65.946     -0.025      0.980
-130.908     127.608
sewer_PUBLIC                     24.3608      2.929      8.317      0.000
18.620      30.102
sewer_PUBLIC RESTRICTED          -8.0389    104.027     -0.077      0.938
-211.938     195.860
heat_source_Electricity/Solar   -38.1259     19.770     -1.928      0.054
-76.876       0.624
heat_source_Gas                  18.2508      2.384      7.657      0.000
13.579      22.923
heat_source_Gas/Solar            65.2818     16.202      4.029      0.000
33.524      97.039
heat_source_Oil                  -4.7264      3.613     -1.308      0.191
-11.808       2.355
heat_source_Oil/Solar            -0.6240     73.567     -0.008      0.993
-144.819     143.571
heat_source_Other                38.3315     33.898      1.131      0.258
-28.110     104.773
waterfront                       63.5322      8.722      7.284      0.000
46.437      80.628
nuisance                        -15.1146      2.404     -6.287      0.000
-19.827     -10.403
view                             29.3734      1.268     23.166      0.000
26.888      31.859
greenbelt                        47.6660      5.702      8.359      0.000
36.489      58.843
sqft_living_log                  30.9219      6.051      5.110      0.000
19.062      42.782
=========================================================================
=====
Omnibus:                     3860.222   Durbin-Watson:
2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            3673
1.967
Skew:                          -0.346   Prob(JB):
0.00
Kurtosis:                       8.568   Cond. No.                     6.6
3e+07
=========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.63e+07. This might indicate that the
re are
strong multicollinearity or other numerical problems.
```



Out[384]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

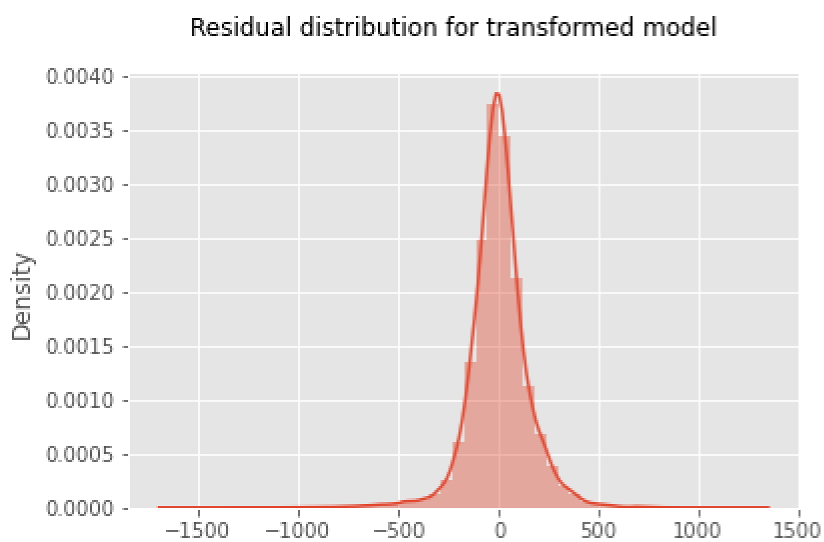
Dropping sewer/heat source data

In [385]:
```
new_outlier_data = outlier_data.drop(['sewer_PRIVATE RESTRICTED','sewer_PUE
```

In [386]:
```
get_OLS_model('transformed', new_outlier_data, y_sqrt)
```

```
                            OLS Regression Results
==========================================================================
=====
Dep. Variable:                  price   R-squared:
0.626
Model:                            OLS   Adj. R-squared:
0.625
Method:                 Least Squares   F-statistic:
2032.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):
0.00
Time:                        20:31:28   Log-Likelihood:           -1.794
8e+05
No. Observations:               28004   AIC:                        3.59
0e+05
Df Residuals:                   27980   BIC:                        3.59
2e+05
Df Model:                          23
Covariance Type:            nonrobust
==========================================================================
================
                            coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                 -1.482e+04   1012.120    -14.646      0.000   -1.68
e+04   -1.28e+04
bathrooms                23.1462      1.799     12.864      0.000      1
9.620      26.673
sqft_lot                  0.0002   1.75e-05     10.683      0.000
0.000       0.000
floors                  -10.4958      2.324     -4.516      0.000     -1
5.051      -5.940
condition                32.4584      1.393     23.299      0.000      2
9.728      35.189
grade                    70.8221      1.378     51.381      0.000      6
8.120      73.524
sqft_above                0.0844      0.003     26.477      0.000
0.078       0.091
sqft_basement             0.0322      0.003     11.866      0.000
0.027       0.038
sqft_garage              -0.0133      0.004     -3.004      0.003      -
0.022      -0.005
sqft_patio                0.0306      0.004      7.208      0.000
0.022       0.039
yr_built                 -0.8757      0.046    -18.901      0.000      -
0.966      -0.785
yr_renovated              0.0147      0.002      6.369      0.000
0.010       0.019
lat                     671.0281      6.466    103.785      0.000     65
8.355     683.701
long                    126.5948      7.616     16.623      0.000     11
1.668     141.522
month                    10.3536      3.075      3.367      0.001
4.327      16.380
day_of_year              -0.5702      0.101     -5.649      0.000      -
0.768      -0.372
```

```
sewer_PUBLIC             24.0610      2.922      8.236      0.000      1
8.334      29.788
heat_source_Gas          19.9302      2.099      9.496      0.000      1
5.816      24.044
heat_source_Gas/Solar    66.9906     16.163      4.145      0.000      3
5.310      98.671
waterfront               64.3185      8.711      7.384      0.000      4
7.245      81.392
nuisance                -15.0879      2.404     -6.276      0.000     -1
9.800     -10.376
view                     29.3481      1.267     23.159      0.000      2
6.864      31.832
greenbelt                47.6602      5.702      8.358      0.000      3
6.483      58.837
sqft_living_log          30.6220      6.047      5.064      0.000      1
8.770      42.474
==========================================================================
=====
Omnibus:                     3860.717   Durbin-Watson:
2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3668
7.399
Skew:                          -0.347   Prob(JB):
0.00
Kurtosis:                       8.564   Cond. No.                      6.6
3e+07
==========================================================================
=====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is co
rrectly specified.
[2] The condition number is large, 6.63e+07. This might indicate that the
re are
strong multicollinearity or other numerical problems.
```



Out[386]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for transformed model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

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.

In [387]:

```
X = new_outlier_data
# Fit a linear regression model
reg = LinearRegression().fit(X, y_sqrt)

# Predict the target values
y_pred = reg.predict(X)

# Calculate the mean absolute error
rmse = mean_squared_error(y_sqrt, y_pred)

print("Root mean squared error: ", rmse)
```

```
Root mean squared error:  21598.900418299952
```

Rechecking VIFs

In [388]:
```
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_ir

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

```
                Variable            VIF
0              bathrooms      24.288806
1               sqft_lot       1.299694
2                 floors      17.333161
3              condition      31.675256
4                  grade     137.636761
5             sqft_above      48.182791
6          sqft_basement       4.898542
7            sqft_garage       4.593212
8             sqft_patio       2.242563
9               yr_built    9589.111873
10          yr_renovated       1.205507
11                   lat  109063.702426
12                  long  123718.741939
13                 month     698.983641
14           day_of_year     614.160609
15          sewer_PUBLIC       8.786042
16       heat_source_Gas       3.862553
17  heat_source_Gas/Solar       1.015119
18            waterfront       1.202680
19              nuisance       1.268623
20                  view       1.425702
21             greenbelt       1.061871
22       sqft_living_log    2675.580945
```

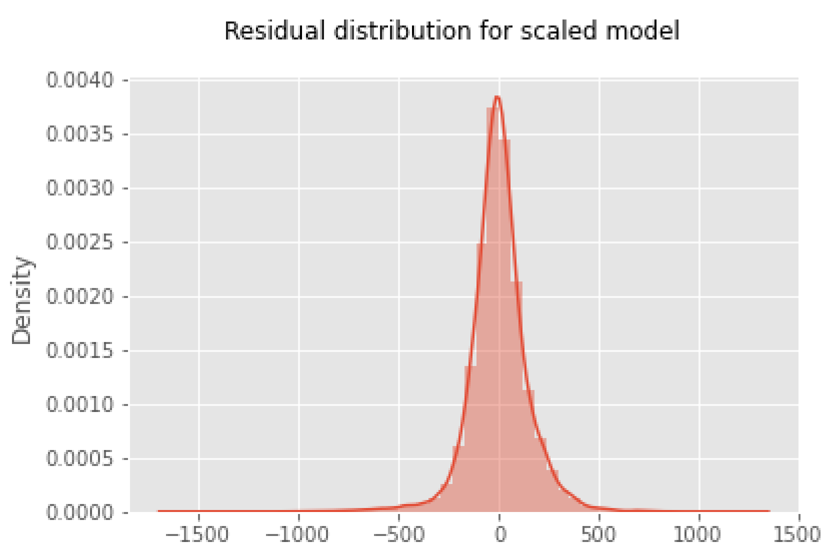
Scaling data

In [389]:
```
scaledX = (new_outlier_data - np.mean(new_outlier_data)) / np.std(new_outli
```

In [390]:
```
get_OLS_model('scaled',scaledX, y_sqrt)
```

```
                            OLS Regression Results                            
==============================================================================
=====
Dep. Variable:                  price   R-squared:                       
0.626
Model:                            OLS   Adj. R-squared:                  
0.625
Method:                 Least Squares   F-statistic:                     
2032.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):              
0.00
Time:                        20:31:40   Log-Likelihood:             -1.794
8e+05
No. Observations:               28004   AIC:                          3.59
0e+05
Df Residuals:                   27980   BIC:                          3.59
2e+05
Df Model:                          23                                         
Covariance Type:            nonrobust                                         
==============================================================================
================
                      coef    std err          t      P>|t|      
[0.025      0.975]
------------------------------------------------------------------------------
----------------
const             963.8260      0.879   1097.000      0.000     96
2.104     965.548
bathrooms          18.9843      1.476     12.864      0.000      1
6.092      21.877
sqft_lot           10.2982      0.964     10.683      0.000       
8.409      12.188
floors             -5.7423      1.272     -4.516      0.000      -
8.235      -3.250
condition          23.0260      0.988     23.299      0.000      2
1.089      24.963
grade              73.8055      1.436     51.381      0.000      7
0.990      76.621
sqft_above         65.5334      2.475     26.477      0.000      6
0.682      70.385
sqft_basement      17.9153      1.510     11.866      0.000      1
4.956      20.874
sqft_garage        -3.6773      1.224     -3.004      0.003      -
6.077      -1.278
sqft_patio          7.1036      0.985      7.208      0.000       
5.172       9.035
yr_built          -27.5969      1.460    -18.901      0.000     -3
0.459     -24.735
yr_renovated        6.0204      0.945      6.369      0.000       
4.168       7.873
lat               100.0913      0.964    103.785      0.000      9
8.201     101.982
long               18.2867      1.100     16.623      0.000      1
6.130      20.443
month              32.0850      9.529      3.367      0.001      1
3.408      50.762
day_of_year       -53.8286      9.528     -5.649      0.000     -7
2.505     -35.153
```

```
sewer_PUBLIC              8.5826      1.042      8.236      0.000
6.540      10.625
heat_source_Gas           9.2671      0.976      9.496      0.000
7.354      11.180
heat_source_Gas/Solar     3.6635      0.884      4.145      0.000
1.931       5.396
waterfront                7.0745      0.958      7.384      0.000
5.197       8.952
nuisance                 -5.6653      0.903     -6.276      0.000      -
7.435      -3.896
view                     23.0870      0.997     23.159      0.000      2
1.133      25.041
greenbelt                 7.5074      0.898      8.358      0.000
5.747       9.268
sqft_living_log          12.8341      2.534      5.064      0.000
7.867      17.801
==========================================================================
=====
Omnibus:                     3860.717   Durbin-Watson:
2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            3668
7.399
Skew:                          -0.347   Prob(JB):
0.00
Kurtosis:                       8.564   Cond. No.
33.1
==========================================================================
=====

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



Out[390]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for scaled model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Adding waterfront dummies to the model

In [391]:
```
water_data = pd.concat([scaledX,waterfront_dummies], axis=1)
```

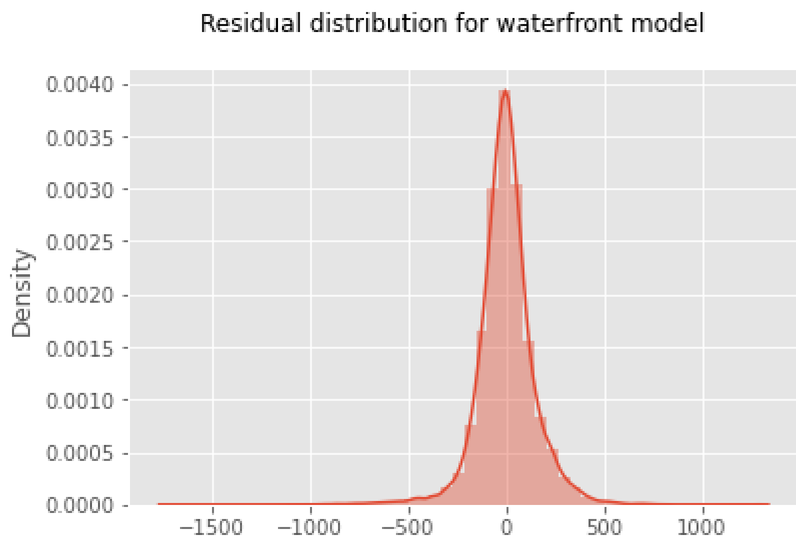
In [392]:
```
water_data.columns
```

Out[392]:
```
Index(['bathrooms', 'sqft_lot', 'floors', 'condition', 'grade', 'sqft_abo
ve',
       'sqft_basement', 'sqft_garage', 'sqft_patio', 'yr_built',
       'yr_renovated', 'lat', 'long', 'month', 'day_of_year', 'sewer_PUBL
IC',
       'heat_source_Gas', 'heat_source_Gas/Solar', 'waterfront', 'nuisanc
e',
       'view', 'greenbelt', 'sqft_living_log', 'water_Elliot Bay',
       'water_Lake Sammamish', 'water_Lake Washington', 'water_Puget Soun
d',
       'water_other'],
      dtype='object')
```

In [393]:
```
get_OLS_model('waterfront',water_data,y_sqrt)
```

```
                            OLS Regression Results                            
==========================================================================
=====
Dep. Variable:                  price   R-squared:                       
0.634
Model:                            OLS   Adj. R-squared:                  
0.634
Method:                 Least Squares   F-statistic:                     
1732.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):              
0.00
Time:                        20:31:43   Log-Likelihood:             -1.791
6e+05
No. Observations:               28004   AIC:                          3.58
4e+05
Df Residuals:                   27975   BIC:                          3.58
6e+05
Df Model:                          28                                         
Covariance Type:            nonrobust                                         
==========================================================================
================
                    coef    std err          t      P>|t|      
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const           927.7565      6.548    141.684      0.000     91
4.922     940.591
bathrooms        18.6287      1.459     12.766      0.000      1
5.769      21.489
sqft_lot         11.0210      0.955     11.543      0.000       
9.150      12.892
floors           -6.0926      1.258     -4.844      0.000      -
8.558      -3.627
condition        23.5247      0.978     24.065      0.000      2
1.609      25.441
grade            69.5427      1.431     48.587      0.000      6
6.737      72.348
sqft_above       64.4651      2.448     26.330      0.000      5
9.666      69.264
sqft_basement    17.7160      1.494     11.857      0.000      1
4.787      20.645
sqft_garage      -3.3880      1.214     -2.791      0.005      -
5.768      -1.008
sqft_patio        7.8678      0.975      8.070      0.000       
5.957       9.779
yr_built        -26.1333      1.447    -18.056      0.000     -2
8.970     -23.296
yr_renovated      6.5953      0.935      7.055      0.000       
4.763       8.428
lat             100.4427      1.010     99.490      0.000      9
8.464     102.422
long             11.5037      1.127     10.205      0.000       
9.294      13.713
month            31.1044      9.420      3.302      0.001      1
2.641      49.568
day_of_year     -52.8929      9.420     -5.615      0.000     -7
1.356     -34.430
```

```
sewer_PUBLIC              5.4042      1.064      5.079      0.000
3.319       7.490
heat_source_Gas           9.3205      0.965      9.654      0.000
7.428      11.213
heat_source_Gas/Solar     3.5643      0.874      4.079      0.000
1.851       5.277
waterfront                7.0714      0.949      7.448      0.000
5.210       8.932
nuisance                 -5.3175      0.893     -5.953      0.000      -
7.068      -3.567
view                     23.1952      0.986     23.517      0.000      2
1.262      25.128
greenbelt                 7.5679      0.888      8.520      0.000
5.827       9.309
sqft_living_log          14.7779      2.507      5.894      0.000
9.864      19.692
water_Elliot Bay         -0.4515      8.572     -0.053      0.958     -1
7.253      16.349
water_Lake Sammamish    140.2069      8.237     17.022      0.000     12
4.062     156.352
water_Lake Washington   -19.8540      9.433     -2.105      0.035     -3
8.343      -1.365
water_Puget Sound         6.8364      8.548      0.800      0.424      -
9.918      23.591
water_other              35.4111      6.609      5.358      0.000      2
2.457      48.366
==========================================================================
=====
Omnibus:                     3994.853   Durbin-Watson:
2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4061
9.628
Skew:                          -0.349   Prob(JB):
0.00
Kurtosis:                       8.859   Cond. No.
41.8
==========================================================================
=====

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



Out[393]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

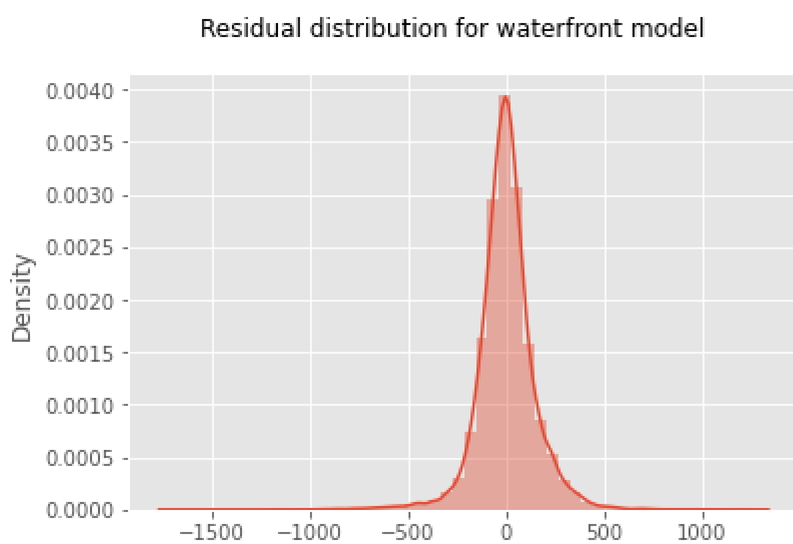
Elliot Bay and Puget Sound present high pvalues indicating a lack of statistical significance. These will be dropped from the model.

In [394]:
```
water_data = water_data.drop(['water_Elliot Bay','water_Puget Sound'], axis
```

In [395]:
```
get_OLS_model('waterfront',water_data,y_sqrt)
```

```
                            OLS Regression Results                            
==============================================================================
=====
Dep. Variable:                  price   R-squared:                       
0.634
Model:                            OLS   Adj. R-squared:                  
0.634
Method:                 Least Squares   F-statistic:                     
1865.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):              
0.00
Time:                        20:31:45   Log-Likelihood:            -1.791
6e+05
No. Observations:               28004   AIC:                         3.58
4e+05
Df Residuals:                   27977   BIC:                         3.58
6e+05
Df Model:                          26                                         
Covariance Type:            nonrobust                                         
==============================================================================
================
                          coef    std err          t      P>|t|      
[0.025      0.975]
------------------------------------------------------------------------------
----------------
const               930.1301      3.373    275.797      0.000     92
3.520     936.740
bathrooms            18.6098      1.459     12.755      0.000      1
5.750      21.470
sqft_lot             11.0471      0.954     11.577      0.000       
9.177      12.917
floors               -6.1257      1.257     -4.872      0.000      -
8.590      -3.661
condition            23.5314      0.977     24.077      0.000      2
1.616      25.447
grade                69.5142      1.431     48.584      0.000      6
6.710      72.319
sqft_above           64.4517      2.448     26.332      0.000      5
9.654      69.249
sqft_basement        17.6850      1.493     11.842      0.000      1
4.758      20.612
sqft_garage          -3.3364      1.213     -2.751      0.006      -
5.714      -0.959
sqft_patio            7.8816      0.975      8.087      0.000       
5.971       9.792
yr_built            -26.0584      1.445    -18.028      0.000     -2
8.891     -23.225
yr_renovated          6.5990      0.935      7.060      0.000       
4.767       8.431
lat                 100.3041      0.995    100.854      0.000      9
8.355     102.253
long                 11.4506      1.126     10.171      0.000       
9.244      13.657
month                31.0770      9.420      3.299      0.001      1
2.614      49.540
day_of_year         -52.8566      9.419     -5.611      0.000     -7
1.319     -34.394
```

```
sewer_PUBLIC               5.3772      1.062      5.061      0.000
3.295       7.460
heat_source_Gas            9.3409      0.965      9.678      0.000
7.449      11.233
heat_source_Gas/Solar      3.5717      0.874      4.088      0.000
1.859       5.284
waterfront                 7.1336      0.947      7.530      0.000
5.277       8.990
nuisance                  -5.3242      0.893     -5.965      0.000      -
7.074      -3.575
view                      23.1869      0.986     23.512      0.000      2
1.254      25.120
greenbelt                  7.5681      0.888      8.521      0.000
5.827       9.309
sqft_living_log           14.8136      2.506      5.910      0.000
9.901      19.726
water_Lake Sammamish     137.8850      5.965     23.117      0.000     12
6.194     149.576
water_Lake Washington    -22.1141      7.532     -2.936      0.003     -3
6.878      -7.350
water_other               33.0328      3.495      9.453      0.000      2
6.183      39.882
==========================================================================
=====
Omnibus:                     3992.548   Durbin-Watson:
2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4059
4.858
Skew:                          -0.348   Prob(JB):
0.00
Kurtosis:                       8.857   Cond. No.
33.1
==========================================================================
=====

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



Out[395]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

Recheck VIFs

In [396]:
```
get_vifs(water_data)
```

	Variable	VIF
0	bathrooms	2.822139
1	sqft_lot	1.207074
2	floors	2.095512
3	condition	1.266178
4	grade	2.708837
5	sqft_above	7.942106
6	sqft_basement	2.955381
7	sqft_garage	1.950322
8	sqft_patio	1.259318
9	yr_built	2.769200
10	yr_renovated	1.158303
11	lat	1.311024
12	long	1.646079
13	month	117.635393
14	day_of_year	117.622422
15	sewer_PUBLIC	1.494553
16	heat_source_Gas	1.235084
17	heat_source_Gas/Solar	1.012087
18	waterfront	1.189363
19	nuisance	1.056349
20	view	1.288746
21	greenbelt	1.045751
22	sqft_living_log	8.326010
23	water_Lake Sammamish	1.133347
24	water_Lake Washington	1.160076
25	water_other	1.008041

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

In [397]:
```
water_data = water_data.drop(['month','day_of_year'], axis =1)
```

In [398]:
```
get_vifs(water_data)
```

```
                 Variable       VIF
0               bathrooms  2.820882
1                sqft_lot  1.206917
2                  floors  2.095106
3               condition  1.265521
4                   grade  2.708681
5              sqft_above  7.941105
6           sqft_basement  2.955260
7             sqft_garage  1.949919
8              sqft_patio  1.259183
9                yr_built  2.767781
10           yr_renovated  1.158123
11                    lat  1.311003
12                   long  1.645893
13           sewer_PUBLIC  1.494486
14        heat_source_Gas  1.235073
15  heat_source_Gas/Solar  1.012068
16             waterfront  1.189304
17               nuisance  1.056119
18                   view  1.288704
19              greenbelt  1.045747
20        sqft_living_log  8.325701
21   water_Lake Sammamish  1.133158
22  water_Lake Washington  1.159873
23            water_other  1.007988
```

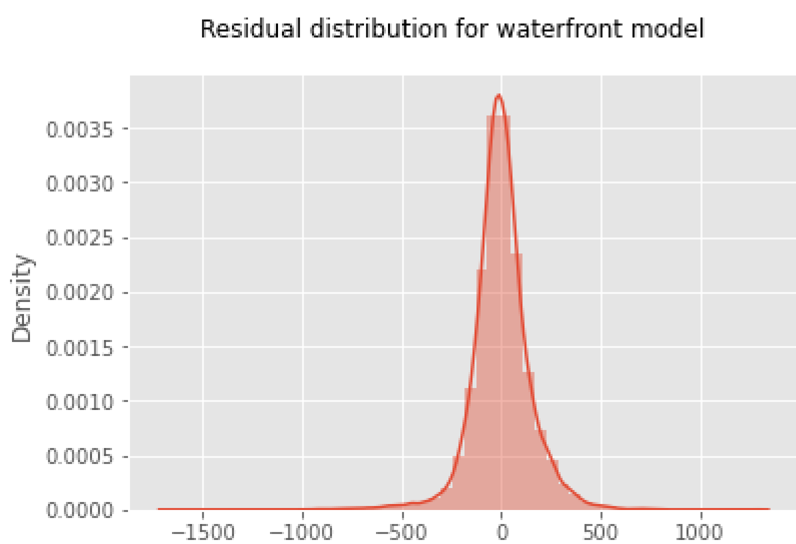
All VIFs are now below 10 with the majority less than 3, meaning the issue of collinearity is now for the most part solved.

Final model

In [399]:
```
get_OLS_model('waterfront',water_data,y_sqrt)
```

```
                            OLS Regression Results                            
==========================================================================
=====
Dep. Variable:                  price   R-squared:                       
0.626
Model:                            OLS   Adj. R-squared:                  
0.625
Method:                 Least Squares   F-statistic:                     
1949.
Date:                Thu, 09 Mar 2023   Prob (F-statistic):              
0.00
Time:                        20:31:59   Log-Likelihood:            -1.794
8e+05
No. Observations:               28004   AIC:                         3.59
0e+05
Df Residuals:                   27979   BIC:                         3.59
2e+05
Df Model:                          24                                         
Covariance Type:            nonrobust                                         
==========================================================================
================
                         coef    std err          t      P>|t|      
[0.025      0.975]
--------------------------------------------------------------------------
----------------
const                929.1767      3.411    272.414      0.000     92
2.491     935.862
bathrooms             17.8321      1.475     12.086      0.000      1
4.940      20.724
sqft_lot              10.7765      0.965     11.166      0.000       
8.885      12.668
floors                -5.7693      1.272     -4.537      0.000      -
8.262      -3.277
condition             23.4380      0.988     23.717      0.000      2
1.501      25.375
grade                 69.7639      1.447     48.208      0.000      6
6.927      72.600
sqft_above            63.9056      2.476     25.815      0.000      5
9.053      68.758
sqft_basement         17.4519      1.510     11.554      0.000      1
4.491      20.412
sqft_garage           -3.1052      1.227     -2.531      0.011      -
5.509      -0.701
sqft_patio             7.7906      0.986      7.903      0.000       
5.858       9.723
yr_built             -26.3626      1.462    -18.037      0.000     -2
9.227     -23.498
yr_renovated           6.5683      0.945      6.948      0.000       
4.715       8.421
lat                  100.3684      1.006     99.778      0.000      9
8.397     102.340
long                  11.7060      1.139     10.281      0.000       
9.474      13.938
sewer_PUBLIC           5.4272      1.075      5.050      0.000       
3.321       7.533
heat_source_Gas        9.3428      0.976      9.570      0.000       
7.429      11.256
```

```
heat_source_Gas/Solar     3.5477      0.884      4.014      0.000
1.816       5.280
waterfront                7.0093      0.958      7.315      0.000
5.131       8.887
nuisance                 -5.6582      0.903     -6.268      0.000      -
7.428      -3.889
view                     23.0579      0.997     23.117      0.000      2
1.103      25.013
greenbelt                 7.5699      0.898      8.427      0.000
5.809       9.331
sqft_living_log          15.1773      2.535      5.987      0.000      1
0.209      20.146
water_Lake Sammamish    137.7419      6.033     22.833      0.000     12
5.918     149.566
water_Lake Washington   -22.6046      7.618     -2.967      0.003     -3
7.537      -7.672
water_other              34.1416      3.534      9.660      0.000      2
7.214      41.069
==========================================================================
=====
Omnibus:                     3707.653   Durbin-Watson:
2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            3577
0.221
Skew:                          -0.301   Prob(JB):
0.00
Kurtosis:                       8.504   Cond. No.
21.4
==========================================================================
=====

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



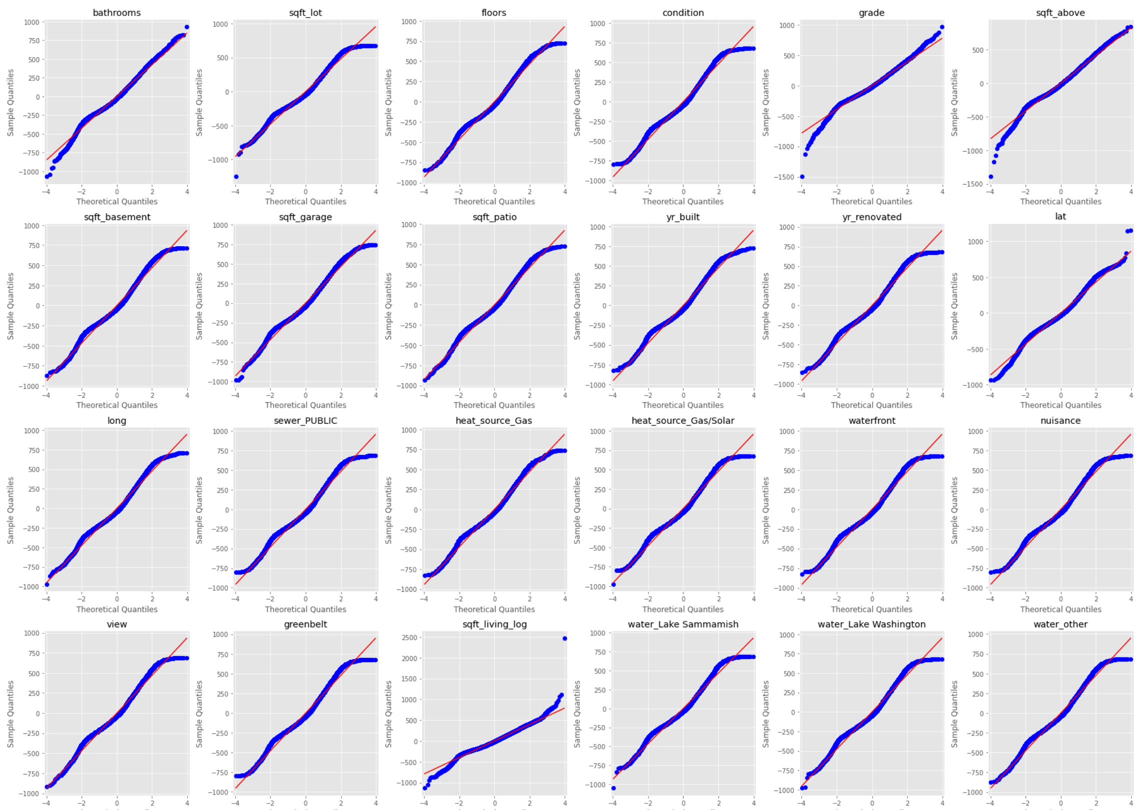
Out[399]:
```
(None,
 Text(0.5, 0.98, 'Residual distribution for waterfront model'),
 <AxesSubplot:ylabel='Density'>,
 None)
```

In [400]:
```
water_data.columns
```

Out[400]:
```
Index(['bathrooms', 'sqft_lot', 'floors', 'condition', 'grade', 'sqft_abo
ve',
       'sqft_basement', 'sqft_garage', 'sqft_patio', 'yr_built',
       'yr_renovated', 'lat', 'long', 'sewer_PUBLIC', 'heat_source_Gas',
       'heat_source_Gas/Solar', 'waterfront', 'nuisance', 'view', 'greenb
elt',
       'sqft_living_log', 'water_Lake Sammamish', 'water_Lake Washingto
n',
       'water_other'],
      dtype='object')
```

Constructing QQplots for all independent variables within the model

In [401]:
```
get_model_qqplots(water_data, y_sqrt)
```



In [402]:
```
model = sm.OLS(y_sqrt, sm.add_constant(water_data))
results = model.fit()
model_residual = results.resid
model_params = results.params

print(results.params)
```

```
const                      929.176743
bathrooms                   17.832111
sqft_lot                    10.776486
floors                      -5.769272
condition                   23.438047
grade                       69.763857
sqft_above                  63.905587
sqft_basement               17.451898
sqft_garage                 -3.105150
sqft_patio                   7.790561
yr_built                   -26.362631
yr_renovated                 6.568336
lat                        100.368386
long                        11.706009
sewer_PUBLIC                 5.427152
heat_source_Gas              9.342830
heat_source_Gas/Solar        3.547675
waterfront                   7.009294
nuisance                    -5.658225
view                        23.057945
greenbelt                    7.569852
sqft_living_log             15.177339
water_Lake Sammamish       137.741862
water_Lake Washington      -22.604616
water_other                 34.141607
dtype: float64
```

We have a linear model with the dependent variable (price) square root transformed, and the following independent variables and their corresponding coefficients:

- Bathrooms: 17.832111
- Sqft_lot: 10.776486
- Floors: -5.769272
- Condition: 23.438047
- Grade: 69.763857
- Sqft_above: 63.905587
- Sqft_basement: 17.451898
- Sqft_garage: -3.105150
- Sqft_patio: 7.790561
- Yr_built: -26.362631
- Yr_renovated: 6.568336
- Lat: 100.368386
- Long: 11.706009
- Sewer_PUBLIC: 5.427152
- Heat_source_Gas: 9.342830
- Heat_source_Gas/Solar: 3.547675

- Waterfront: 7.009294
- Nuisance: -5.658225
- View: 23.057945
- Greenbelt: 7.569852
- Sqft_living_log: 15.177339
- Water_Lake Sammamish: 137.741862
- Water_Lake Washington: -22.604616
- Water_other: 34.141607

A positive coefficient indicates that as the corresponding independent variable increases, the square root of the price of the house also increases, while a negative coefficient indicates that as the corresponding independent variable increases, the square root of the price of the house decreases.

In this model, we see that the most important variable in predicting the square root of house prices is the latitude of the house, with a coefficient of 100.368386. This suggests that houses located further north tend to have higher prices. The next most important variable is water proximity, with Water_Lake Sammamish variable having a very high coefficient of 137.741862, suggesting that houses located near this lake tend to have much higher prices than other houses. On the other hand, the Water_Lake Washington variable has a negative coefficient, indicating that houses located near this lake tend to have lower prices than other houses.

Other important variables include the grade of the house, the square footage of the house above ground, and the condition of the house, all with coefficients greater than 20. The number of bathrooms, square footage of the basement, and the size of the view from the house are also important, with coefficients greater than 15.

On the other hand, variables such as the square footage of the garage and the presence of a nuisance nearby have negative coefficients, indicating that houses with larger garages or located near nuisances tend to have lower prices. The year the house was built and the longitude of the house also have negative coefficients, suggesting that older houses and houses located further west tend to have lower prices.

Overall, these results suggest that there are many factors that contribute to the price of a house, and that location, house size and quality, and the presence of nearby amenities all play important roles in determining the square root of house prices.