Creation of Model Statistics DataFrame

- Student name: Greg Osborne
- Student pace: self paced / part time
- Scheduled project review date/time: 8/2/22
- Instructor name: Clause Fried
- Blog post URL: https://medium.com/@gregosborne

This Jupyter Notebook is exclusively for creating a DataFrame, and an excel file, that includes basic model data of multiple possible combinations of variables put into a model using ordinary least squares through the ModelStatistics Python library. It should be treated as a giant function rather than a presentation of the logic behind this. For that, please see the student Jupyter notebook.

In [1]:
```
import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import ols
import matplotlib.pyplot as plt
import numpy as np
import scipy.stats as stats
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import PolynomialFeatures
#from itertools import combinations
#from itertools import combinations_with_replacement
from itertools import product
from itertools import permutations

plt.style.use('seaborn')
data = pd.read_csv('data/kc_house_data.csv')
pd.set_option("display.max_columns", None)
```

This next cell scrubs the data the same way described in the Student Jupyter Notebook.

In [2]:
```
#Dropping columns advised by client.
columns_to_drop = ['date', 'view', 'sqft_above', 'sqft_basement',
                   'yr_renovated', 'zipcode', 'lat', 'long', 'sqft_living15',
                   'sqft_lot15']
data = data.drop(columns_to_drop,axis='columns')

#Fixing Waterfront data
data.loc[data['waterfront'].isna(),'waterfront'] = 'NO'
data['waterfront'].replace({'NO':0,'YES':1},inplace=True)

#Fixing Condition data
change_numerical_condition = {'Poor' : 1, 'Fair' : 2, 'Average' : 3,
                              'Good' : 4, 'Very Good' : 5}
data['condition'].replace(change_numerical_condition, inplace=True)

#Fixing Grade data
change_numerical_grade = {'3 Poor' : 3, '4 Low' : 4, '5 Fair' : 5,
                          '6 Low Average' : 6, '7 Average' : 7, '8 Good' : 8,
                          '9 Better' : 9, '10 Very Good' : 10,
                          '11 Excellent' : 11, '12 Luxury' : 12,
                          '13 Mansion' : 13}

data['grade'].replace(change_numerical_grade, inplace=True)
```

```
#Dropping id column
model_data = data.drop(columns='id')

index_nums = model_data.reset_index()

#Dropping a single outlier due to bad bedroom data.
bedrooms_max = 12
bed_drop = index_nums['bedrooms'][index_nums['bedrooms']>=bedrooms_max]
model_data = index_nums.drop(index=bed_drop.index,columns='index')
```

These next cells create the functions to create the transformations needed for this analysis.

In [3]:
```
#This logs all the independent variables in the data.
def log_data(df, var,keep=False):
    order = df.columns
    df_log = pd.DataFrame()
    new_col = []
    for i in range(len(order)):
        if order[i] not in var:
            df_log[order[i]] = df[order[i]]
        else:
            new_col.append(order[i] + '_log')
            df_log[order[i] + '_log'] = df[order[i]].map(lambda x : np.log(x))
    if keep == True:
        df_new = df.copy()
        for col in new_col:
            df_new[new_col] = df_log[new_col]
        return df_new
    return df_log
```

In [4]:
```
#This performs a min-max scaling of all independent variables in the data.
def scale_data(df,var,keep=False):
    order = df.columns
    df_scale = pd.DataFrame()
    new_col = []
    for i in range(len(order)):
        if order[i] not in var:
            df_scale[order[i]] = df[order[i]]
        else:
            mn = df[order[i]].min()
            mx = df[order[i]].max()
            new_col.append(order[i] + '_scale')
            df_scale[order[i] + '_scale'] = df[
                order[i]].map(lambda x : (x-mn)/(mx-mn))
    if keep == True:
        df_new = df.copy()
        for col in new_col:
            df_new[new_col] = df_scale[new_col]
        return df_new
    return df_scale
```

In [5]:
```
#This squares all the independent variables in the data.
def sq_data(df, var,keep=False):
    order = df.columns
    df_pow2 = pd.DataFrame()
    new_col = []
    for i in range(len(order)):
        if order[i] not in var:
            df_pow2[order[i]] = df[order[i]]
        else:
```

```
            new_col.append(order[i] + '_sq')
            df_pow2[order[i] + '_sq'] = df[order[i]].map(lambda x : x**2)
    if keep == True:
        df_new = df.copy()
        for col in new_col:
            df_new[new_col] = df_pow2[new_col]
        return df_new
    return df_pow2
```

In [6]:

```
#This returns five DataFrames, one for untransformed data and one for each
#transformation this model analyzes.
def tran_data(df,var):
    if 'waterfront' in var:
        var.remove('waterfront')
    df_scale = scale_data(df, var)
    df_log = log_data(df, var)
    df_log_scale = scale_data(df_log,
                list(df_log.drop(columns=['price','waterfront']).columns))
    df_sq = sq_data(df,var)
    return [df, df_scale, df_log, df_log_scale, df_sq]
```

Since the testing for the multicollinearity is critical to the function definitons below, I will leave this test here.

In [7]:

```
#This creates the initial DataFrame to be added to later.
multicollinearity_test_data = model_data.copy()

#This returns five DataFrames, one for untransformed data and one for each
#transformation this model analyzes.
dfr,dfs,dfl,dfls,dfsq = tran_data(
    model_data,list(model_data.iloc[0:1,1:].columns))

#Getting rid of values I don't want repeated.
dfs = dfs.drop(columns=['price','waterfront'])
dfl = dfl.drop(columns=['price','waterfront'])
dfls = dfls.drop(columns=['price','waterfront'])
dfsq = dfsq.drop(columns=['price','waterfront'])

#Constructiong a large DataFrame with all raw and transformed values.
multicollinearity_test_data = pd.concat([dfr,dfs,dfl,dfls,dfsq],axis=1)
del dfr,dfs,dfl,dfls,dfsq

#Running the multicollinearity test and displaying the results.
multicollinearity_test = multicollinearity_test_data.corr()
for c in multicollinearity_test.columns:
    multicollinearity_test[c] = multicollinearity_test[c].map('{:.3f}'.format)
multicollinearity_test
```

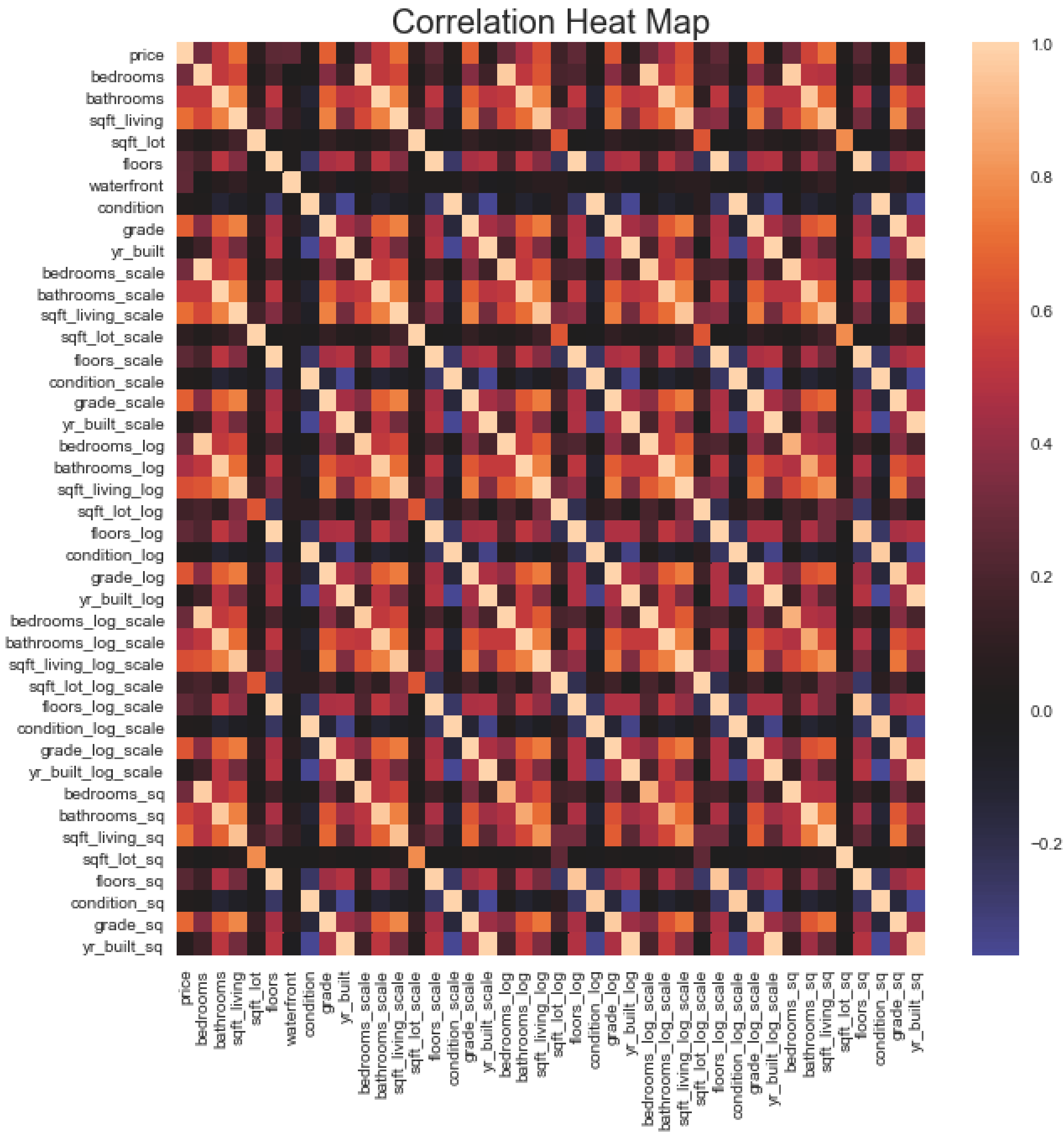
Out[7]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	condition	grade	yr_bui
price	1.000	0.316	0.526	0.702	0.090	0.257	0.264	0.036	0.668	0.05
bedrooms	0.316	1.000	0.528	0.593	0.034	0.184	-0.002	0.023	0.366	0.16
bathrooms	0.526	0.528	1.000	0.756	0.088	0.503	0.064	-0.126	0.666	0.50
sqft_living	0.702	0.593	0.756	1.000	0.173	0.354	0.105	-0.059	0.763	0.3
sqft_lot	0.090	0.034	0.088	0.173	1.000	-0.005	0.021	-0.009	0.115	0.05
floors	0.257	0.184	0.503	0.354	-0.005	1.000	0.021	-0.264	0.459	0.48
waterfront	0.264	-0.002	0.064	0.105	0.021	0.021	1.000	0.017	0.083	-0.02

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	condition	grade	yr_bui
condition	0.036	0.023	-0.126	-0.059	-0.009	-0.264	0.017	1.000	-0.147	-0.36
grade	0.668	0.366	0.666	0.763	0.115	0.459	0.083	-0.147	1.000	0.44
yr_built	0.054	0.161	0.507	0.318	0.053	0.489	-0.024	-0.362	0.448	1.00
bedrooms_scale	0.316	1.000	0.528	0.593	0.034	0.184	-0.002	0.023	0.366	0.16
bathrooms_scale	0.526	0.528	1.000	0.756	0.088	0.503	0.064	-0.126	0.666	0.50
sqft_living_scale	0.702	0.593	0.756	1.000	0.173	0.354	0.105	-0.059	0.763	0.31
sqft_lot_scale	0.090	0.034	0.088	0.173	1.000	-0.005	0.021	-0.009	0.115	0.05
floors_scale	0.257	0.184	0.503	0.354	-0.005	1.000	0.021	-0.264	0.459	0.48
condition_scale	0.036	0.023	-0.126	-0.059	-0.009	-0.264	0.017	1.000	-0.147	-0.36
grade_scale	0.668	0.366	0.666	0.763	0.115	0.459	0.083	-0.147	1.000	0.44
yr_built_scale	0.054	0.161	0.507	0.318	0.053	0.489	-0.024	-0.362	0.448	1.00
bedrooms_log	0.299	0.972	0.525	0.581	0.033	0.192	-0.008	0.026	0.381	0.19
bathrooms_log	0.456	0.521	0.971	0.708	0.075	0.501	0.047	-0.113	0.644	0.53
sqft_living_log	0.612	0.638	0.762	0.955	0.150	0.368	0.079	-0.050	0.743	0.35
sqft_lot_log	0.162	0.190	0.101	0.345	0.639	-0.237	0.072	0.074	0.184	-0.00
floors_log	0.267	0.206	0.513	0.377	0.001	0.991	0.021	-0.261	0.468	0.47
condition_log	0.037	0.028	-0.118	-0.052	-0.011	-0.255	0.015	0.989	-0.133	-0.34
grade_log	0.635	0.374	0.665	0.743	0.104	0.463	0.073	-0.145	0.993	0.46
yr_built_log	0.053	0.161	0.506	0.317	0.053	0.485	-0.024	-0.360	0.447	1.00
bedrooms_log_scale	0.299	0.972	0.525	0.581	0.033	0.192	-0.008	0.026	0.381	0.19
bathrooms_log_scale	0.456	0.521	0.971	0.708	0.075	0.501	0.047	-0.113	0.644	0.53
sqft_living_log_scale	0.612	0.638	0.762	0.955	0.150	0.368	0.079	-0.050	0.743	0.35
sqft_lot_log_scale	0.162	0.190	0.101	0.345	0.639	-0.237	0.072	0.074	0.184	-0.00
floors_log_scale	0.267	0.206	0.513	0.377	0.001	0.991	0.021	-0.261	0.468	0.47
condition_log_scale	0.037	0.028	-0.118	-0.052	-0.011	-0.255	0.015	0.989	-0.133	-0.34
grade_log_scale	0.635	0.374	0.665	0.743	0.104	0.463	0.073	-0.145	0.993	0.46
yr_built_log_scale	0.053	0.161	0.506	0.317	0.053	0.485	-0.024	-0.360	0.447	1.00
bedrooms_sq	0.310	0.974	0.499	0.565	0.031	0.165	0.003	0.021	0.327	0.11
bathrooms_sq	0.573	0.500	0.961	0.757	0.099	0.456	0.078	-0.124	0.639	0.43
sqft_living_sq	0.727	0.487	0.674	0.936	0.178	0.299	0.123	-0.058	0.692	0.25
sqft_lot_sq	0.035	0.003	0.031	0.054	0.788	0.009	0.002	-0.004	0.034	0.01
floors_sq	0.237	0.150	0.472	0.311	-0.013	0.985	0.020	-0.258	0.431	0.48
condition_sq	0.037	0.021	-0.126	-0.061	-0.008	-0.262	0.018	0.995	-0.151	-0.36
grade_sq	0.693	0.354	0.657	0.772	0.123	0.445	0.091	-0.145	0.994	0.42
yr_built_sq	0.055	0.161	0.509	0.319	0.052	0.493	-0.025	-0.363	0.449	1.00

In [8]:

```
import seaborn as sns
plt.figure(figsize = (10,10))
heat = sns.heatmap(multicollinearity_test_data.corr(), center=0)
heat.set_title('Correlation Heat Map',size=20);
```



These functions provide the means to create the excel file with the model statistics needed.

In [9]:

```
#This function creates a model from a Pandas DataFrame and a list of variables
#to include. Price will always be the dependent variable.
def make_model(mod, lst):
    outcome = 'price'
    predictors = mod[lst]
    pred_sum = '+'.join(predictors.columns)
    formula = outcome + '~' + pred_sum
    return ols(formula=formula, data=mod).fit()
```

In [10]:

```
#This reformats the variable names into a more aesthetic format.
```

```python
def new_name(var):
    rename_var = {'bedrooms' : 'Bed', 'bathrooms' : 'Bath',
        'sqft_living': 'SF Liv', 'grade' : 'Grade',
        'sqft_lot' : 'SF Lot','floors': 'Floors',
        'waterfront': 'WF' ,'condition': 'Cond',
        'yr_built' : 'Yr',
        'bedrooms_log' : 'Log(Bed)', 'bathrooms_log' : 'Log(Bath)',
        'sqft_living_log': 'Log(SF Liv)', 'grade_log' : 'Log(Grade)',
        'sqft_lot_log' : 'Log(SF Lot)','floors_log': 'Log(Floors)',
        'waterfront_log': 'Log(WF)' ,'condition_log': 'Log(Cond)',
        'yr_built_log' : 'Log(Yr)',
        'bedrooms_scale' : 'Norm Bed','bathrooms_scale':'Norm Bath',
        'sqft_living_scale':'Norm SF Liv','grade_scale':'Norm Grade',
        'sqft_lot_scale':'Norm SF Lot','floors_scale':'Norm Floors',
        'waterfront_scale': 'Norm WF' ,'condition_scale':'Norm Cond',
        'yr_built_scale' : 'Norm Yr',
        'bedrooms_log_scale':'Norm Log(Bed)',
        'bathrooms_log_scale':'Norm Log(Bath)',
        'sqft_living_log_scale':'Norm Log(SF Liv)',
        'grade_log_scale':'Norm Log(Grade)',
        'sqft_lot_log_scale':'Norm Log(SF Lot)',
        'floors_log_scale':'Norm Log(Floors)',
        'waterfront_log_scale': 'Norm Log(WF)' ,
        'condition_log_scale':'Norm Log(Cond)',
        'yr_built_log_scale' : 'Norm Log(Yr)',
        'bedrooms_sq' : 'Bed²', 'bathrooms_sq' : 'Bath²',
        'sqft_living_sq': 'SF Liv²', 'grade_sq' : 'Grade²',
        'sqft_lot_sq' : 'SF Lot²','floors_sq': 'Floors²',
        'waterfront_sq': 'WF²' ,'condition_sq': 'Cond²',
        'yr_built_sq' : 'Yr²'}
    new_var = []
    for v in var:
        new_var.append(rename_var[v])
    return new_var
```

In [11]:

```python
#This returns a list of strings that identify the variables used in that
#row's model.
def model_variables(lst):
    fv = new_name(lst) #fv = fomatted variable
    if len(fv) == 1:
        text = f'{fv[0]}'
        return text
    elif len(fv) == 2:
        text = f'{fv[0]} & {fv[1]}'
        return text
    text = f'{fv[0]}, {fv[1]}, {fv[2]}'
    i = 3
    while i < len(fv):
        text = text + f', {fv[i]}'
        i += 1
    return text
```

In [12]:

```python
#This formats the DataFrame's column names with better aesthetics.
def format_df(df, var):
    rename_coefficients = {'bedrooms' : 'Bed Coef', 'bathrooms' : 'Bath Coef',
        'sqft_living': 'SF Liv Coef', 'grade' : 'Grade Coef',
        'sqft_lot' : 'SF Lot Coef','floors': 'Floors Coef',
        'waterfront': 'WF Coef' ,'condition': 'Cond Coef',
        'yr_built' : 'Yr Coef',
        'bedrooms_log' : 'Log(Bed) Coef', 'bathrooms_log' : 'Log(Bath) Coef',
        'sqft_living_log': 'Log(SF Liv) Coef', 'grade_log' : 'Log(Grade) Coef',
```

```
        'sqft_lot_log' : 'Log(SF Lot) Coef','floors_log': 'Log(Floors) Coef',
        'waterfront_log': 'Log(WF) Coef' ,'condition_log': 'Log(Cond) Coef',
        'yr_built_log' : 'Log(Yr) Coef',
        'bedrooms_scale' : 'Norm Bed Coef','bathrooms_scale':'Norm Bath Coef',
        'sqft_living_scale':'Norm SF Liv Coef','grade_scale':'Norm Grade Coef',
        'sqft_lot_scale':'Norm SF Lot Coef','floors_scale':'Norm Floors Coef',
        'waterfront_scale': 'Norm WF Coef' ,'condition_scale':'Norm Cond Coef',
        'yr_built_scale' : 'Norm Yr Coef',
        'bedrooms_log_scale':'Norm Log(Bed) Coef',
        'bathrooms_log_scale':'Norm Log(Bath) Coef',
        'sqft_living_log_scale':'Norm Log(SF Liv) Coef',
        'grade_log_scale':'Norm Log(Grade) Coef',
        'sqft_lot_log_scale':'Norm Log(SF Lot) Coef',
        'floors_log_scale':'Norm Log(Floors) Coef',
        'waterfront_log_scale': 'Norm Log(WF) Coef' ,
        'condition_log_scale':'Norm Log(Cond) Coef',
        'yr_built_log_scale' : 'Norm Log(Yr) Coef',
        'bedrooms_sq' : 'Bed² Coef', 'bathrooms_sq' : 'Bath² Coef',
        'sqft_living_sq': 'SF Liv² Coef', 'grade_sq' : 'Grade² Coef',
        'sqft_lot_sq' : 'SF Lot² Coef','floors_sq': 'Floors² Coef',
        'waterfront_sq': 'WF² Coef' ,'condition_sq': 'Cond² Coef',
        'yr_built_sq' : 'Yr² Coef'}

    df['R²'] = df['R²'].map('{:.3f}'.format)
    df['P-value'] = df['P-value'].map('{:.1f}'.format)
    df['Intercept'] = df['Intercept'].map('{:.0f}'.format)
    for v in var:
        if v in df.columns:
            df[v] = df[v].map('{:.2f}'.format)
    floats = list(df.iloc[:,1:].columns)
    for obj in floats:
        df[obj] = df[obj].astype(float)

    df = df.rename(columns = rename_coefficients)
    return df
```

In [13]:

```
#Splitting the DataFrame parameter into individual DataFrames, each all the
#same variable with all the applicable applications.
def var_tran_dfs(df):
    drop = ['price','waterfront']
    var = list(df.drop(columns=drop).columns)
    num = len(var) + 1
    dfr,dfs,dfl,dfls,dfsq = tran_data(df,var)
    dfp,dfbd,dfbt,dflv,dflt,dffl,dfwf,dfct,dfgd,dfyb=0,0,0,0,0,0,0,0,0,0
    create_dfs = [dfp,dfbd,dfbt,dflv,dflt,dffl,dfwf,dfct,dfgd,dfyb]
    for i in range(len(df.columns)):
        create_dfs[i] = pd.DataFrame()
        for dfi in [dfr,dfs,dfl,dfls,dfsq]:
            create_dfs[i][dfi.iloc[:,i].name] = dfi.iloc[:,i]
    iter_df = []
    static_df = []
    static_df_num = [0]
    for i in range(len(create_dfs)):
        if i in static_df_num:
            static_df.append(create_dfs[i])
        else:
            iter_df.append(create_dfs[i])
    return static_df, iter_df
```

In [14]:

```
#This creates the DataFrame with the included model's pertinent statistics.
def model_stats(df,var):
```

```python
    #A DataFrame to sort the outcomes and make it look pretty.
    dfr = pd.DataFrame(columns =['Model Variable(s)',
                                 'R²','P-value','Neg Coef?'])

    neg = False

    #Running StatsModels
    m = make_model(df,var)

    #Testing to see if this model meets the requirements. The function returns
    #an empty DataFrame if it does not.
    d = dict(m.params)
    if m.rsquared <= 0.54:
        return dfr
    for v in var:
        if d[v] < 0:
            neg = True
            return dfr

    #Building the DataFrame to return.
    d.update({'Model Variable(s)' : model_variables(var),
              'R²' : m.rsquared,'P-value' : m.f_pvalue,
              'Neg Coef?' : neg})
    dfr = dfr.append(d,ignore_index = True)

    #Formatting the DataFrame to return.
    dfr = format_df(dfr,var)
    dfr['Neg Coef?'] = dfr['Neg Coef?'].astype(bool)

    return dfr
```

In [15]:

```python
#This runs thousands of models, with the specified number of variables per
#model and returns a DataFrame with the pertinent statistics.
def mega_models(df,beg,end):

    #Variable declaration
    #Variable that keeps a list of the column names of each df iteration.
    check = []
    #Variables related to printing text to provide feedback to the user to
    #ensure the function is still running.
    skip = False
    skip_count = 0
    discarded_models = 0
    cnt_var = 0
    cur_var = beg
    #Creating an empty list of DataFrames (dfs) that I will use to build each
    #iteration of the DataFrames in the for loop below.
    dfs = [None,None,None,None,None,None,None,None,None]

    #Splitting the DataFrame parameter into individual DataFrames, each all the
    #same variable with all the applicable applications.
    dep_df, ind_df =var_tran_dfs(df)

    #Creating the DataFrame with the columns in an order that makes sense.
    dfr_col = ['Model Variable(s)','R²','P-value','Neg Coef?','Intercept']
    for i in range(len(ind_df[0].columns)):
        for df_temp in ind_df:
            if i > len(df_temp.columns)-1:
                continue
            else:
                dfr_col.append(df_temp.iloc[:,i].name)
    dfr = pd.DataFrame(columns=dfr_col)
```

```python
    dfr = format_df(dfr, dfr_col)
    dfr['Neg Coef?'] = dfr['Neg Coef?'].astype(bool)

    #Creating dictionary of combinations. The key is the number of variables
    #per iteration, and the value is a list of which variables to include.
    num_var = dict(zip([0,1,2,3,4,5,6,7,8,9],
                  [[],[],[],[],[],[],[],[],[],[]]))
    for lst in product(range(2),repeat=9):
        num_var[sum(lst)].append(lst)

    #This for loop is the major work of this function. It creates a DataFrame
    #with pertinent information on different combinations of variables
    #specified in the parameters.
    for num in range(beg,end+1):

        #Prints feedback text so the user knows the program is running.
        cnt_var += 1
        print(f'Combos of {cur_var} variables ({cnt_var}/{end-beg+1}).')
        cur_var += 1

        #Resetting count variables related to each combination of variables
        model_count = 0
        skipped = 0

        #Calculating the total number of iterations for this combination of
        #variables.
        var_tot_iter = len(num_var[num])*len(list(product(
            range(len(ind_df[0].columns)),repeat=num)))

        #This first for loop iterates through the combinations of which
        #variables to include in the current interation.
        for vars_used in num_var[num]:
            #The second for loop specifies which transformations to use for
            #the current iteration.
            for tran_nums in product(range(len(ind_df[0].columns)),repeat=num):
                i,j,=0,0 #Specifies iteration of transformation.

                #Prints feedback so the user knows the program is running.
                #if ten thousand DataFrames are skipped, it prints this.
                if skip == True:
                    skip_count += 1
                    if skip_count % 10000 == 0:
                        print(f'Model {model_count} of {var_tot_iter}.')
                #If the program completed a thousand DataFrame iterations:
                if skip == False:
                    if model_count % 1000 == 0:
                        print(f'Model {model_count} of {var_tot_iter}.')
                        skip_count = 0

                #Sets a list of which variables are used to check if the
                #the current iteration is a repeat.
                check_iter = []

                #Checks if current iteration includes variables that are
                #I previously learned are multicollinear. This portion of code
                #doesn't check for multicollinearity, but rather filters out
                #variables I previously tested positive for multicollinearity.
                if vars_used[2] == 1:
                    if vars_used[1] == 1 or vars_used[7] == 1:
                        #This function returns True or False whether the
                        #values are multicollinear.
                        if multi_corr(vars_used,tran_nums):
```

```
                        skip=True
                        skipped += 1
                        var_tot_iter -= 1
                        continue

            #This creates the list of DataFrames for this iteration.
            for var in vars_used:
                if var == 1:
                    if i == 5:
                        dfs[i] = ind_df[i].copy()
                        check_iter.append(var)
                    else:
                        dfs[i] = ind_df[i].iloc[:,tran_nums[j]].copy()
                        check_iter.append([var,tran_nums[j]])
                    i+=1
                    j+=1

                #This creates the list to check if this iteration
                #is a repeat.
                else:
                    dfs[i]=None
                    check_iter.append(var)
                    i+=1

            #If the iteration is a repeat, this skips the model creation,
            #which saves time to run the function. Otherwise, the iteration
            #list is added to the check list to be filtered out later.
            if check_iter in check:
                skip=True
                skipped += 1
                var_tot_iter -= 1
                continue
            else:
                check.append(check_iter)

            #This puts the gleaned variables together into one DataFrame.
            iter_df = pd.concat([dep_df[0],dfs[0],dfs[1],dfs[2],
                                 dfs[3],dfs[4],dfs[5],dfs[6],
                                 dfs[7],dfs[8],],axis = 1)

            #And finally, the program runs the iteration through
            #StatsModels, and creates a DataFrame of the critical
            #statistics.
            col = list(iter_df.iloc[:,1:].columns)
            dfr_temp = model_stats(iter_df,col)
            model_count += 1 #We have successfully created a model.
            skip=False #We didn't skip this iteration.

            #The model_stats function will return a an empty DataFrame if
            #the model's data didn't meet certain requirements.
            if dfr_temp.empty:
                discarded_models += 1
                continue

            #DataFrame statistics is now added to the DataFrame to return.
            dfr = dfr.append(dfr_temp, ignore_index=True)

    #Ending text to confirm the function performed correctly.
    print(f'Model {model_count} of {var_tot_iter}.',
          f'Skipped {skipped} iterations.')
    print(f'{discarded_models} models discarded due to',
           'low R² or negative coefficients.')
```

```
    print(len(dfr), 'Models met minimum requirements.')

    #Organizing the DataFrame to return.
    dfr = dfr.sort_values('R²',ascending=False)
    dfr = dfr.reset_index(drop=True)
    return dfr
```

In [16]:

```
#This tests if the current iteration is multicollinear. It returns a boolean.
def multi_corr(var_tup,tran_tup):
    #Variable declaration
    detected = False
    var_lst = []

    #This creates a list of which variables are included and which
    #transformation form it is in.
    i=0
    for n in range(9):
        var_lst.append(var_tup[n] - 1)
        if var_lst[n] == 0:
            var_lst[n] = tran_tup[i]
            i+=1

    #These two if loops test if the combination of variables match the
    #previously calculated multicollinear pairs.
    if var_lst[2] in [0,1]:
        if var_lst[1] in [0,1,4] or var_lst[7] in [0,1,4]:
            detected = True
            return detected
    if var_lst[2] in [2,3]:
        if var_lst[1] in [0,1,2,3]:
            detected = True

    return detected
```

In [17]:

```
mega1 = mega_models(model_data,1,1)
mega1
```

```
Combos of 1 variables (1/1).
Model 0 of 45.
Model 41 of 41. Skipped 4 iterations.
41 models discarded due to low R² or negative coefficients.
0 Models met minimum requirements.
```

Out[17]:

Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef	Norm Bed Coef	Norm Bath Coef

There were no single variable models that met the requirements.

In [18]:

```
mega2 = mega_models(model_data,2,2)
mega2
```

```
Combos of 2 variables (1/1).
Model 0 of 900.
Model 720 of 720. Skipped 180 iterations.
714 models discarded due to low R² or negative coefficients.
6 Models met minimum requirements.
```

Out[18]:

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef	N
0	SF Liv^2 & Grade^2	0.588	0.0	False	-29632.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	SF Liv^2 & Grade	0.581	0.0	False	-396479.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	98988.34	NaN	
2	SF Liv^2 & Norm Grade	0.581	0.0	False	-99514.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	SF Liv^2 & Log(Grade)	0.573	0.0	False	-1035517.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	SF Liv^2 & Norm Log(Grade)	0.573	0.0	False	-284330.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
5	SF Liv^2 & WF	0.560	0.0	False	287040.0	NaN	NaN	NaN	NaN	NaN	795513.11	NaN	NaN	NaN	

In [19]:

```
mega3 = mega_models(model_data,3,3)
mega3
```

```
Combos of 3 variables (1/1).
Model 0 of 10500.
Model 1000 of 9900.
Model 2000 of 9134.
Model 3000 of 8874.
Model 4000 of 8374.
Model 5000 of 7598.
Model 6000 of 7368.
Model 7000 of 7168.
Model 7098 of 7098. Skipped 3402 iterations.
7046 models discarded due to low R² or negative coefficients.
52 Models met minimum requirements.
```

Out[19]:

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Gr C
0	SF Liv^2, WF, Grade^2	0.619	0.0	False	-26515.0	NaN	NaN	NaN	NaN	NaN	791493.93	NaN	N
1	SF Liv^2, WF, Grade	0.612	0.0	False	-395900.0	NaN	NaN	NaN	NaN	NaN	798901.88	NaN	9923
2	SF Liv^2, WF, Norm Grade	0.612	0.0	False	-98181.0	NaN	NaN	NaN	NaN	NaN	798901.88	NaN	N
3	SF Liv^2, WF, Log(Grade)	0.605	0.0	False	-1043976.0	NaN	NaN	NaN	NaN	NaN	804722.88	NaN	N
4	SF Liv^2, WF, Norm Log(Grade)	0.605	0.0	False	-286654.0	NaN	NaN	NaN	NaN	NaN	804722.88	NaN	N

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Gr C
5	SF Liv², Cond², Grade²	0.602	0.0	False	-164589.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
6	SF Liv², Norm Cond, Grade²	0.602	0.0	False	-214731.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
7	SF Liv², Cond, Grade²	0.602	0.0	False	-281149.0	NaN	NaN	NaN	NaN	NaN	NaN	66417.20	N
8	SF Liv², Log(Cond), Grade²	0.600	0.0	False	-323805.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
9	SF Liv², Norm Log(Cond), Grade²	0.600	0.0	False	-323805.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
10	SF Liv², Cond², Norm Grade	0.595	0.0	False	-238729.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
11	SF Liv², Cond², Grade	0.595	0.0	False	-559145.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	10680
12	SF Liv², Norm Cond, Grade	0.594	0.0	False	-605894.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	10634
13	SF Liv², Norm Cond, Norm Grade	0.594	0.0	False	-286858.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
14	SF Liv², Cond, Grade	0.594	0.0	False	-670812.0	NaN	NaN	NaN	NaN	NaN	NaN	64917.49	10634
15	SF Liv², Cond, Norm Grade	0.594	0.0	False	-351775.0	NaN	NaN	NaN	NaN	NaN	NaN	64917.49	N
16	SF Liv², Norm Log(Cond), Norm Grade	0.593	0.0	False	-391080.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
17	SF Liv², Log(Cond), Norm Grade	0.593	0.0	False	-391080.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Gr C
18	SF Liv², Log(Cond), Grade	0.593	0.0	False	-706788.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	10523
19	SF Liv², Norm Log(Cond), Grade	0.593	0.0	False	-706788.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	10523
20	SF Liv², Cond², Log(Grade)	0.586	0.0	False	-1246753.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
21	SF Liv², Cond², Norm Log(Grade)	0.586	0.0	False	-435195.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
22	SF Liv², Cond, Norm Log(Grade)	0.585	0.0	False	-542413.0	NaN	NaN	NaN	NaN	NaN	NaN	62542.59	N
23	SF Liv², Cond, Log(Grade)	0.585	0.0	False	-1349787.0	NaN	NaN	NaN	NaN	NaN	NaN	62542.59	N
24	SF Liv², Norm Cond, Norm Log(Grade)	0.585	0.0	False	-479871.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
25	SF Liv², Norm Cond, Log(Grade)	0.585	0.0	False	-1287244.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
26	SF Liv², Norm Log(Cond), Norm Log(Grade)	0.584	0.0	False	-576344.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
27	SF Liv², Norm Log(Cond), Log(Grade)	0.584	0.0	False	-1374211.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
28	SF Liv², Log(Cond), Norm Log(Grade)	0.584	0.0	False	-576344.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
29	SF Liv², Log(Cond), Log(Grade)	0.584	0.0	False	-1374211.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
30	SF Liv², WF, Cond²	0.566	0.0	False	217108.0	NaN	NaN	NaN	NaN	NaN	786995.54	NaN	N

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Gr C
31	SF Liv², WF, Norm Cond	0.566	0.0	False	184220.0	NaN	NaN	NaN	NaN	NaN	787443.36	NaN	N
32	SF Liv², WF, Cond	0.566	0.0	False	142192.0	NaN	NaN	NaN	NaN	NaN	787443.36	42027.65	N
33	SF Liv², WF, Norm Log(Cond)	0.565	0.0	False	109045.0	NaN	NaN	NaN	NaN	NaN	788560.10	NaN	N
34	SF Liv², WF, Log(Cond)	0.565	0.0	False	109045.0	NaN	NaN	NaN	NaN	NaN	788560.10	NaN	N
35	Bath, SF Liv², WF	0.563	0.0	False	231335.0	NaN	34390.81	NaN	NaN	NaN	801848.71	NaN	N
36	Norm Bath, SF Liv², WF	0.563	0.0	False	248531.0	NaN	NaN	NaN	NaN	NaN	801848.71	NaN	N
37	Bath², SF Liv², WF	0.563	0.0	False	267377.0	NaN	NaN	NaN	NaN	NaN	800290.82	NaN	N
38	SF Liv², Floors, WF	0.562	0.0	False	244630.0	NaN	NaN	NaN	NaN	31659.19	798899.39	NaN	N
39	SF Liv², Norm Floors, WF	0.562	0.0	False	276289.0	NaN	NaN	NaN	NaN	NaN	798899.39	NaN	N
40	SF Liv², Log(Floors), WF	0.562	0.0	False	276163.0	NaN	NaN	NaN	NaN	NaN	799081.53	NaN	N
41	SF Liv², Norm Log(Floors), WF	0.562	0.0	False	276163.0	NaN	NaN	NaN	NaN	NaN	799081.53	NaN	N
42	SF Liv², Floors², WF	0.562	0.0	False	266888.0	NaN	NaN	NaN	NaN	NaN	798438.93	NaN	N
43	Log(Bath), SF Liv², WF	0.562	0.0	False	263839.0	NaN	NaN	NaN	NaN	NaN	801614.78	NaN	N
44	Norm Log(Bath), SF Liv², WF	0.562	0.0	False	228725.0	NaN	NaN	NaN	NaN	NaN	801614.78	NaN	N
45	Norm SF Liv, WF, Norm Log(Grade)	0.559	0.0	False	-384242.0	NaN	NaN	NaN	NaN	NaN	870100.03	NaN	N
46	Norm SF Liv, WF, Log(Grade)	0.559	0.0	False	-1074987.0	NaN	NaN	NaN	NaN	NaN	870100.03	NaN	N
47	SF Liv, WF, Log(Grade)	0.559	0.0	False	-1147789.0	NaN	NaN	196.76	NaN	NaN	870100.03	NaN	N

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Gr C
48	SF Liv, WF, Norm Log(Grade)	0.559	0.0	False	-457044.0	NaN	NaN	196.76	NaN	NaN	870100.03	NaN	N
49	$Bath^2$, WF, $Grade^2$	0.546	0.0	False	-190870.0	NaN	NaN	NaN	NaN	NaN	890771.52	NaN	N
50	Norm Log(SF Liv), WF, $Grade^2$	0.543	0.0	False	-368537.0	NaN	NaN	NaN	NaN	NaN	897180.44	NaN	N
51	Log(SF Liv), WF, $Grade^2$	0.543	0.0	False	-1500584.0	NaN	NaN	NaN	NaN	NaN	897180.44	NaN	N

In [20]:

```
mega4 = mega_models(model_data,4,4)
mega4
```

```
Combos of 4 variables (1/1).
Model 0 of 78750.
Model 1000 of 77350.
Model 2000 of 75850.
Model 3000 of 75250.
Model 4000 of 75250.
Model 5000 of 72870.
Model 6000 of 71890.
Model 7000 of 70570.
Model 8000 of 70060.
Model 9000 of 68420.
Model 10000 of 68230.
Model 11000 of 67730.
Model 12000 of 66230.
Model 13000 of 66230.
Model 14000 of 65170.
Model 15000 of 64730.
Model 16000 of 64670.
Model 17000 of 63230.
Model 18000 of 63230.
Model 19000 of 61560.
Model 20000 of 58394.
Model 21000 of 56228.
Model 22000 of 55228.
Model 23000 of 54628.
Model 24000 of 54628.
Model 25000 of 53128.
Model 26000 of 53128.
Model 27000 of 51628.
Model 28000 of 51628.
Model 29000 of 51008.
Model 30000 of 50888.
Model 31000 of 49298.
Model 32000 of 49148.
Model 33000 of 48648.
Model 34000 of 48498.
Model 35000 of 47998.
Model 36000 of 47998.
Model 37000 of 46998.
Model 38000 of 46498.
Model 39000 of 46494.
Model 40000 of 45998.
```

```
Model 41000 of 45994.
Model 42000 of 45168.
Model 43000 of 43258.
Model 43118 of 43118. Skipped 35632 iterations.
42832 models discarded due to low R² or negative coefficients.
286 Models met minimum requirements.
```

Out[20]:

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	(
0	SF Liv², WF, Cond², Grade²	0.632	0.0	False	-156463.0	NaN	NaN	NaN	NaN	NaN	778168.02	NaN	NaN	I
1	SF Liv², WF, Cond, Grade²	0.631	0.0	False	-268789.0	NaN	NaN	NaN	NaN	NaN	778932.53	63963.5	NaN	I
2	SF Liv², WF, Norm Cond, Grade²	0.631	0.0	False	-204826.0	NaN	NaN	NaN	NaN	NaN	778932.53	NaN	NaN	I
3	SF Liv², WF, Norm Log(Cond), Grade²	0.630	0.0	False	-310505.0	NaN	NaN	NaN	NaN	NaN	780897.24	NaN	NaN	I
4	SF Liv², WF, Log(Cond), Grade²	0.630	0.0	False	-310505.0	NaN	NaN	NaN	NaN	NaN	780897.24	NaN	NaN	I
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
281	Bath², SF Liv², Log(Floors), Log(Cond)	0.540	0.0	False	28227.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
282	Bath², SF Liv², Norm Floors, Norm Log(Cond)	0.540	0.0	False	26842.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
283	Bath², SF Liv², Norm Floors, Log(Cond)	0.540	0.0	False	26842.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
284	Bath², SF Liv², Floors, Norm Log(Cond)	0.540	0.0	False	-7823.0	NaN	NaN	NaN	NaN	34664.69	NaN	NaN	NaN	I
285	Bath, SF Liv², Norm Log(Floors), Cond²	0.540	0.0	False	141943.0	NaN	25720.98	NaN	NaN	NaN	NaN	NaN	NaN	I

286 rows × 46 columns

In [21]:
```python
mega5 = mega_models(model_data,5,5)
mega5
```

```
Combos of 5 variables (1/1).
Model 0 of 393750.
Model 1000 of 390250.
Model 2000 of 388750.
Model 3000 of 388750.
Model 4000 of 388750.
Model 5000 of 386350.
Model 6000 of 382350.
Model 7000 of 378185.
Model 8000 of 377850.
Model 9000 of 377850.
Model 10000 of 373600.
Model 11000 of 369560.
Model 12000 of 369300.
Model 13000 of 369300.
Model 14000 of 364850.
Model 15000 of 360935.
Model 16000 of 360750.
Model 17000 of 360750.
Model 18000 of 360750.
Model 19000 of 360750.
Model 20000 of 360750.
Model 21000 of 360000.
Model 22000 of 360000.
Model 23000 of 358020.
Model 24000 of 353270.
Model 25000 of 349850.
Model 26000 of 349850.
Model 27000 of 349850.
Model 28000 of 349850.
Model 29000 of 345850.
Model 30000 of 342350.
Model 31000 of 342350.
Model 32000 of 342350.
Model 33000 of 342350.
Model 34000 of 338350.
Model 35000 of 334850.
Model 36000 of 334850.
Model 37000 of 334850.
Model 38000 of 334850.
Model 39000 of 334850.
Model 40000 of 334850.
Model 41000 of 334850.
Model 42000 of 334850.
Model 43000 of 334850.
Model 44000 of 334850.
Model 45000 of 331770.
Model 46000 of 327770.
Model 47000 of 325300.
Model 48000 of 316130.
Model 49000 of 314080.
Model 50000 of 312830.
Model 51000 of 311030.
Model 52000 of 303370.
Model 53000 of 300120.
Model 54000 of 298870.
Model 55000 of 296970.
Model 56000 of 292370.
Model 57000 of 286460.
Model 58000 of 284960.
Model 59000 of 283410.
Model 60000 of 282160.
Model 61000 of 277810.
Model 62000 of 276810.
```

```
Model 63000 of 276810.
Model 64000 of 276810.
Model 65000 of 274110.
Model 66000 of 270110.
Model 67000 of 269310.
Model 68000 of 269310.
Model 69000 of 269310.
Model 70000 of 266610.
Model 71000 of 262610.
Model 72000 of 261810.
Model 73000 of 261810.
Model 74000 of 261810.
Model 75000 of 261810.
Model 76000 of 261810.
Model 77000 of 261810.
Model 78000 of 261810.
Model 79000 of 261810.
Model 80000 of 261810.
Model 81000 of 260070.
Model 82000 of 256070.
Model 83000 of 254030.
Model 84000 of 253710.
Model 85000 of 252600.
Model 86000 of 247980.
Model 87000 of 245460.
Model 88000 of 245160.
Model 89000 of 245010.
Model 90000 of 245010.
Model 91000 of 245010.
Model 92000 of 244860.
Model 93000 of 244560.
Model 94000 of 244260.
Model 95000 of 241021.
Model 96000 of 236585.
Model 97000 of 236210.
Model 98000 of 235860.
Model 99000 of 235860.
Model 100000 of 235860.
Model 101000 of 235860.
Model 102000 of 235651.
Model 103000 of 235276.
Model 104000 of 234570.
Model 105000 of 229820.
Model 106000 of 227460.
Model 107000 of 227460.
Model 108000 of 227460.
Model 109000 of 227310.
Model 110000 of 227010.
Model 111000 of 226710.
Model 112000 of 226710.
Model 113000 of 226710.
Model 114000 of 226710.
Model 115000 of 224210.
Model 116000 of 224210.
Model 117000 of 224210.
Model 118000 of 224110.
Model 119000 of 220110.
Model 120000 of 216710.
Model 121000 of 216710.
Model 122000 of 216710.
Model 123000 of 216710.
Model 124000 of 216710.
Model 125000 of 216710.
Model 126000 of 216710.
Model 127000 of 216710.
Model 128000 of 216710.
Model 129000 of 216710.
```

```
Model 130000 of 213570.
Model 131000 of 209570.
Model 132000 of 209210.
Model 133000 of 209210.
Model 134000 of 209210.
Model 135000 of 209210.
Model 136000 of 209210.
Model 137000 of 209210.
Model 138000 of 209210.
Model 139000 of 209210.
Model 140000 of 209210.
Model 141000 of 207070.
Model 142000 of 203070.
Model 143000 of 201710.
Model 144000 of 201710.
Model 145000 of 201710.
Model 146000 of 201710.
Model 147000 of 201710.
Model 148000 of 201710.
Model 149000 of 201710.
Model 150000 of 201710.
Model 151000 of 201710.
Model 152000 of 200554.
Model 153000 of 197880.
Model 154000 of 196260.
Model 155000 of 195000.
Model 156000 of 191870.
Model 157000 of 184200.
Model 158000 of 183000.
Model 159000 of 181000.
Model 160000 of 179700.
Model 161000 of 176000.
Model 162000 of 174700.
Model 163000 of 172760.
Model 164000 of 171500.
Model 165000 of 167700.
Model 166000 of 166350.
Model 166200 of 166200. Skipped 227550 iterations.
165640 models discarded due to low R² or negative coefficients.
560 Models met minimum requirements.
```

Out[21]:

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef
0	Bath², SF Liv², WF, Cond², Grade²	0.632	0.0	False	-157416.0	NaN	NaN	NaN	NaN	NaN	778702.83	NaN	NaN	NaN
1	Bath², SF Liv², WF, Cond, Grade²	0.631	0.0	False	-270388.0	NaN	NaN	NaN	NaN	NaN	779482.03	64293.87	NaN	NaN
2	Bath², SF Liv², WF, Norm Cond, Grade²	0.631	0.0	False	-206094.0	NaN	NaN	NaN	NaN	NaN	779482.03	NaN	NaN	NaN
3	Bath², SF Liv², WF, Log(Cond), Grade²	0.630	0.0	False	-312169.0	NaN	NaN	NaN	NaN	NaN	781395.80	NaN	NaN	NaN

	Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef
4	Bath^2, SF Liv^2, WF, Norm Log(Cond), Grade^2	0.630	0.0	False	-312169.0	NaN	NaN	NaN	NaN	NaN	781395.80	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
555	Norm Log(Bed), SF Lot^2, WF, Cond, Grade^2	0.540	0.0	False	-569304.0	NaN	NaN	NaN	NaN	NaN	901755.82	72979.66	NaN	NaN
556	Log(Bath), SF Lot^2, WF, Cond^2, Grade^2	0.540	0.0	False	-385095.0	NaN	NaN	NaN	NaN	NaN	892155.14	NaN	NaN	NaN
557	Log(Bed), SF Lot, WF, Cond, Grade^2	0.540	0.0	False	-569363.0	NaN	NaN	NaN	0.02	NaN	901583.35	72981.20	NaN	NaN
558	Norm Log(Bed), SF Lot^2, WF, Norm Cond, Grade^2	0.540	0.0	False	-496325.0	NaN	NaN	NaN	NaN	NaN	901755.82	NaN	NaN	NaN
559	Log(Bed), Norm SF Lot, WF, Norm Cond, Grade^2	0.540	0.0	False	-496373.0	NaN	NaN	NaN	NaN	NaN	901583.35	NaN	NaN	NaN

560 rows × 46 columns

In [22]:
```
mega6 = mega_models(model_data,6,6)
mega6
```

In [23]:
```
mega7 = mega_models(model_data,7,7)
mega7
```

```
Combos of 7 variables (1/1).
Model 0 of 2812500.
Model 900 of 2802500.
Model 1000 of 2801505.
Model 1850 of 2791505.
Model 2000 of 2790005.
Model 3000 of 2782250.
Model 4000 of 2778250.
Model 5000 of 2774250.
Model 6000 of 2770250.
Model 7000 of 2766250.
```

```
Model 8000 of 2762250.
Model 9000 of 2758250.
Model 10000 of 2754250.
Model 11000 of 2750250.
Model 12000 of 2746250.
Model 13000 of 2742250.
Model 14000 of 2738250.
Model 15000 of 2734250.
Model 16000 of 2730250.
Model 17000 of 2726250.
Model 18000 of 2722250.
Model 19000 of 2718250.
Model 20000 of 2714250.
Model 21000 of 2710250.
Model 22000 of 2706250.
Model 23000 of 2702250.
Model 24000 of 2698250.
Model 25000 of 2694250.
Model 26000 of 2690250.
Model 27000 of 2686250.
Model 27500 of 2676250.
Model 28000 of 2669750.
Model 28125 of 2659750.
Model 29000 of 2651380.
Model 29375 of 2641380.
Model 30000 of 2638060.
Model 30375 of 2628060.
Model 31000 of 2625250.
Model 31875 of 2615250.
Model 32000 of 2615000.
Model 32875 of 2605000.
Model 33000 of 2598500.
Model 33500 of 2588500.
Model 34000 of 2582000.
Model 34625 of 2572000.
Model 35000 of 2568000.
Model 35750 of 2558000.
Model 36000 of 2554000.
Model 36875 of 2544000.
Model 37000 of 2543750.
Model 38000 of 2539750.
Model 38250 of 2529750.
Model 39000 of 2525750.
Model 40000 of 2525750.
Model 41000 of 2525750.
Model 41375 of 2515750.
Model 42000 of 2513250.
Model 43000 of 2513250.
Model 44000 of 2513250.
Model 45000 of 2512505.
Model 46000 of 2511005.
Model 47000 of 2509505.
Model 48000 of 2503250.
Model 49000 of 2503250.
Model 50000 of 2503250.
Model 51000 of 2501935.
Model 52000 of 2500435.
Model 53000 of 2493250.
Model 54000 of 2493250.
Model 55000 of 2493250.
Model 56000 of 2487000.
Model 57000 of 2487000.
Model 58000 of 2487000.
Model 59000 of 2487000.
Model 60000 of 2487000.
Model 61000 of 2487000.
Model 62000 of 2487000.
```

```
Model 63000 of 2487000.
Model 64000 of 2487000.
Model 65000 of 2487000.
Model 65125 of 2477000.
Model 65750 of 2467000.
Model 66000 of 2458600.
Model 66875 of 2448600.
Model 67000 of 2444600.
Model 68000 of 2436855.
Model 68625 of 2426855.
Model 69000 of 2420350.
Model 70000 of 2416350.
Model 70500 of 2406350.
Model 71000 of 2399850.
Model 71125 of 2389850.
Model 72000 of 2383350.
Model 73000 of 2379350.
Model 73925 of 2369350.
Model 74000 of 2369100.
Model 74925 of 2359100.
Model 75000 of 2358850.
Model 75925 of 2348850.
Model 76000 of 2348600.
Model 77000 of 2344600.
Model 77375 of 2334600.
Model 78000 of 2328100.
Model 78000 of 2318100.
Model 79000 of 2308737.
Model 79700 of 2298737.
Model 80000 of 2295726.
Model 80450 of 2285726.
Model 81000 of 2277350.
Model 82000 of 2273350.
Model 83000 of 2269750.
Model 84000 of 2265750.
Model 85000 of 2261750.
Model 86000 of 2257750.
Model 87000 of 2253750.
Model 88000 of 2249750.
Model 89000 of 2245750.
Model 90000 of 2241750.
Model 91000 of 2237750.
Model 92000 of 2233750.
Model 93000 of 2229750.
Model 94000 of 2225750.
Model 95000 of 2221750.
Model 96000 of 2217750.
Model 97000 of 2213750.
Model 98000 of 2209750.
Model 99000 of 2202000.
Model 100000 of 2198000.
Model 101000 of 2192130.
Model 102000 of 2186250.
Model 103000 of 2182250.
Model 104000 of 2174500.
Model 105000 of 2170500.
Model 106000 of 2162755.
Model 107000 of 2158750.
Model 108000 of 2153560.
Model 109000 of 2147000.
Model 110000 of 2143000.
Model 111000 of 2135250.
Model 112000 of 2131250.
Model 113000 of 2124185.
Model 114000 of 2119500.
Model 115000 of 2115500.
Model 116000 of 2107750.
```

```
Model 117000 of 2103750.
Model 118000 of 2096000.
Model 119000 of 2092000.
Model 120000 of 2086130.
Model 121000 of 2080250.
Model 122000 of 2076250.
Model 123000 of 2073935.
Model 124000 of 2072435.
Model 125000 of 2071500.
Model 126000 of 2071500.
Model 127000 of 2071500.
Model 128000 of 2071500.
Model 129000 of 2071500.
Model 130000 of 2071500.
Model 131000 of 2071500.
Model 132000 of 2071500.
Model 133000 of 2071500.
Model 134000 of 2071500.
Model 135000 of 2070005.
Model 136000 of 2068505.
Model 137000 of 2067750.
Model 138000 of 2067750.
Model 139000 of 2067750.
Model 140000 of 2067750.
Model 141000 of 2067750.
Model 142000 of 2067750.
Model 143000 of 2067750.
Model 144000 of 2067750.
Model 145000 of 2067750.
Model 146000 of 2067560.
Model 147000 of 2066060.
Model 148000 of 2064560.
Model 149000 of 2064000.
Model 150000 of 2064000.
Model 151000 of 2064000.
Model 152000 of 2064000.
Model 153000 of 2064000.
Model 154000 of 2064000.
Model 155000 of 2064000.
Model 156000 of 2064000.
Model 157000 of 2064000.
Model 158000 of 2063630.
Model 159000 of 2062130.
Model 160000 of 2060630.
Model 161000 of 2060250.
Model 162000 of 2060250.
Model 163000 of 2060250.
Model 164000 of 2060250.
Model 165000 of 2060250.
Model 166000 of 2060250.
Model 167000 of 2060250.
Model 168000 of 2060250.
Model 169000 of 2060250.
Model 170000 of 2059685.
Model 171000 of 2058185.
Model 172000 of 2056685.
Model 173000 of 2056500.
Model 174000 of 2056500.
Model 175000 of 2056500.
Model 176000 of 2056500.
Model 177000 of 2056500.
Model 178000 of 2056500.
Model 179000 of 2056500.
Model 180000 of 2056500.
Model 181000 of 2056500.
Model 182000 of 2050855.
Model 183000 of 2046850.
```

```
Model 184000 of 2041860.
Model 185000 of 2035100.
Model 186000 of 2031100.
Model 187000 of 2023350.
Model 188000 of 2019350.
Model 189000 of 2012485.
Model 190000 of 2007600.
Model 191000 of 2003600.
Model 192000 of 1995850.
Model 193000 of 1991850.
Model 194000 of 1987850.
Model 195000 of 1983850.
Model 196000 of 1979850.
Model 197000 of 1975850.
Model 198000 of 1971850.
Model 199000 of 1967850.
Model 200000 of 1963850.
Model 201000 of 1959850.
Model 202000 of 1955850.
Model 203000 of 1951850.
Model 204000 of 1947850.
Model 205000 of 1943850.
Model 206000 of 1939850.
Model 207000 of 1935850.
Model 208000 of 1931850.
Model 209000 of 1927850.
Model 210000 of 1920100.
Model 211000 of 1916100.
Model 212000 of 1908350.
Model 213000 of 1904350.
Model 214000 of 1898476.
Model 215000 of 1892600.
Model 216000 of 1888600.
Model 217000 of 1880850.
Model 218000 of 1876850.
Model 219000 of 1869101.
Model 220000 of 1865100.
Model 221000 of 1861500.
Model 222000 of 1857500.
Model 223000 of 1853500.
Model 224000 of 1849500.
Model 225000 of 1845500.
Model 226000 of 1841500.
Model 227000 of 1837500.
Model 228000 of 1833500.
Model 229000 of 1829500.
Model 230000 of 1825500.
Model 231000 of 1821500.
Model 232000 of 1817500.
Model 233000 of 1813500.
Model 234000 of 1809500.
Model 235000 of 1805500.
Model 236000 of 1801500.
Model 237000 of 1797500.
Model 238000 of 1793500.
Model 239000 of 1789500.
Model 240000 of 1785500.
Model 241000 of 1781500.
Model 242000 of 1777500.
Model 243000 of 1773500.
Model 244000 of 1769500.
Model 245000 of 1765500.
Model 246000 of 1761500.
Model 247000 of 1757500.
Model 248000 of 1753500.
Model 249000 of 1749500.
Model 250000 of 1745500.
```

```
Model 251000 of 1741500.
Model 252000 of 1737500.
Model 253000 of 1736500.
Model 254000 of 1736500.
Model 255000 of 1736500.
Model 256000 of 1736500.
Model 257000 of 1736500.
Model 258000 of 1736500.
Model 259000 of 1736500.
Model 260000 of 1736500.
Model 261000 of 1736500.
Model 262000 of 1736500.
Model 263000 of 1736500.
Model 264000 of 1736500.
Model 265000 of 1736500.
Model 266000 of 1736500.
Model 267000 of 1736500.
Model 268000 of 1736500.
Model 269000 of 1736500.
Model 270000 of 1736500.
Model 271000 of 1736500.
Model 272000 of 1736500.
Model 273000 of 1736500.
Model 274000 of 1736500.
Model 275000 of 1736500.
Model 276000 of 1736500.
Model 277000 of 1736500.
Model 278000 of 1736500.
Model 279000 of 1736500.
Model 280000 of 1736500.
Model 281000 of 1736500.
Model 282000 of 1736500.
Model 283000 of 1736500.
Model 284000 of 1736500.
Model 285000 of 1736500.
Model 286000 of 1736500.
Model 287000 of 1736500.
Model 288000 of 1736500.
Model 289000 of 1736500.
Model 290000 of 1736500.
Model 291000 of 1736500.
Model 292000 of 1736500.
Model 293000 of 1736500.
Model 294000 of 1736500.
Model 295000 of 1736500.
Model 296000 of 1736500.
Model 297000 of 1736500.
Model 298000 of 1736500.
Model 299000 of 1736500.
Model 300000 of 1736500.
Model 301000 of 1736500.
Model 302000 of 1736500.
Model 303000 of 1736500.
Model 304000 of 1736500.
Model 305000 of 1736500.
Model 306000 of 1736500.
Model 307000 of 1736500.
Model 308000 of 1736500.
Model 309000 of 1736500.
Model 310000 of 1736500.
Model 311000 of 1736500.
Model 312000 of 1736500.
Model 313000 of 1736500.
Model 314000 of 1736500.
Model 315000 of 1736500.
Model 316000 of 1736500.
Model 317000 of 1736500.
```

```
Model 318000 of 1736500.
Model 319000 of 1736500.
Model 320000 of 1736500.
Model 321000 of 1736500.
Model 322000 of 1736500.
Model 323000 of 1736500.
Model 324000 of 1736500.
Model 325000 of 1736500.
Model 326000 of 1736500.
Model 327000 of 1736500.
Model 328000 of 1736500.
Model 329000 of 1736500.
Model 330000 of 1736500.
Model 331000 of 1733600.
Model 332000 of 1729600.
Model 333000 of 1725600.
Model 334000 of 1721600.
Model 335000 of 1717600.
Model 336000 of 1713600.
Model 337000 of 1709600.
Model 338000 of 1705600.
Model 339000 of 1701600.
Model 340000 of 1697600.
Model 341000 of 1693600.
Model 342000 of 1689600.
Model 343000 of 1685600.
Model 344000 of 1681600.
Model 345000 of 1677600.
Model 346000 of 1673600.
Model 347000 of 1669600.
Model 348000 of 1665600.
Model 349000 of 1661600.
Model 350000 of 1657600.
Model 351000 of 1653600.
Model 352000 of 1649600.
Model 353000 of 1645600.
Model 354000 of 1641600.
Model 355000 of 1637600.
Model 356000 of 1633600.
Model 357000 of 1629600.
Model 358000 of 1625600.
Model 359000 of 1621600.
Model 360000 of 1617600.
Model 361000 of 1613600.
Model 362000 of 1609600.
Model 363000 of 1605600.
Model 364000 of 1601600.
Model 365000 of 1597600.
Model 366000 of 1593600.
Model 367000 of 1589600.
Model 368000 of 1585600.
Model 369000 of 1581600.
Model 370000 of 1577600.
Model 371000 of 1573600.
Model 372000 of 1569600.
Model 373000 of 1565600.
Model 374000 of 1561600.
Model 375000 of 1557600.
Model 376000 of 1553600.
Model 377000 of 1549600.
Model 377650 of 1539600.
Model 378000 of 1535450.
Model 378725 of 1525450.
Model 379000 of 1521400.
Model 379750 of 1511400.
Model 380000 of 1506850.
Model 380825 of 1496850.
```

```
Model 381000 of 1494050.
Model 381775 of 1484050.
Model 382000 of 1481255.
Model 383000 of 1473750.
Model 384000 of 1470880.
Model 385000 of 1469060.
Model 386000 of 1467250.
Model 387000 of 1466000.
Model 388000 of 1463500.
Model 389000 of 1461000.
Model 390000 of 1459000.
Model 391000 of 1457000.
Model 392000 of 1455750.
Model 393000 of 1455750.
Model 394000 of 1450750.
Model 395000 of 1450005.
Model 396000 of 1448185.
Model 397000 of 1445500.
Model 398000 of 1445500.
Model 399000 of 1443000.
Model 400000 of 1440310.
Model 401000 of 1438500.
Model 402000 of 1436500.
Model 403000 of 1435250.
Model 404000 of 1435250.
Model 405000 of 1430250.
Model 406000 of 1428250.
Model 407000 of 1426250.
Model 408000 of 1425000.
Model 409000 of 1425000.
Model 409950 of 1415000.
Model 410000 of 1413600.
Model 410780 of 1403600.
Model 411000 of 1400735.
Model 411650 of 1390735.
Model 412000 of 1387100.
Model 412725 of 1377100.
Model 413000 of 1372850.
Model 413800 of 1362850.
Model 414000 of 1359985.
Model 414750 of 1349985.
Model 415000 of 1346350.
Model 416000 of 1338600.
Model 416925 of 1328600.
Model 417000 of 1328350.
Model 417625 of 1318350.
Model 418000 of 1316850.
Model 418925 of 1306850.
Model 419000 of 1306600.
Model 419925 of 1296600.
Model 420000 of 1296350.
Model 420925 of 1286350.
Model 421000 of 1286100.
Model 421750 of 1276100.
Model 422000 of 1272600.
Model 422825 of 1262600.
Model 423000 of 1259976.
Model 423775 of 1249976.
Model 424000 of 1246100.
Model 424850 of 1236100.
Model 425000 of 1231850.
Model 425925 of 1221850.
Model 426000 of 1220100.
Model 426625 of 1210100.
Model 427000 of 1208350.
Model 428000 of 1203250.
Model 429000 of 1201250.
```

```
Model 430000 of 1199250.
Model 431000 of 1198000.
Model 432000 of 1198000.
Model 433000 of 1195500.
Model 434000 of 1192435.
Model 435000 of 1190630.
Model 436000 of 1187750.
Model 437000 of 1187750.
Model 438000 of 1185250.
Model 439000 of 1182750.
Model 440000 of 1180750.
Model 441000 of 1178750.
Model 442000 of 1177500.
Model 443000 of 1177500.
Model 444000 of 1172500.
Model 445000 of 1170500.
Model 446000 of 1169755.
Model 447000 of 1167250.
Model 448000 of 1167250.
Model 449000 of 1164750.
Model 450000 of 1161880.
Model 451000 of 1160060.
Model 452000 of 1158250.
Model 453000 of 1157000.
Model 454000 of 1154100.
Model 454575 of 1144100.
Model 455000 of 1139850.
Model 455650 of 1129850.
Model 456000 of 1128100.
Model 456580 of 1118100.
Model 457000 of 1113850.
Model 457635 of 1103850.
Model 458000 of 1099600.
Model 458680 of 1089600.
Model 459000 of 1085350.
Model 459925 of 1075350.
Model 460000 of 1075100.
Model 460750 of 1065100.
Model 461000 of 1063600.
Model 462000 of 1055850.
Model 462875 of 1045850.
Model 463000 of 1044350.
Model 463875 of 1034350.
Model 464000 of 1032850.
Model 464875 of 1022850.
Model 465000 of 1021350.
Model 466000 of 1013600.
Model 466650 of 1003600.
Model 467000 of 999350.
Model 467695 of 989350.
Model 468000 of 985100.
Model 468725 of 975100.
Model 469000 of 970850.
Model 469725 of 960850.
Model 470000 of 957850.
Model 470765 of 947850.
Model 471000 of 944851.
Model 472000 of 937750.
Model 473000 of 934880.
Model 474000 of 933060.
Model 475000 of 931250.
Model 476000 of 930000.
Model 477000 of 927500.
Model 478000 of 925000.
Model 479000 of 923000.
Model 480000 of 921000.
Model 481000 of 919750.
```

```
Model 482000 of 919750.
Model 483000 of 914750.
Model 484000 of 914005.
Model 485000 of 912185.
Model 486000 of 909500.
Model 487000 of 909500.
Model 488000 of 907000.
Model 489000 of 904310.
Model 490000 of 902500.
Model 491000 of 900500.
Model 492000 of 899250.
Model 493000 of 899250.
Model 494000 of 894250.
Model 495000 of 892250.
Model 496000 of 890250.
Model 497000 of 889000.
Model 498000 of 889000.
Model 499000 of 886500.
Model 500000 of 884000.
Model 501000 of 882750.
Model 502000 of 882750.
Model 503000 of 881500.
Model 504000 of 880250.
Model 505000 of 880250.
Model 506000 of 875250.
Model 507000 of 875250.
Model 508000 of 874000.
Model 509000 of 874000.
Model 510000 of 872750.
Model 511000 of 871500.
Model 512000 of 871500.
Model 513000 of 866500.
Model 514000 of 866500.
Model 515000 of 865250.
Model 516000 of 865250.
Model 517000 of 864000.
Model 518000 of 862750.
Model 519000 of 862750.
Model 520000 of 857750.
Model 521000 of 857750.
Model 522000 of 856500.
Model 523000 of 856500.
Model 524000 of 855250.
Model 525000 of 854000.
Model 526000 of 851500.
Model 527000 of 849000.
Model 528000 of 849000.
Model 529000 of 847750.
Model 530000 of 847750.
Model 531000 of 845250.
Model 532000 of 845250.
Model 533000 of 842750.
Model 534000 of 840250.
Model 535000 of 838250.
Model 536000 of 836250.
Model 537000 of 835000.
Model 538000 of 835000.
Model 539000 of 830000.
Model 540000 of 828000.
Model 541000 of 827251.
Model 542000 of 824750.
Model 543000 of 824750.
Model 544000 of 822250.
Model 545000 of 819376.
Model 546000 of 817562.
Model 547000 of 815750.
Model 548000 of 814500.
```

```
Model 549000 of 812000.
Model 550000 of 809500.
Model 551000 of 807500.
Model 552000 of 805500.
Model 553000 of 804250.
Model 554000 of 804250.
Model 555000 of 799250.
Model 556000 of 798501.
Model 557000 of 796687.
Model 558000 of 794000.
Model 559000 of 794000.
Model 560000 of 787520.
Model 561000 of 779770.
Model 561875 of 769770.
Model 562000 of 768270.
Model 562875 of 758270.
Model 563000 of 756770.
Model 563875 of 746770.
Model 564000 of 745270.
Model 565000 of 737520.
Model 565935 of 727520.
Model 566000 of 727270.
Model 566625 of 717270.
Model 567000 of 713572.
Model 567700 of 703572.
Model 568000 of 699520.
Model 568775 of 689520.
Model 569000 of 686521.
Model 569745 of 676521.
Model 570000 of 672822.
Model 570800 of 662822.
Model 571000 of 658770.
Model 571875 of 648770.
Model 572000 of 647020.
Model 572935 of 637020.
Model 573000 of 636770.
Model 573935 of 626770.
Model 574000 of 626520.
Model 574750 of 616520.
Model 575000 of 615020.
Model 576000 of 607270.
Model 576875 of 597270.
Model 577000 of 595770.
Model 577875 of 585770.
Model 578000 of 584270.
Model 578750 of 578750. Skipped 2233750 iterations.
578750 models discarded due to low R² or negative coefficients.
0 Models met minimum requirements.
```

Out[23]:

Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef	Norm Bed Coef	Norm Bath Coef

There were no seven variable models that met the requirements.

In [26]:

```
mega8 = mega_models(model_data,8,8)
mega8 = mega8.insert(1,'Var Count',8)

mega8.to_excel("mega8.xlsx")
```

```
Combos of 8 variables (1/1).
Model 0 of 3515625.
Model 0 of 3505625.
Model 0 of 3495625.
```

```
Model 0 of 3485625.
Model 0 of 3475625.
Model 0 of 3465625.
Model 0 of 3455625.
Model 1000 of 3449625.
Model 2000 of 3445625.
Model 3000 of 3441625.
Model 3125 of 3431625.
Model 3125 of 3421625.
Model 3125 of 3411625.
Model 3125 of 3401625.
Model 3125 of 3391625.
Model 3125 of 3381625.
Model 4000 of 3375125.
Model 5000 of 3371125.
Model 6000 of 3367125.
Model 7000 of 3357505.
Model 7850 of 3347505.
Model 8000 of 3346005.
Model 8750 of 3336005.
Model 8750 of 3326005.
Model 8750 of 3316005.
Model 8750 of 3306005.
Model 9000 of 3305125.
Model 10000 of 3301125.
Model 11000 of 3297125.
Model 12000 of 3291935.
Model 12875 of 3281935.
Model 13000 of 3280435.
Model 13875 of 3270435.
Model 14000 of 3268935.
Model 14375 of 3258935.
Model 14375 of 3248935.
Model 14375 of 3238935.
Model 15000 of 3231125.
Model 16000 of 3227125.
Model 17000 of 3223125.
Model 17500 of 3213125.
Model 17500 of 3203125.
Model 17500 of 3193125.
Model 18000 of 3187875.
Model 19000 of 3183875.
Model 20000 of 3179875.
Model 21000 of 3175875.
Model 22000 of 3171875.
Model 23000 of 3167875.
Model 24000 of 3163875.
Model 25000 of 3159875.
Model 26000 of 3155875.
Model 27000 of 3150685.
Model 27875 of 3140685.
Model 28000 of 3139185.
Model 28875 of 3129185.
Model 29000 of 3127685.
Model 30000 of 3121125.
Model 31000 of 3117125.
Model 32000 of 3113125.
Model 33000 of 3109125.
Model 34000 of 3105125.
Model 35000 of 3101125.
Model 36000 of 3097125.
Model 37000 of 3093125.
Model 38000 of 3089125.
Model 39000 of 3083255.
Model 39850 of 3073255.
Model 40000 of 3071755.
Model 40850 of 3061755.
```

```
Model 41000 of 3060255.
Model 42000 of 3054375.
Model 43000 of 3050375.
Model 44000 of 3046375.
Model 45000 of 3042375.
Model 46000 of 3038375.
Model 47000 of 3034375.
Model 48000 of 3030375.
Model 49000 of 3026375.
Model 50000 of 3022375.
Model 51000 of 3015310.
Model 51875 of 3005310.
Model 52000 of 3003810.
Model 52875 of 2993810.
Model 53000 of 2992310.
Model 54000 of 2987625.
Model 55000 of 2983625.
Model 56000 of 2979625.
Model 57000 of 2975625.
Model 58000 of 2971625.
Model 59000 of 2967625.
Model 60000 of 2963625.
Model 61000 of 2959625.
Model 62000 of 2955625.
Model 63000 of 2947880.
Model 63850 of 2937880.
Model 64000 of 2936380.
Model 64850 of 2926380.
Model 65000 of 2924880.
Model 66000 of 2920875.
Model 67000 of 2916875.
Model 68000 of 2912875.
Model 69000 of 2908875.
Model 70000 of 2904875.
Model 71000 of 2900875.
Model 72000 of 2896875.
Model 73000 of 2892875.
Model 74000 of 2888875.
Model 75000 of 2879935.
Model 75875 of 2869935.
Model 76000 of 2868435.
Model 76875 of 2858435.
Model 77000 of 2858125.
Model 78000 of 2854125.
Model 79000 of 2850125.
Model 80000 of 2846125.
Model 81000 of 2842125.
Model 82000 of 2838125.
Model 83000 of 2834125.
Model 84000 of 2830125.
Model 85000 of 2826125.
Model 86000 of 2822125.
Model 87000 of 2818125.
Model 88000 of 2814125.
Model 89000 of 2810125.
Model 90000 of 2806125.
Model 91000 of 2802125.
Model 92000 of 2798125.
Model 93000 of 2794125.
Model 94000 of 2790125.
Model 95000 of 2786125.
Model 96000 of 2782125.
Model 97000 of 2778125.
Model 98000 of 2774125.
Model 99000 of 2770125.
Model 100000 of 2766125.
Model 101000 of 2762125.
```

```
Model 102000 of 2758125.
Model 103000 of 2754125.
Model 104000 of 2750125.
Model 105000 of 2746125.
Model 106000 of 2742125.
Model 107000 of 2738125.
Model 108000 of 2734125.
Model 109000 of 2730125.
Model 110000 of 2726125.
Model 111000 of 2722125.
Model 112000 of 2718125.
Model 113000 of 2714125.
Model 114000 of 2710125.
Model 115000 of 2706125.
Model 116000 of 2702125.
Model 117000 of 2698125.
Model 118000 of 2694125.
Model 119000 of 2690125.
Model 120000 of 2686125.
Model 121000 of 2682125.
Model 122000 of 2678125.
Model 123000 of 2674125.
Model 124000 of 2670125.
Model 125000 of 2666125.
Model 126000 of 2662125.
Model 127000 of 2658125.
Model 128000 of 2654125.
Model 129000 of 2650125.
Model 130000 of 2646125.
Model 131000 of 2642125.
Model 132000 of 2638125.
Model 133000 of 2634125.
Model 134000 of 2630125.
Model 135000 of 2626125.
Model 136000 of 2622125.
Model 137000 of 2618125.
Model 138000 of 2614125.
Model 139000 of 2610125.
Model 140000 of 2606125.
Model 141000 of 2602125.
Model 142000 of 2598125.
Model 143000 of 2594125.
Model 144000 of 2590125.
Model 145000 of 2586125.
Model 146000 of 2582125.
Model 147000 of 2578125.
Model 148000 of 2574125.
Model 149000 of 2570125.
Model 150000 of 2566125.
Model 151000 of 2562125.
Model 152000 of 2558125.
Model 153000 of 2554125.
Model 154000 of 2550125.
Model 155000 of 2546125.
Model 156000 of 2542125.
Model 157000 of 2538125.
Model 158000 of 2534125.
Model 159000 of 2530125.
Model 160000 of 2526125.
Model 161000 of 2522125.
Model 162000 of 2518125.
Model 163000 of 2514125.
Model 164000 of 2510125.
Model 164375 of 2500125.
Model 165000 of 2493625.
Model 165000 of 2483625.
Model 166000 of 2474060.
```

```
Model 166750 of 2464060.
Model 167000 of 2461255.
Model 167375 of 2451255.
Model 168000 of 2442875.
Model 169000 of 2438875.
Model 169750 of 2428875.
Model 170000 of 2422375.
Model 170375 of 2412375.
Model 171000 of 2405875.
Model 171500 of 2395875.
Model 172000 of 2391875.
Model 172625 of 2381875.
Model 173000 of 2377875.
Model 173875 of 2367875.
Model 174000 of 2367625.
Model 175000 of 2363625.
Model 175125 of 2353625.
Model 175750 of 2343625.
Model 176000 of 2334625.
Model 176875 of 2324625.
Model 177000 of 2320625.
Model 178000 of 2312880.
Model 178625 of 2302880.
Model 179000 of 2296375.
Model 180000 of 2292375.
Model 180500 of 2282375.
Model 181000 of 2275875.
Model 181125 of 2265875.
Model 182000 of 2257505.
Model 182375 of 2247505.
Model 183000 of 2244185.
Model 183375 of 2234185.
Model 184000 of 2231375.
Model 184875 of 2221375.
Model 185000 of 2221125.
Model 185875 of 2211125.
Model 186000 of 2204625.
Model 186500 of 2194625.
Model 187000 of 2188125.
Model 187625 of 2178125.
Model 188000 of 2174125.
Model 188750 of 2164125.
Model 189000 of 2160125.
Model 189875 of 2150125.
Model 190000 of 2149875.
Model 191000 of 2145875.
Model 191250 of 2135875.
Model 191875 of 2125875.
Model 192000 of 2116875.
Model 193000 of 2109130.
Model 193725 of 2099130.
Model 194000 of 2095810.
Model 194750 of 2085810.
Model 195000 of 2078625.
Model 196000 of 2074625.
Model 196625 of 2064625.
Model 197000 of 2058125.
Model 197250 of 2048125.
Model 198000 of 2040435.
Model 198375 of 2030435.
Model 199000 of 2027625.
Model 199500 of 2017625.
Model 200000 of 2013625.
Model 200875 of 2003625.
Model 201000 of 2003375.
Model 202000 of 1999375.
Model 202000 of 1989375.
```

```
Model 202625 of 1979375.
Model 203000 of 1970375.
Model 203750 of 1960375.
Model 204000 of 1956375.
Model 204875 of 1946375.
Model 205000 of 1942375.
Model 205875 of 1932375.
Model 206000 of 1932125.
Model 207000 of 1928125.
Model 207375 of 1918125.
Model 208000 of 1911625.
Model 208000 of 1901625.
Model 209000 of 1892060.
Model 209750 of 1882060.
Model 210000 of 1879255.
Model 210375 of 1869255.
Model 211000 of 1860875.
Model 212000 of 1856875.
Model 212750 of 1846875.
Model 213000 of 1840375.
Model 213375 of 1830375.
Model 214000 of 1823875.
Model 214500 of 1813875.
Model 215000 of 1809875.
Model 215625 of 1799875.
Model 216000 of 1795875.
Model 216875 of 1785875.
Model 217000 of 1785625.
Model 218000 of 1781625.
Model 218125 of 1771625.
Model 219000 of 1768125.
Model 220000 of 1768125.
Model 221000 of 1768125.
Model 221250 of 1758125.
Model 222000 of 1755625.
Model 223000 of 1755625.
Model 224000 of 1755625.
Model 225000 of 1754685.
Model 226000 of 1753185.
Model 227000 of 1745625.
Model 228000 of 1745625.
Model 229000 of 1745625.
Model 230000 of 1745625.
Model 231000 of 1744130.
Model 232000 of 1742630.
Model 233000 of 1735625.
Model 234000 of 1735625.
Model 235000 of 1735625.
Model 236000 of 1729375.
Model 237000 of 1729375.
Model 238000 of 1729375.
Model 239000 of 1729375.
Model 240000 of 1729375.
Model 241000 of 1729375.
Model 242000 of 1729375.
Model 243000 of 1729375.
Model 244000 of 1729375.
Model 245000 of 1729375.
Model 245000 of 1719375.
Model 246000 of 1716875.
Model 247000 of 1716875.
Model 248000 of 1716875.
Model 248125 of 1706875.
Model 249000 of 1704375.
Model 250000 of 1704375.
Model 251000 of 1704375.
Model 252000 of 1703255.
```

```
Model 253000 of 1701755.
Model 254000 of 1694375.
Model 255000 of 1694375.
Model 256000 of 1694375.
Model 257000 of 1694185.
Model 258000 of 1692685.
Model 259000 of 1691185.
Model 260000 of 1684375.
Model 261000 of 1684375.
Model 262000 of 1684375.
Model 263000 of 1678125.
Model 264000 of 1678125.
Model 265000 of 1678125.
Model 266000 of 1678125.
Model 267000 of 1678125.
Model 268000 of 1678125.
Model 269000 of 1678125.
Model 270000 of 1678125.
Model 271000 of 1678125.
Model 271875 of 1668125.
Model 272000 of 1665625.
Model 273000 of 1665625.
Model 274000 of 1665625.
Model 275000 of 1665625.
Model 275000 of 1655625.
Model 276000 of 1653125.
Model 277000 of 1653125.
Model 278000 of 1653125.
Model 279000 of 1651810.
Model 280000 of 1650310.
Model 281000 of 1643125.
Model 282000 of 1643125.
Model 283000 of 1643125.
Model 284000 of 1642755.
Model 285000 of 1641255.
Model 286000 of 1639755.
Model 287000 of 1633125.
Model 288000 of 1633125.
Model 289000 of 1633125.
Model 290000 of 1626875.
Model 291000 of 1626875.
Model 292000 of 1626875.
Model 293000 of 1626875.
Model 294000 of 1626875.
Model 295000 of 1626875.
Model 296000 of 1626875.
Model 297000 of 1626875.
Model 298000 of 1626875.
Model 298750 of 1616875.
Model 299000 of 1614375.
Model 300000 of 1614375.
Model 301000 of 1614375.
Model 301875 of 1604375.
Model 302000 of 1601875.
Model 303000 of 1601875.
Model 304000 of 1601875.
Model 305000 of 1601875.
Model 306000 of 1600380.
Model 307000 of 1598880.
Model 308000 of 1591875.
Model 309000 of 1591875.
Model 310000 of 1591875.
Model 311000 of 1591310.
Model 312000 of 1589810.
Model 313000 of 1588310.
Model 314000 of 1581875.
Model 315000 of 1581875.
```

```
Model 316000 of 1581875.
Model 317000 of 1575625.
Model 318000 of 1575625.
Model 319000 of 1575625.
Model 320000 of 1575625.
Model 321000 of 1575625.
Model 322000 of 1575625.
Model 323000 of 1575625.
Model 324000 of 1575625.
Model 325000 of 1575625.
Model 325625 of 1565625.
Model 326000 of 1563125.
Model 327000 of 1563125.
Model 328000 of 1563125.
Model 328750 of 1553125.
Model 329000 of 1550625.
Model 330000 of 1550625.
Model 331000 of 1550625.
Model 332000 of 1550435.
Model 333000 of 1548935.
Model 334000 of 1547435.
Model 335000 of 1540625.
Model 336000 of 1540625.
Model 337000 of 1540625.
Model 338000 of 1539880.
Model 339000 of 1538380.
Model 340000 of 1536880.
Model 341000 of 1530625.
Model 342000 of 1530625.
Model 343000 of 1530625.
Model 344000 of 1524375.
Model 345000 of 1524375.
Model 346000 of 1524375.
Model 347000 of 1524375.
Model 348000 of 1524375.
Model 349000 of 1524375.
Model 350000 of 1524375.
Model 351000 of 1524375.
Model 352000 of 1524375.
Model 352500 of 1514375.
Model 353000 of 1509975.
Model 353125 of 1499975.
Model 354000 of 1491605.
Model 354450 of 1481605.
Model 355000 of 1478485.
Model 355375 of 1468485.
Model 356000 of 1465475.
Model 356925 of 1455475.
Model 357000 of 1455225.
Model 357875 of 1445225.
Model 358000 of 1438725.
Model 358500 of 1428725.
Model 359000 of 1422225.
Model 359625 of 1412225.
Model 360000 of 1408225.
Model 360750 of 1398225.
Model 361000 of 1394225.
Model 361925 of 1384225.
Model 362000 of 1383975.
Model 363000 of 1379975.
Model 363250 of 1369975.
Model 363875 of 1359975.
Model 364000 of 1350975.
Model 365000 of 1343230.
Model 365725 of 1333230.
Model 366000 of 1330110.
Model 366700 of 1320110.
```

```
Model 367000 of 1312725.
Model 368000 of 1308725.
Model 368625 of 1298725.
Model 369000 of 1292225.
Model 369250 of 1282225.
Model 370000 of 1274735.
Model 370375 of 1264735.
Model 371000 of 1261725.
Model 371500 of 1251725.
Model 372000 of 1247725.
Model 372925 of 1237725.
Model 373000 of 1237475.
Model 374000 of 1233475.
Model 374000 of 1223475.
Model 374625 of 1213475.
Model 375000 of 1204475.
Model 375750 of 1194475.
Model 376000 of 1190475.
Model 376875 of 1180475.
Model 377000 of 1176475.
Model 377925 of 1166475.
Model 378000 of 1166225.
Model 379000 of 1162225.
Model 379375 of 1152225.
Model 380000 of 1145725.
Model 380000 of 1135725.
Model 381000 of 1129225.
Model 381925 of 1119225.
Model 382000 of 1118975.
Model 383000 of 1114975.
Model 383925 of 1104975.
Model 384000 of 1104725.
Model 384925 of 1094725.
Model 385000 of 1094475.
Model 386000 of 1090475.
Model 386250 of 1080475.
Model 386875 of 1070475.
Model 387000 of 1061475.
Model 388000 of 1057475.
Model 388925 of 1047475.
Model 389000 of 1047225.
Model 390000 of 1043225.
Model 390925 of 1033225.
Model 391000 of 1032975.
Model 391925 of 1022975.
Model 392000 of 1022725.
Model 393000 of 1018725.
Model 393125 of 1008725.
Model 393750 of 998725.
Model 394000 of 989725.
Model 395000 of 985725.
Model 395925 of 975725.
Model 396000 of 975475.
Model 397000 of 971475.
Model 397925 of 961475.
Model 398000 of 961225.
Model 398925 of 951225.
Model 399000 of 950975.
Model 400000 of 946975.
Model 400000 of 936975.
Model 400625 of 926975.
Model 401000 of 917975.
Model 402000 of 913975.
Model 402925 of 903975.
Model 403000 of 903725.
Model 404000 of 899725.
Model 404925 of 889725.
```

```
Model 405000 of 889475.
Model 405925 of 879475.
Model 406000 of 879225.
Model 406875 of 869225.
Model 407000 of 862725.
Model 407500 of 852725.
Model 408000 of 846225.
Model 409000 of 842225.
Model 409925 of 832225.
Model 410000 of 831975.
Model 411000 of 827975.
Model 411875 of 817975.
Model 412000 of 811475.
Model 413000 of 807475.
Model 413750 of 797475.
Model 414000 of 790975.
Model 414375 of 780975.
Model 415000 of 774475.
Model 415500 of 764475.
Model 416000 of 760475.
Model 416625 of 750475.
Model 417000 of 746475.
Model 417925 of 736475.
Model 418000 of 736225.
Model 419000 of 732225.
Model 419125 of 722225.
Model 419750 of 712225.
Model 420000 of 703225.
Model 420875 of 693225.
Model 421000 of 689225.
Model 422000 of 681476.
Model 422625 of 671476.
Model 423000 of 664975.
Model 424000 of 660975.
Model 424500 of 650975.
Model 425000 of 644475.
Model 425125 of 634475.
Model 426000 of 626101.
Model 426450 of 616101.
Model 427000 of 612987.
Model 427375 of 602987.
Model 428000 of 599975.
Model 428925 of 589975.
Model 429000 of 589725.
Model 429875 of 579725.
Model 430000 of 573225.
Model 430500 of 563225.
Model 431000 of 556725.
Model 431625 of 546725.
Model 432000 of 542725.
Model 432750 of 532725.
Model 433000 of 528725.
Model 433925 of 518725.
Model 434000 of 518475.
Model 435000 of 514475.
Model 435250 of 504475.
Model 435875 of 494475.
Model 436000 of 485475.
Model 437000 of 477726.
Model 437725 of 467726.
Model 438000 of 464612.
Model 438700 of 454612.
Model 439000 of 447225.
Model 440000 of 443225.
Model 440625 of 440625. Skipped 3075000 iterations.
440625 models discarded due to low R² or negative coefficients.
0 Models met minimum requirements.
```

There were no eight variable models that met the requirements.

In [28]:
```
mega9 = mega_models(model_data,9,9)
mega9
```

```
Combos of 9 variables (1/1).
Model 0 of 1953125.
Model 0 of 1943125.
Model 0 of 1933125.
Model 0 of 1923125.
Model 0 of 1913125.
Model 0 of 1903125.
Model 0 of 1893125.
Model 1000 of 1887125.
Model 2000 of 1883125.
Model 3000 of 1879125.
Model 3125 of 1869125.
Model 3125 of 1859125.
Model 3125 of 1849125.
Model 3125 of 1839125.
Model 3125 of 1829125.
Model 3125 of 1819125.
Model 4000 of 1812625.
Model 5000 of 1808625.
Model 6000 of 1804625.
Model 7000 of 1795005.
Model 7850 of 1785005.
Model 8000 of 1783505.
Model 8750 of 1773505.
Model 8750 of 1763505.
Model 8750 of 1753505.
Model 8750 of 1743505.
Model 9000 of 1742625.
Model 10000 of 1738625.
Model 11000 of 1734625.
Model 12000 of 1729435.
Model 12875 of 1719435.
Model 13000 of 1717935.
Model 13875 of 1707935.
Model 14000 of 1706435.
Model 14375 of 1696435.
Model 14375 of 1686435.
Model 14375 of 1676435.
Model 15000 of 1668625.
Model 16000 of 1664625.
Model 17000 of 1660625.
Model 17500 of 1650625.
Model 17500 of 1640625.
Model 17500 of 1630625.
Model 18000 of 1625375.
Model 19000 of 1621375.
Model 20000 of 1617375.
Model 21000 of 1613375.
Model 22000 of 1609375.
Model 23000 of 1605375.
Model 24000 of 1601375.
Model 25000 of 1597375.
Model 26000 of 1593375.
Model 26875 of 1583375.
Model 26875 of 1573375.
Model 26875 of 1563375.
Model 26875 of 1553375.
Model 26875 of 1543375.
Model 26875 of 1533375.
Model 27000 of 1526875.
Model 28000 of 1522875.
Model 29000 of 1518875.
```

```
Model 30000 of 1514875.
Model 30000 of 1504875.
Model 30000 of 1494875.
Model 30000 of 1484875.
Model 30000 of 1474875.
Model 30000 of 1464875.
Model 30000 of 1454875.
Model 31000 of 1448375.
Model 32000 of 1444375.
Model 33000 of 1440375.
Model 33925 of 1430375.
Model 34000 of 1429560.
Model 34875 of 1419560.
Model 35000 of 1418060.
Model 35625 of 1408060.
Model 35625 of 1398060.
Model 35625 of 1388060.
Model 36000 of 1378375.
Model 37000 of 1374375.
Model 38000 of 1370375.
Model 39000 of 1364505.
Model 39850 of 1354505.
Model 40000 of 1353005.
Model 40850 of 1343005.
Model 41000 of 1341505.
Model 41250 of 1331505.
Model 41250 of 1321505.
Model 41250 of 1311505.
Model 42000 of 1304375.
Model 43000 of 1300375.
Model 44000 of 1296375.
Model 44375 of 1286375.
Model 44375 of 1276375.
Model 44375 of 1266375.
Model 45000 of 1261125.
Model 46000 of 1257125.
Model 47000 of 1253125.
Model 48000 of 1249125.
Model 49000 of 1245125.
Model 50000 of 1241125.
Model 51000 of 1237125.
Model 52000 of 1233125.
Model 53000 of 1229125.
Model 53750 of 1219125.
Model 53750 of 1209125.
Model 53750 of 1199125.
Model 53750 of 1189125.
Model 53750 of 1179125.
Model 53750 of 1169125.
Model 54000 of 1162625.
Model 55000 of 1158625.
Model 56000 of 1154625.
Model 56875 of 1144625.
Model 56875 of 1134625.
Model 56875 of 1124625.
Model 56875 of 1114625.
Model 56875 of 1104625.
Model 56875 of 1094625.
Model 57000 of 1088125.
Model 58000 of 1084125.
Model 59000 of 1080125.
Model 60000 of 1076125.
Model 60850 of 1066125.
Model 61000 of 1064630.
Model 61850 of 1054630.
Model 62000 of 1053130.
Model 62500 of 1043130.
```

```
Model 62500 of 1033130.
Model 62500 of 1023130.
Model 63000 of 1014125.
Model 64000 of 1010125.
Model 65000 of 1006125.
Model 66000 of 999060.
Model 66875 of 989060.
Model 67000 of 987560.
Model 67875 of 977560.
Model 68000 of 976060.
Model 68125 of 966060.
Model 68125 of 956060.
Model 68125 of 946060.
Model 69000 of 940125.
Model 70000 of 936125.
Model 71000 of 932125.
Model 71250 of 922125.
Model 71250 of 912125.
Model 71250 of 902125.
Model 72000 of 896875.
Model 73000 of 892875.
Model 74000 of 888875.
Model 75000 of 884875.
Model 76000 of 880875.
Model 77000 of 876875.
Model 78000 of 872875.
Model 79000 of 868875.
Model 80000 of 864875.
Model 80625 of 854875.
Model 80625 of 844875.
Model 80625 of 834875.
Model 80625 of 824875.
Model 80625 of 814875.
Model 80625 of 804875.
Model 81000 of 798375.
Model 82000 of 794375.
Model 83000 of 790375.
Model 83750 of 780375.
Model 83750 of 770375.
Model 83750 of 760375.
Model 83750 of 750375.
Model 83750 of 740375.
Model 83750 of 730375.
Model 84000 of 723875.
Model 85000 of 719875.
Model 86000 of 715875.
Model 87000 of 710685.
Model 87875 of 700685.
Model 88000 of 699185.
Model 88875 of 689185.
Model 89000 of 687685.
Model 89375 of 677685.
Model 89375 of 667685.
Model 89375 of 657685.
Model 90000 of 649875.
Model 91000 of 645875.
Model 92000 of 641875.
Model 93000 of 634130.
Model 93850 of 624130.
Model 94000 of 622630.
Model 94850 of 612630.
Model 95000 of 611130.
Model 95000 of 601130.
Model 95000 of 591130.
Model 95000 of 581130.
Model 96000 of 575875.
Model 97000 of 571875.
```

```
Model 98000 of 567875.
Model 98125 of 557875.
Model 98125 of 547875.
Model 98125 of 537875.
Model 99000 of 532625.
Model 100000 of 528625.
Model 101000 of 524625.
Model 102000 of 520625.
Model 103000 of 516625.
Model 104000 of 512625.
Model 105000 of 508625.
Model 106000 of 504625.
Model 107000 of 500625.
Model 107500 of 490625.
Model 107500 of 480625.
Model 107500 of 470625.
Model 107500 of 460625.
Model 107500 of 450625.
Model 107500 of 440625.
Model 108000 of 434125.
Model 109000 of 430125.
Model 110000 of 426125.
Model 110625 of 416125.
Model 110625 of 406125.
Model 110625 of 396125.
Model 110625 of 386125.
Model 110625 of 376125.
Model 110625 of 366125.
Model 111000 of 359625.
Model 112000 of 355625.
Model 113000 of 351625.
Model 114000 of 345755.
Model 114850 of 335755.
Model 115000 of 334255.
Model 115850 of 324255.
Model 116000 of 322755.
Model 116250 of 312755.
Model 116250 of 302755.
Model 116250 of 292755.
Model 117000 of 285625.
Model 118000 of 281625.
Model 119000 of 277625.
Model 120000 of 268685.
Model 120875 of 258685.
Model 121000 of 257185.
Model 121875 of 247185.
Model 121875 of 237185.
Model 121875 of 227185.
Model 121875 of 217185.
Model 122000 of 215625.
Model 123000 of 211625.
Model 124000 of 207625.
Model 125000 of 203625.
Model 125000 of 193625.
Model 125000 of 183625.
Model 125000 of 173625.
Model 126000 of 168375.
Model 127000 of 164375.
Model 128000 of 160375.
Model 129000 of 156375.
Model 130000 of 152375.
Model 131000 of 148375.
Model 132000 of 144375.
Model 133000 of 140375.
Model 134000 of 136375.
Model 134375 of 134375. Skipped 1818750 iterations.
```

```
134375 models discarded due to low R² or negative coefficients.
0 Models met minimum requirements.
```

Out[28]:

Model Variable(s)	R^2	P-value	Neg Coef?	Intercept	Bed Coef	Bath Coef	SF Liv Coef	SF Lot Coef	Floors Coef	WF Coef	Cond Coef	Grade Coef	Yr Coef	Norm Bed Coef	Norm Bath Coef

There were no nine variable models that met the requirements.

In [ ]:

```
This puts all the data together and then creates the excel file.
mega = mega1.append([mega2,mega3,mega4,mega5,mega6])
mega = mega.sort_values('R²',ascending=False)
mega = mega.reset_index(drop=True)
```

In [ ]:

```
mega.to_excel("Mega Pertinent Stats.xlsx")
```