House Prices Using Backward Elimination

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline
```

In [3]:
```
dataset = pd.read_csv(r"D:\NIT Daily Task\Sep\26th- mlr\26th- mlr\MLR\House_data
dataset
```

Out[3]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7
2	5631500400	20150225T000000	180000.0	2	1.00	770	10
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8
...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1

21613 rows × 21 columns

In [5]:
```
dataset.head()
```

Out[5]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080

5 rows × 21 columns

In [7]:
```
print(dataset.isnull().any())
```

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

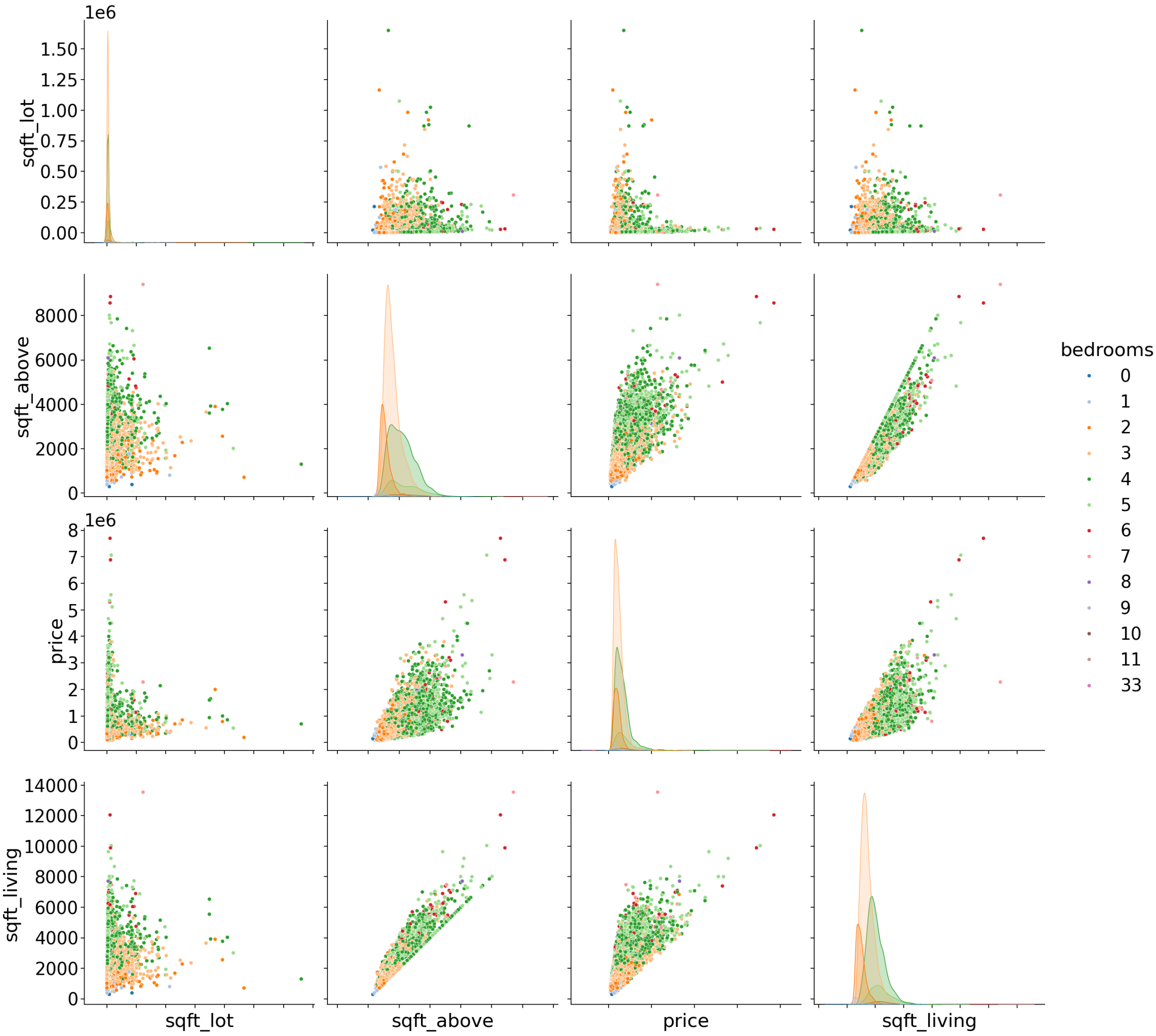
In [9]:
```
print(dataset.dtypes)
```

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

In [11]:
```
dataset = dataset.drop(['id', 'date'], axis = 1)
```

In [13]:
```
with sns.plotting_context("notebook",font_scale=2.5):
    g = sns.pairplot(dataset[['sqft_lot','sqft_above','price','sqft_living','bed
                 hue='bedrooms', palette='tab20',size=6)
g.set(xticklabels=[]);
```

```
C:\Users\chitt\anaconda3\Lib\site-packages\seaborn\axisgrid.py:2100: UserWarning:
The `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```



In [17]:
```
#separating independent and dependent variable
X = dataset.iloc[:,1:].values
y = dataset.iloc[:,0].values
#splitting dataset into training and testing dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, randc
```

In [19]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

In [21]:
```
import statsmodels.api as sm
def backwardElimination(x, SL):
    numVars = len(x[0])
    temp = np.zeros((21613,19)).astype(int)
    for i in range(0, numVars):
        regressor_OLS = sm.OLS(y, x).fit()
        maxVar = max(regressor_OLS.pvalues).astype(float)
        adjR_before = regressor_OLS.rsquared_adj.astype(float)
        if maxVar > SL:
            for j in range(0, numVars - i):
                if (regressor_OLS.pvalues[j].astype(float) == maxVar):
                    temp[:,j] = x[:, j]
                    x = np.delete(x, j, 1)
                    tmp_regressor = sm.OLS(y, x).fit()
```

```
                    adjR_after = tmp_regressor.rsquared_adj.astype(float)
                    if (adjR_before >= adjR_after):
                        x_rollback = np.hstack((x, temp[:,[0,j]]))
                        x_rollback = np.delete(x_rollback, j, 1)
                        print (regressor_OLS.summary())
                        return x_rollback
                    else:
                        continue
    regressor_OLS.summary()
    return x

SL = 0.05
X_opt = X[:, [0, 1, 2, 3, 4, 5,6,7,8,9,10,11,12,13,14,15,16,17]]
X_Modeled = backwardElimination(X_opt, SL)
```

```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                      y   R-squared (uncentered):                   0.905
Model:                            OLS   Adj. R-squared (uncentered):              0.905
Method:                 Least Squares   F-statistic:                          1.211e+04
Date:                Fri, 27 Sep 2024   Prob (F-statistic):                        0.00
Time:                        08:46:27   Log-Likelihood:                     -2.9461e+05
No. Observations:               21613   AIC:                                  5.892e+05
Df Residuals:                   21596   BIC:                                  5.894e+05
Df Model:                          17
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1         -3.551e+04   1888.716    -18.802      0.000   -3.92e+04   -3.18e+04
x2          4.105e+04   3253.759     12.618      0.000    3.47e+04    4.74e+04
x3           110.2642      2.268     48.607      0.000     105.818     114.711
x4             0.1334      0.048      2.786      0.005       0.040       0.227
x5          5261.5471   3541.347      1.486      0.137   -1679.755    1.22e+04
x6          5.833e+05   1.74e+04     33.598      0.000    5.49e+05    6.17e+05
x7          5.236e+04   2128.298     24.600      0.000    4.82e+04    5.65e+04
x8          2.721e+04   2323.818     11.709      0.000    2.27e+04    3.18e+04
x9          9.548e+04   2145.492     44.503      0.000    9.13e+04    9.97e+04
x10           71.3928      2.238     31.902      0.000      67.006      75.779
x11           38.8714      2.624     14.813      0.000      33.728      44.015
x12        -2561.7953     68.006    -37.670      0.000   -2695.092   -2428.498
x13           20.4187      3.646      5.600      0.000      13.272      27.566
x14         -519.0756     17.826    -29.119      0.000    -554.016    -484.136
x15         6.022e+05   1.07e+04     56.106      0.000    5.81e+05    6.23e+05
x16        -2.179e+05   1.31e+04    -16.683      0.000   -2.44e+05   -1.92e+05
x17           23.0994      3.392      6.811      0.000      16.452      29.747
x18           -0.3761      0.073     -5.137      0.000      -0.520      -0.233
==============================================================================
Omnibus:                    18403.146   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1873534.498
Skew:                           3.572   Prob(JB):                         0.00
Kurtosis:                      48.049   Cond. No.                     4.88e+17
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The smallest eigenvalue is 9.21e-22. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

Completed

In [ ]: