Final Project

STA 141A: Fundamentals of Statistical Data Science
Instructor: Emanuela Furfaro

September 7, 2022

1 Introduction

Different housing characteristics can easily affect house prices. Recently, due to increased work-from-home migration during the COVID-19 pandemic, housing demand has been increasing, and supply chain issues for building materials have led to a decrease in housing supply. In addition, the inflation rate in the United States has risen sharply, which has led to the deterioration of the low-income family as their living sources, and the need to replace housing to make up for living money. Housing supply then slowly increased, but not enough to meet demand. Therefore, the imbalance between supply and demand for housing has raised concerns about housing prices. We are trying to help buyers and house sellers to make more informed decisions at the ideal price. In this report, we will approach our proposal from the point of view of buyers and sellers. For house sellers, we will use statistical models based on house prices in King County, Washington to produce projected home prices for given housing conditions. For house buyers, we will use statistical knowledge to help buyers by entering ZIP codes in King County, Washington to identify homes at different house price levels.

2 Dataset Description

The dataset we used is obtained from Kaggle and describes the sale of individual residential properties in King County, Washington between May 2014 and May 2015. The dataset contains 21613 observations consisting of 21 aspects involved in assessing house values. We are using 19 aspects include price, bedrooms, bathrooms, sqft_living, sqft_lot, floors, waterfront, view, condition, grade, sqft_above, sqft_basement, yr_built, zipcode, lat, long, sqft_living15, and sqft_lot15.We have removed id and date from the dataset. Since the date of house sale is within a year, the variable of "date" will not be highly correlated with house price. And there is no relationship between the id of the house sold and the house price. We will be focusing on descriptive and predictive analytics, with suggestions on how additional data could be obtained to conduct more experiments to optimize the purchase. After separating the data into numerical and categorical variables, we picked some variables which could be associated with our outcome here.

2.1. Summary of response variable

Min.	1st Qu.	Medium	Mean	3rd Qu.	Max.
75000	321950	450000	540088	645000	7700000

Summary: Since the house price is our response variable, we made a summary table to observe the five statistics. The range of house prices is from 75000 to 7700000, but our mean is only 540088 which indicates that there may be outliers in our dataset. Also, the 75th percentile of our prices is 645000 which means that 75 percentage of the data is less than or equal to 645000, which is much lower than the maximum 7700000, once again indicating that the range is very large and there may be outliers.

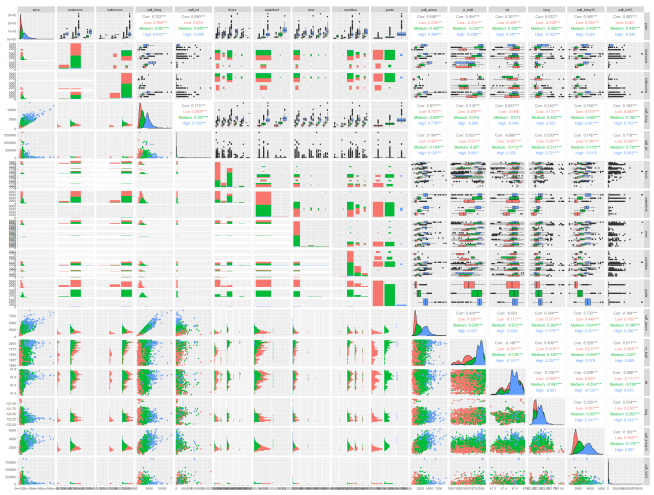
3 All of our analyses rely on the assumptions below

- All observations are randomly sampled (independent)
- All groups are independent (The variances of the population are equal)
- E - N(0, sigma2)

4 Research Question

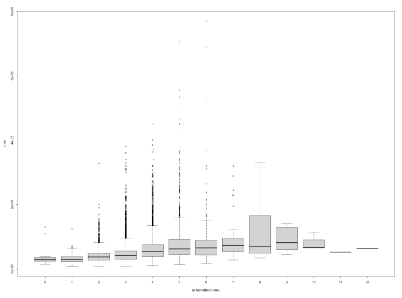
1. For house sellers (**In Section 5-7**): How could we help the house buyers to predict the price for their house by using some provided house features?
2. For house buyers (**In Section 8**): When the house buyer provides his preference for the house location such as zip codes, could we provide this specific location's house type (if the houses in this location are majorly luxury or economic) back to them?

5 Data Visualization and Model Analysis

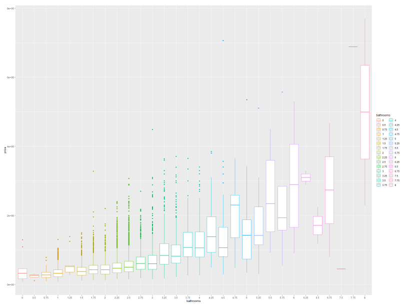


5.1 Data Tidy

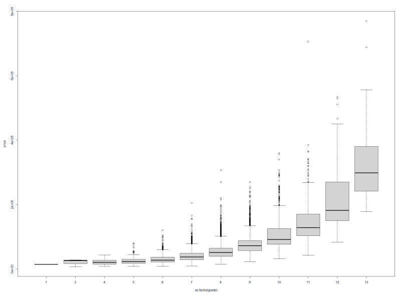
Firstly, we delete the variables "id", "data", "sqft_basement", "yr_renovated" and "zip code" because we don't want to consider those features. (Most of the houses do not have a basement and have not yet been renovated since built. for zip code, we can replace it with latitude and longitude.) **Secondly**, we check whether there is NA in our dataset. Since the sum of NA is zero, there is no need for us to do NA removal. **Thirdly**, We recognize which variables are categorical and which are numeric based on the data we have already manipulated: the categorical variables are bedroom, bathroom, floors, waterfront, view, grade, condition, the rest of variables are numeric. We convert all the variables we think are categorical to categorical variables. For bedroom, bathroom and grade, we divide each of them into three levels, so that it will be easier for us to do the prediction.



(a) Bedroom



(b) Grade



(c) Bathrooms

We use the boxplot to display the distribution of Bedrooms data based on a five number summary. By observing these 13 boxplots above, we found that the medians of boxplots 0 to 2 are very close; the medians of boxplots 3 to 6 are approximately close; the medians of boxplots 7 to 33 are close. We set 0 to 2 as the "Small" group, 3 to 6 as the "Medium" group, and 7 to 33 as the " A lot" group.

For our categorical variable of Bathrooms, we create a ggplot style plot and use the geom-boxplot function to display the distribution of Bathrooms data based on a five number summary. We use an extra aesthetic mapping color = bathrooms to display the boxplots in different colors for different groups. The plot will also include a legend, which should include the categories in bathrooms and the corresponding colors. On the x axis the variable is bathrooms, and on the y axis the variable is price.

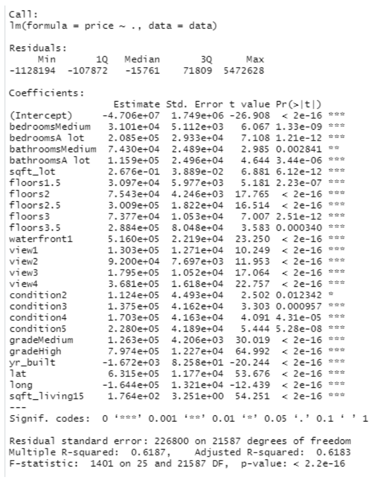
For our categorical variable of grade, we also use the boxplot to set levels by observing their medians, and separate them into three groups: “Low”, “Medium”, “High”. We set 1 to 7 as the “Low” group; 8 to 10 as the “Medium” group; and 11 to 13 as the “High” group. On the x axis the variable is as.factor(grade), and on the y axis the variable is price.

5.2 Variable Selection

```
##       [,1]            [,2]
##  [1,] "sqft_living"   "bathrooms"
##  [2,] "bathrooms"     "sqft_living"
##  [3,] "grade"         "sqft_living"
##  [4,] "sqft_above"    "sqft_living"
##  [5,] "sqft_living15" "sqft_living"
##  [6,] "sqft_lot15"    "sqft_lot"
##  [7,] "sqft_living"   "grade"
##  [8,] "sqft_above"    "grade"
##  [9,] "sqft_living15" "grade"
## [10,] "sqft_living"   "sqft_above"
## [11,] "grade"         "sqft_above"
## [12,] "sqft_living15" "sqft_above"
## [13,] "sqft_living"   "sqft_living15"
## [14,] "grade"         "sqft_living15"
## [15,] "sqft_above"    "sqft_living15"
## [16,] "sqft_lot"      "sqft_lot15"
```

Before constructing the model, we want to know if there are some variables that have strong correlation, if so, we will delete them. The correlation matrix we get using the ‘cor()’ function tells us that some variables are highly correlated with each other. In this project, we assume that if the correlation is larger than 0.7, there is a strong correlation. The variable names on the matrix_relation up there showing the variables who have strong correlation (In this matrix, two variable names on each row are strongly correlated) we decide to delete variables ’sqft_lot15’, ’sqft_living’, and ’sqft_above’ . the reason is that For ’sqft_living’, we find has strong correlation with a lot of other variables, if we keep ’sqft_living’, many other variables cannot be use, so we delete it. For ’sqft_lot15’, we find it has a strong correlation with ’sqft_lot’, and we decide to keep ’sqft_lot’. For ’sqft_above’, we find it has a strong correlation with ’sqft_living’, and we decide to delete ’sqft_above’.

6 Construct Model & Analysis

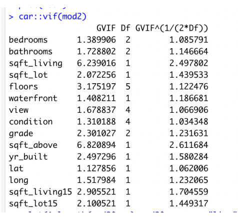


```
Call:
lm(formula = price ~ ., data = data)

Residuals:
     Min       1Q   Median       3Q      Max 
-1128194  -107872   -15761    71809  5472628 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
(Intercept)    -4.706e+07  1.749e+06 -26.908  < 2e-16 ***
bedroomsMedium  3.101e+04  5.112e+03   6.067 1.33e-09 ***
bedroomsA lot   2.085e+05  2.933e+04   7.108 1.21e-12 ***
bathroomsMedium 7.430e+04  2.489e+04   2.985 0.002841 ** 
bathroomsA lot  1.159e+05  2.496e+04   4.644 3.44e-06 ***
sqft_lot        2.676e-01  3.889e-02   6.881 6.12e-12 ***
floors1.5       3.097e+04  5.977e+03   5.181 2.23e-07 ***
floors2         7.543e+04  4.246e+03  17.765  < 2e-16 ***
floors2.5       3.009e+05  1.822e+04  16.514  < 2e-16 ***
floors3         7.377e+04  1.053e+04   7.007 2.51e-12 ***
floors3.5       2.884e+05  8.048e+04   3.583 0.000340 ***
waterfront1     5.160e+05  2.219e+04  23.250  < 2e-16 ***
view1           1.303e+05  1.271e+04  10.249  < 2e-16 ***
view2           9.200e+04  7.697e+03  11.953  < 2e-16 ***
view3           1.795e+05  1.052e+04  17.064  < 2e-16 ***
view4           3.681e+05  1.618e+04  22.757  < 2e-16 ***
condition2      1.124e+05  4.493e+04   2.502 0.012342 *  
condition3      1.375e+05  4.162e+04   3.303 0.000957 ***
condition4      1.703e+05  4.163e+04   4.091 4.31e-05 ***
condition5      2.280e+05  4.189e+04   5.444 5.28e-08 ***
gradeMedium     1.263e+05  4.206e+03  30.019  < 2e-16 ***
gradeHigh       7.974e+05  1.227e+04  64.992  < 2e-16 ***
yr_built       -1.672e+03  8.258e+01 -20.244  < 2e-16 ***
lat             6.315e+05  1.177e+04  53.676  < 2e-16 ***
long           -1.644e+05  1.321e+04 -12.439  < 2e-16 ***
sqft_living15   1.764e+02  3.251e+00  54.251  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 226800 on 21587 degrees of freedom
Multiple R-squared:  0.6187,	Adjusted R-squared:  0.6183 
F-statistic:  1401 on 25 and 21587 DF,  p-value: < 2.2e-16
```

6.1 Initial Model

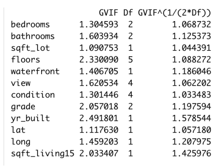
Firstly, we can see that the P-value for each variable is extremely small, that is to say, all of those variables are statistically significant for the response variable price.

Secondly, the adjusted R squared is 0.6183 this shows that almost 61.88% can be explained by our model. This number is not that high, so we can conclude that our model is not that good. We need do some adjustments to this model. And also, the residual standard error is extremely large which also shows that the mode is that good.



```
> car::vif(mod2)
                   GVIF Df GVIF^(1/(2*Df))
bedrooms       1.389906  2        1.085791
bathrooms      1.728802  2        1.146664
sqft_living    6.239016  1        2.497802
sqft_lot       2.072256  1        1.439533
floors         3.175197  5        1.122476
waterfront     1.408211  1        1.186681
view           1.678837  4        1.066906
condition      1.310188  4        1.034348
grade          2.301027  2        1.231631
sqft_above     6.820894  1        2.611684
yr_built       2.497296  1        1.580284
lat            1.127856  1        1.062006
long           1.517984  1        1.232065
sqft_living15  2.905521  1        1.704559
sqft_lot15     2.100521  1        1.449317
```

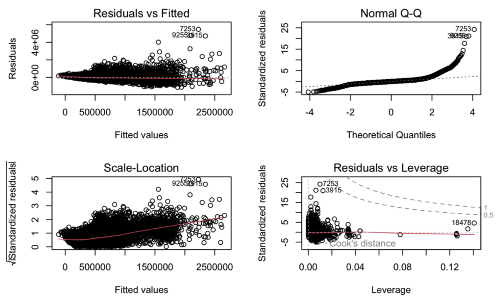
(d) VIF 1



```
                   GVIF Df GVIF^(1/(2*Df))
bedrooms       1.304593  2        1.068732
bathrooms      1.603934  2        1.125373
sqft_lot       1.090753  1        1.044391
floors         2.330090  5        1.088272
waterfront     1.406705  1        1.186046
view           1.620534  4        1.062202
condition      1.301446  4        1.033483
grade          2.057018  2        1.197594
yr_built       2.491801  1        1.578544
lat            1.117630  1        1.057180
long           1.459203  1        1.207975
sqft_living15  2.033407  1        1.425976
```

(e) VIF 2

We use variance inflation factor (VIF) to detect the severity of multicollinearity in the ordinary least square (OLS) regression analysis. We found these VIF values are within 5, so there is a moderate correlation. We do not need to remove any predictors to avoid multicollinearity.



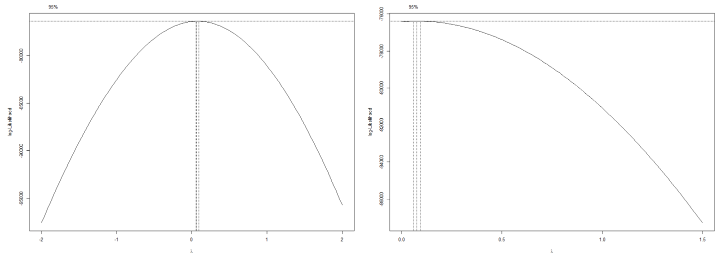
To find out what is wrong with this model that makes it less fittable, we will use the following diagram for diagnosis.

The Residuals vs Fitted plot suggests heteroskedasticity since the variance of the errors seems to increase when large values are fitted, and it also suggests that units 7253, 9255, and 3915 are outliers. The Normal Q-Q plot shows that the normality assumption does not hold up well in Normal Q-Q plot, since the points at the beginning and the end do not fit around the Q-Q dash line with obvious deviation patterns and have heavy tails. The Scale-Location plot confirms our suspicions about heteroskedasticity and the outliers discussed before. The Residuals vs. Leverage plot shows us that observations 7253, 3915, and 18478 are outliers, but there are not any high leverage points in our regression model because there is no plots beyond the cock's distance 0.5.

In the next section, we are going to fix these problems and provide a better linear prediction model called mod2.

6.2 Diagnosis

6.2.1 Solve Heteroskedasticity & Heavy Tails

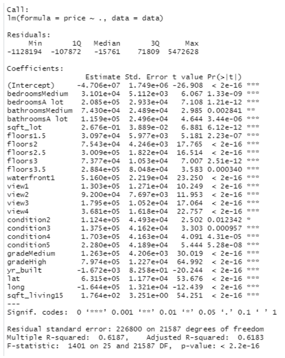


Since there is heteroskedasticity problem by looking at the residual plots, we use the concave function to adjust the price so that the data has a constant variance. And we decide to logarithmic transformation on our response variables because from the box-test we find that the lambda is almost zero. (As the plots above show that the 0 is inside the confidence interval of the optimal lambda () and as the estimation of the parameter is really close to 0 in this case, it suggests the best option is to apply the logarithmic transformation of the data.)

6.2.2 Solve Outliers

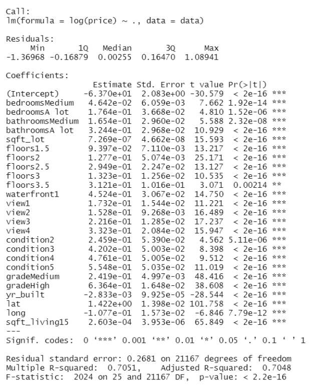
We find the 7253, 9255, and 3915 outliers from the above four graphs. So delete all the rows in that data set that contain those outliers.

6.3 Final Model (After Adjustment)



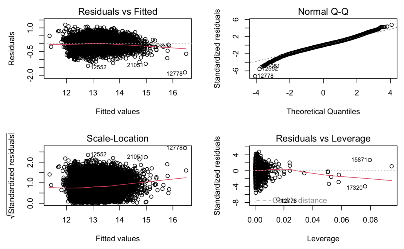
```
Call:
lm(formula = price ~ ., data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1128194  -107872   -15761    71809  5472628

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -4.706e+07  1.749e+06 -26.908  < 2e-16 ***
bedroomsMedium   3.101e+04  5.112e+03   6.067 1.33e-09 ***
bedroomsA lot    2.085e+05  2.933e+04   7.108 1.21e-12 ***
bathroomsMedium  7.430e+04  2.489e+04   2.985 0.002841 **
bathroomsA lot   1.159e+05  2.496e+04   4.644 3.44e-06 ***
sqft_lot         2.676e-01  3.889e-02   6.881 6.12e-12 ***
floors1.5        3.097e+04  5.977e+03   5.181 2.23e-07 ***
floors2          7.543e+04  4.246e+03  17.765  < 2e-16 ***
floors2.5        3.009e+05  1.822e+04  16.514  < 2e-16 ***
floors3          7.377e+04  1.053e+04   7.007 2.51e-12 ***
floors3.5        2.884e+05  8.048e+04   3.583 0.000340 ***
waterfront1      5.160e+05  2.219e+04  23.250  < 2e-16 ***
view1            1.303e+05  1.271e+04  10.249  < 2e-16 ***
view2            9.200e+04  7.697e+03  11.953  < 2e-16 ***
view3            1.795e+05  1.052e+04  17.064  < 2e-16 ***
view4            3.681e+05  1.618e+04  22.757  < 2e-16 ***
condition2       1.124e+05  4.493e+04   2.502 0.012342 *
condition3       1.375e+05  4.162e+04   3.303 0.000957 ***
condition4       1.703e+05  4.163e+04   4.091 4.31e-05 ***
condition5       2.280e+05  4.189e+04   5.444 5.28e-08 ***
gradeMedium      1.263e+05  4.206e+03  30.019  < 2e-16 ***
gradeHigh        7.974e+05  1.227e+04  64.992  < 2e-16 ***
yr_built        -1.672e+03  8.258e+01 -20.244  < 2e-16 ***
lat              6.315e+05  1.177e+04  53.676  < 2e-16 ***
long            -1.644e+05  1.321e+04 -12.439  < 2e-16 ***
sqft_living15    1.764e+02  3.251e+00  54.251  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 226800 on 21587 degrees of freedom
Multiple R-squared:  0.6187,    Adjusted R-squared:  0.6183
F-statistic:  1401 on 25 and 21587 DF,  p-value: < 2.2e-16
```

(f) Summary of old model

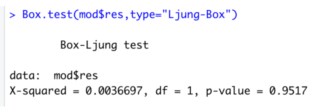


```
Call:
lm(formula = log(price) ~ ., data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.36968 -0.16879  0.00255  0.16470  1.08941

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -6.370e+01  2.083e+00 -30.579  < 2e-16 ***
bedroomsMedium   4.642e-02  6.059e-03   7.662 1.92e-14 ***
bedroomsA lot    1.764e-01  3.668e-02   4.810 1.52e-06 ***
bathroomsMedium  1.654e-01  2.960e-02   5.588 2.32e-08 ***
bathroomsA lot   3.244e-01  2.968e-02  10.929  < 2e-16 ***
sqft_lot         7.269e-07  4.662e-08  15.593  < 2e-16 ***
floors1.5        9.397e-02  7.110e-03  13.217  < 2e-16 ***
floors2          1.277e-01  5.074e-03  25.171  < 2e-16 ***
floors2.5        2.949e-01  2.247e-02  13.127  < 2e-16 ***
floors3          1.323e-01  1.256e-02  10.535  < 2e-16 ***
floors3.5        3.121e-01  1.016e-01   3.071  0.00214 **
waterfront1      4.524e-01  3.067e-02  14.750  < 2e-16 ***
view1            1.732e-01  1.544e-02  11.221  < 2e-16 ***
view2            1.528e-01  9.268e-03  16.489  < 2e-16 ***
view3            2.216e-01  1.285e-02  17.237  < 2e-16 ***
view4            3.323e-01  2.084e-02  15.947  < 2e-16 ***
condition2       2.459e-01  5.390e-02   4.562 5.11e-06 ***
condition3       4.202e-01  5.003e-02   8.398  < 2e-16 ***
condition4       4.761e-01  5.005e-02   9.512  < 2e-16 ***
condition5       5.548e-01  5.035e-02  11.019  < 2e-16 ***
gradeMedium      2.419e-01  4.997e-03  48.416  < 2e-16 ***
gradeHigh        6.364e-01  1.648e-02  38.608  < 2e-16 ***
yr_built        -2.833e-03  9.925e-05 -28.544  < 2e-16 ***
lat              1.422e+00  1.398e-02 101.758  < 2e-16 ***
long            -1.077e-01  1.573e-02  -6.846 7.79e-12 ***
sqft_living15    2.603e-04  3.953e-06  65.849  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2681 on 21167 degrees of freedom
Multiple R-squared:  0.7051,    Adjusted R-squared:  0.7048
F-statistic:  2024 on 25 and 21167 DF,  p-value: < 2.2e-16
```

(g) Summary of new model

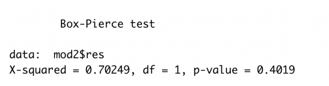


(h) The log transformation of Y without outliers



```
> Box.test(mod$res,type="Ljung-Box")

        Box-Ljung test

data:  mod$res
X-squared = 0.0036697, df = 1, p-value = 0.9517
```

(i) Box-Pierce test of model 1



```
        Box-Pierce test

data:  mod2$res
X-squared = 0.70249, df = 1, p-value = 0.4019
```

(j) Box-Pierce test of model 2

R Squared comparison: Above is the summary for our second model based on the data after diagnosis. We can observe the R Squared increased from 0.6183 to 0.7048. The bigger R Squared means the better this model performed. This shows that the multiple linear regression model after diagnosis performs better than the previous one.

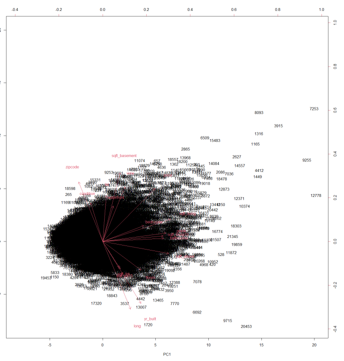
In The Residuals vs Fitted plot compared to the previous model, we can observe that the data points are distributed more evenly along the line of zero. It shows that our variance after diagnosis is closer to constant variance. The information we can learn from the comparison of the two models QQ Plot is that the data residual fits more along the indicated line and we definitely observe a lighter tail compared to the previous graph. Comparing two graphs from the box-test, we find the box-test for model 2 that almost all of the residuals are in the range (-1,1) which is more stationary. The P-value from box-test plot for model 2 is 0.40190 which is much smaller than 0.9517 for model 1.

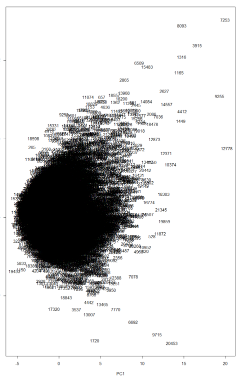
The final equation

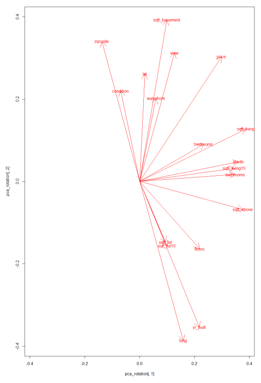
$$Y_i = -6.370e + 01 + 4.642e - 02X_{bedroomsMedium} + 1.764e - 01X_{bedroomsAlot} + 1.654e - 01X_{bathroomsMedium}$$
$$+ 3.244e - 01X_{bathroomsAlot} + 7.269e - 07X_{sqft_lot} + 9.397e - 02X_{floors1.5} + 1.277e - 01X_{floors2} + 2.949e - 01X_{floors2.5}$$
$$+ 1.323e - 01X_{floors3} + 3.121e - 01X_{floor3.5} + 4.524e - 01X_{waterfront1} + 1.732e - 01X_{view1} + 1.528e - 01X_{view2} + 2.216e - 01X_{view3}$$
$$+ 3.323e - 01X_{view4} + 2.459e - 01X_{condition2} + 4.202e - 01X_{condition3} + 4.761e - 01X_{condition4} + 5.548e - 01X_{condition5}$$
$$+ 2.419e - 01X_{gradeMedium} + 6.364e - 01X_{gradeHigh} - 2.833e - 03X_{yr_built} + 1.422e + 00X_{lat} - 1.077e - 01X_{long}$$
$$+ 2.603e - 04X_{sqft_living15}$$

7 Checking Computations

7.1 Principal Component Analysis (PCA)



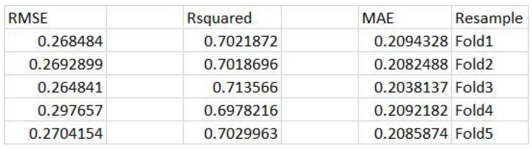




We use an unsupervised learning method that creates a principal component analysis (PCA) algorithm to understand how much information we lose by reducing the dimension of the data set. Using dimensionality reduction techniques like PCA can downscale our data to 19 aspects and visualize it in two dimensions. PCA is a good algorithm for dimensionality reduction, by using PCA we determine again which variables are important and which are not for our analysis. The PCA plot shows 19 aspects, while previously we had artificially selected 16 aspects when we did the variables selection, these 16 variables are all among the 19 variables selected by the machine through PCA dimensionality reduction, indicating that our variables were well selected and we did not select any variables that were not useful. The principal component(arrows) direction of the data shows what the observations vary the most, and projected the observations onto this panel. Our graph's main cluster is too dense to allow for linear separation between categories, but the arrow direction could point to the corresponding variables, and the length of each variable's solid line represents the percentage of each variable on each PC.

7.2 K-fold Cross-validation

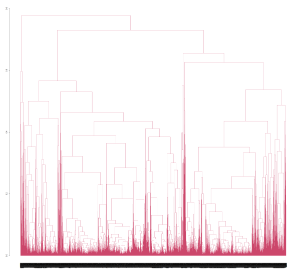
Since we have got our fit model, we want to check if our model has a good quantity. And what if our model is overfitting? We will divide the whole data to k=5 groups sub-data and use the k-fold cross validation measure to find out the MSE for each test data and training data. Then we compared the training MSE and testing MSE.
There are five pairs of training MSE and testing MSE. Firstly, MSE for training data is smaller than MSE for testing data. More importantly, we find that there is not a huge difference between MSE for training data and for testing data. It makes sense because we construct the model based on our training data, that is to say, our model may pick up some pattern that is just caused by randomness. So the difference between them is acceptable. In conclusion, our model is not overfitting and it has a good quantity for prediction.

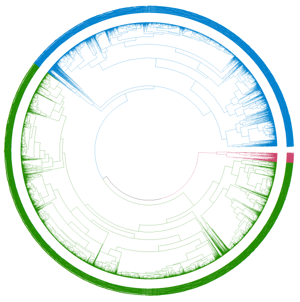


RMSE		Rsquared		MAE	Resample
0.268484		0.7021872		0.2094328	Fold1
0.2692899		0.7018696		0.2082488	Fold2
0.264841		0.713566		0.2038137	Fold3
0.297657		0.6978216		0.2092182	Fold4
0.2704154		0.7029963		0.2085874	Fold5

8 Unsupervised learning

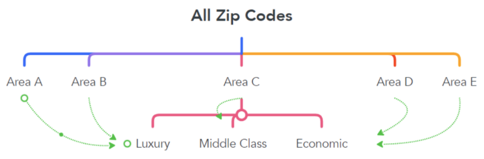
8.1 Hierarchical Clustering



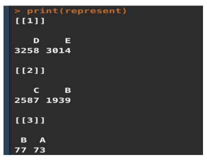


After scaling all the numeric variables with standard deviation 1 and mean 0 and using complete linkage, we cluster the house. Here we get two dendrograms: the first one shows the whole clustering for each house, and in the second one we get tree clusters for houses(the green is for cluster 1, the blue is for cluster 2, the red is for cluster 3). From unsupervised learning, we know that the houses in each cluster have similarities. We use this property to find and compare the mean of the house price for each cluster's houses. We find that cluster 1 has the lowest price, cluster 2 has the middle price, and cluster 3 has the highest price. Hence, we can summarize cluster 1 as "Economic", cluster 2 as "Middle Class", and cluster 3 as "Luxury". From the figure, green has the largest proportion, blue has the second largest proportion, and red has the smallest proportion. That is saying that the number of Economic houses is larger than the Middle-Class houses, and the number of Luxury houses is the smallest. It also makes sense when we look at the data set.

8.2 Set the Game: A



(a) **Figure 1**



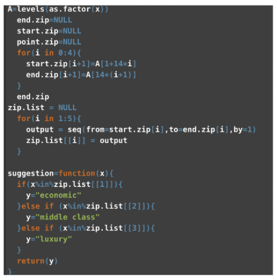
```
> print(represent)
[[1]]

   D    E
3258 3014

[[2]]

   C    B
2587 1939

[[3]]

 B  A
77 73
```

(b) **Figure 2**

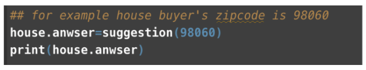
On the **Figure 1** above, We divided all the zipcodes in King County into ive groups which are A, B, C, D, and E, according to the principle of similar ending numbers (the closer the ending numbers of the zipcodes, the closer the area they represent geographically). Also, we have divided all the houses in King County into three levels by price, which are economic level houses, middle-class level houses, and luxury level houses. Based on the algorithm construction above, our model will determine in each group which level appears most, then we will define the group as this level, for example, luxury-level houses appear most in group A, so the areas represented by group A are the luxury area. What our model ultimately does is that when the buyer gives us a zip code, our model will tell the buyer whether this area(zip code) is at an economic level, a middle-class level, or a luxury level. Therefore, we can help buyers determine if the area represented by this zip code meets their budget.

On the **Figure 2** above, the list names 1, 2, and 3 are corresponding to clusters 1, 2, and 3. And by the same logic, clusters 1, 2, and 3 are corresponding to houses type “Economic”, “Middle Class”, and “Luxury”. The houses in area D and E show up almost the same time in cluster 1, so we say the houses in area D and E are generally Economic houses; the houses in area C shows up almost twice as in acre B, we say the houses in area C are generally Middle-class houses; the houses show up times in area B and A are both small and have not a big difference, we summary that the houses in area B and A are Luxury house.

8.3 Set up function



```
A=levels(as.factor(x))
  end.zip=NULL
  start.zip=NULL
  point.zip=NULL
  for(i in 0:4){
    start.zip[i+1]=A[1+14*i]
    end.zip[i+1]=A[14*(i+1)]
  }
  end.zip
zip.list = NULL
  for(i in 1:5){
    output = seq(from=start.zip[i],to=end.zip[i],by=1)
    zip.list[[i]] = output
  }

suggestion=function(x){
  if(x%in%zip.list[[1]]){
    y="economic"
  }else if (x%in%zip.list[[2]]){
    y="middle class"
  }else if (x%in%zip.list[[3]]){
    y="luxury"
  }
  return(y)
}
```



```
## for example house buyer's zipcode is 98060
house.anwser=suggestion(98060)
print(house.anwser)
```

When a house buyer comes to us with his zipcode, we can tell him what the houses around his location are. Therefore, the buyer can decide whether to buy a house in this area based on his financial situation or to find a place where the house generally met his needs for the type of house he wanted.

For example, if the house buyer tells us he wants to buy a house with the zipcode 98060, and he says he is a poor guy he just wants an economical house. We can tell him that the house around the area he prefers is almost a luxury house, so we suggest he should buy a house in another place. Then, he can tell us his second preference for the house location, and then we can suggest if it is better to buy a house from this location. He can tell us whatever positions he prefers and we can give him a suggestion accordingly.

9 Discussion and Results

Task: (Big Question)

1. **For house sellers: How could we help the house buyers to predict the price for their house by using some provided house features?**
   Answer: We construct a multiple linear regression model with house price as our response variables and features of the house as our predictors.

2. **For house buyers: When the house buyer provides his preference for the house location such as zip codes, could we provide this specific location’s house type (if the houses in this location are majorly luxury or economic) back to them?**
   Answer: We apply an unsupervised learning technique to separate the whole data set into 3 different clusters, and based on each cluster’s house price to manually tag the clusters. Finally we can relate the clusters back to zip codes to provide the housing information in each area.

Procedures: (Sub-Question)

1. **Which manipulation should we do on the data set ?**
   - Firstly, we let the data to be data frame
   - We need to set the discontinuous variables to category variables
   - If there is NA, if so, delete NA

2. **Which variables should we select, and which one should be deleted?**
   - We can do the data visualization, for example, the cor() function to check if there are collinearity problems among the variables, if so, we will choose one of the variables from the multicollinearity variables to avoid it.

- Using the VIF() function. If the VIF is larger than 5 or 10, it shows the strong correlation, and then we will delete that variable.
- Using the PCA to recheck

3. **Does the model satisfy our three statistical assumptions(the residuals are independent, have constant variance and normal distribution)?**
   - Using box-test to test if the residuals are independent
   - Using Q-Q plot to test if it is normally distributed, if not, use box-cox to adjust it
   - Using residual plot to see if has constant variance
   - Using the box-test to check if residuals are around zero
4. **By diagnosing our model, what adjustments do we need to make for our model? (is there an outlier or higher leverage point? Is our linear model linear? Is our data normally distributed?)**
   - Data visualization to check Normal Q-Q plot, residual plot and cook's distance and so on
   - Use the box cox to find how to adjust the response in our model
   - Check the adjusted R square to find how much can be explained by our model
   - Check Cook's distance
5. **Which model should we use to predict house prices? Is our model over-fitting?**
   - Select the model has bigger adjusted R
   - Select the model has smaller MSE
   - Use the cross validation to check the quantity of our model
6. **How to use hierarchical clustering to solve the big question two?**
   - First, scale the data to let the data has standard division 1 and mean 0
   - Second, divide the zip code to five group so that can show a area rather than a point
   - Cut the cluster to three cluster
   - Find the mean price for each cluster so that we can identify the three type of the similar houses in each cluster
   - Count the frequency for each levels of zip code in each cluster, and then identify the area for each cluster
   - Set up a function to connect the type of house and area: when buyer show his location we can tell him the type of house

10 Conclusion

For house sellers, we construct a multiple linear regression model, with the response variables and house characteristics as predictors realized the prediction of the seller's house price. For house buyers, we apply unsupervised learning techniques to achieve that when the buyer provides a zipcode, we can give him feedback on the general level(economic, middle class, luxury) of the houses in this area. After adjusting and diagnosing our model, such as removing outliers, filtering variables a second time by PCA downscaling, reconstructing the model to have smaller MSE, and using cross-validation to check the model's quantity, we find the best model which is very reasonable and statistically significant. Also, our model satisfies the research assumptions quite well. Now, our model could count the frequency for each level of zipcode in each cluster and then identify the area for each cluster. Then, we set up a function to connect the type of house and area: when the buyer shows his location we can tell him the type of house. In future work, we could use a more complex model. Because the overall/average house price in an area where the targeting house is located is only a rough predictor for the house's level. The model and dataset should allow for multiple dimensions of the city's house market, such as including the number of schools, stores, and traffic stations in the living circle.

11 Reply to Professor's comments

1. **Are the methodologies proposed appropriate to answer the research questions? Are they appropriate to deal with the type of data?**

   Yes. The statistical methods we used mentioned can help us to reach the main goals of this project and it is appropriate to deal with this type of data.

2. **Are the key questions clearly stated? Can they be answered using the chosen dataset?**

   We considered your precious advice and re-structured our key questions. Finally we nail the key question down to only 2 parts, solve one question for house sellers and one question for house buyers. Now we think the questions are clearly stated and can be answered by the chosen dataset.

Appendix code

~~~
```{r ref.label=knitr::all_labels(), echo = T, eval = F}
rm(list=ls())
kc=read.csv("kc_house_data.csv")
kc=as.data.frame(kc)
kc=kc[,c(-1,-2,-14,-17,-16)]
class(kc)
print(sum(is.na(kc)))


#################################################################
##########################  variable selection ##################
#################################################################

# we consider the correlation be larger than 0.5 represents the strong
relationship
matrix_cor=cor(kc[,-1])
print(matrix_cor)
dim=which(abs(cor(kc[,-1]))>=0.7,arr.ind = T)
dim=dim[dim[,1]!=dim[,2],]
matrix_relation= cbind(rownames(matrix_cor)[dim[,1]],colnames(matrix_cor)
[dim[,2]])
print(matrix_relation)
#we find that there are strong relationship
#delete the variables: sqft_lot15，sqft_living，sqft_above
# we set the  bedroom, bathrooms, waterfront, view, condition, grade, as
categorical variables
# for bedrooms and bathrooms we want to group them into some levels
plot(price~as.factor(bedrooms),data = kc)
#according to the plot, we set the bedrooms to "small","medium","A lot" those
three group.
kc$bedrooms=factor(ifelse(as.numeric(kc$bedrooms)<3,'Small',ifelse(as.numeric(kc$bedrooms)
lot','Medium')),levels = c('Small','Medium','A lot'))
plot(price~as.factor(grade),data = kc)
#according to the plot, we set the grade to "Low","medium","High" those three
group.
kc$grade=factor(ifelse(as.numeric(kc$grade)<8,'Low',ifelse(as.numeric(kc$grade)>10,'High',
= c('Low','Medium','High'))

library(ggplot2)
kc$bathrooms=as.factor(kc$bathrooms)
ggplot(data = kc)+
  geom_boxplot(mapping = aes(x=bathrooms,y=price,color=bathrooms))
#according to the plot, we set the bathrooms to "small","medium","A lot" those
three group.
kc$bathrooms=factor(ifelse(as.numeric(kc$bathrooms)<3.5,'Small',ifelse(as.numeric(kc$bathr
lot','Medium')),levels = c('Small','Medium','A lot'))
kc$waterfront=factor(kc$waterfront,ordered = FALSE)
kc$view=factor(kc$view,ordered = FALSE)
kc$condition=factor(kc$condition,ordered = FALSE)
kc$floors=factor(kc$floors,ordered = FALSE)

## before construct model, let re check the data
~~~

```
install.packages("GGally",repos = "http://cran.us.r-project.org")
library(GGally)
library(ggplot2)
ggpairs(kc, columns = 1:16, aes(color = grade))

data = kc[,!(colnames(kc) == 'sqft_living'| colnames(kc) == 'sqft_lot15' |
colnames(kc) == 'sqft_above')]
class(data)

########################################################################################
######################### construct model & data visualization
########################
########################################################################################

mod = lm(price~.,data=data)
summary(mod)
par(mfrow = c(2,2))
plot(mod)
car::vif(mod)
qplot(1:length(mod$res),mod$res,geom="line") +
 geom_hline(yintercept = 0, colour = "red")
Box.test(mod$res)
#data visualization:
#1.from the residual plots we find there is a heteroskedasticity problems, so
we need to use the concave function to adjust the model in the following steps
#2. the variance is not that constant, so we hope use the box-cox method to do
some adjustment for Y
#3.from the cock's distance we find there isn't high leverage point
#4. the Q-Q plot shows a heavy tail distribution.


########solve outlier ########
outliers = which(abs(rstandard(mod))>3)
outliers
data = data[-outliers,]

######solve heteroskedasticity & solve non-constant variance########
library(MASS)
boxcox(mod, plotit = TRUE)
boxcox(mod, plotit = TRUE, lambda = seq(0, 1.5, by = 0.1))
mod2 = lm(log(price)~.,data=data)
summary(mod2)


##model analysis, diagnose##
par(mfrow = c(2,2))
plot(mod2)
car::vif(mod2)
qplot(1:length(mod2$res),mod2$res,geom="line") +
 geom_hline(yintercept = 0, colour = "red")
Box.test(mod2$res)
## after analysis, mod2 will be our model to predict price

##################################################
```

```r
###############  unsupervising learning ##########
#######################PCA######################
kc_2=read.csv("~/Desktop/STA141/kc_house_data.csv")#orignal
kc_2 = as.data.frame(kc_2[,-c(1,2,16)])
kc_pca <- prcomp(kc_2,scale.=TRUE)
par(mfrow=c(1,1))
biplot(kc_pca,scale = 0)


######################### K-fold Cross Validation #########################
install.packages("caret")
library(caret)
ctrl <- trainControl(method = "cv", number = 5)
model <- train(log(price)~ ., data = data, method = "lm", trControl = ctrl)
model$finalModel
print(model$resample)


#########################   PCA    ################################
kc_2=read.csv("~/Desktop/STA141/kc_house_data.csv")
kc_2 = as.data.frame(kc_2[,-c(1,2,16)])
kc_pca <- prcomp(kc_2,scale.=TRUE)
par(mfrow=c(1,1))
biplot(kc_pca,scale = 0)

plot(kc_pca$x[,1:2], # first two columns : (PC1,PC2)
     pch=rownames(kc_2),type='n')
text(kc_pca$x[,1],kc_pca$x[,2],labels=rownames(kc_2))
head(kc_pca$x[,1:2])


pca_rotation <- kc_pca$rotation[,1:2]
arrows( x0=0,y0=0, pca_rotation[,1], pca_rotation[,2], col='red' )
text( pca_rotation[,1], pca_rotation[,2],
labels=rownames(pca_rotation),col='red')


par(mfrow=c(1,2))
plot(kc_pca$x[,1:2], # first two columns : (PC1,PC2)
     pch=rownames(kc_2),type='n')
text(kc_pca$x[,1],kc_pca$x[,2],labels=rownames(kc_2))
plot(pca_rotation[,1], pca_rotation[,2],
     xlim=range(pca_rotation[,1],-pca_rotation[,1]),
     ylim=range(pca_rotation[,2],-pca_rotation[,2]),
     type='n')
arrows( x0=0,y0=0, pca_rotation[,1], pca_rotation[,2], col='red' )
text( pca_rotation[,1], pca_rotation[,2],
labels=rownames(pca_rotation),col='red')



##############################################################################
##################################for buyer #################################
##############################################################################
```

```
rm(list=ls())
kc=read.csv("~/Desktop/STA141/kc_house_data.csv")
kc=as.data.frame(kc[order(kc$zipcode),])
kc=kc[,c(-1,-2,-16,-18,-19)]
zip.fun=function(x){
  A=levels(as.factor(kc$zipcode))
  end.zip=NULL
  start.zip=NULL
  point.zip=NULL
  for(i in 0:4){
    start.zip[i+1]=A[1+14*i]
    end.zip[i+1]=A[14*(i+1)]
  }
  y=NULL
  zip.list = NULL
  for(i in 1:5){
    output = seq(from=start.zip[i],to=end.zip[i],by=1)
    zip.list[[i]] = output
  }
  for(i in 1:length(x)){
    if(x[i]%in%zip.list[[1]]){
      y[i]="A"
    }else if(x[i]%in%zip.list[[2]]){
      y[i]="B"
    }else if(x[i]%in%zip.list[[3]]){
      y[i]="C"
    }else if (x[i]%in%zip.list[[4]]){
      y[i]="D"
    }else if (x[i]%in%zip.list[[5]]){
      y[i]="E"
    }
  }
  return(y)
}

x=kc$zipcode
 kc$zipcode= as.factor(zip.fun(x))
 zipcode=kc$zipcode
kc=as.data.frame(kc)
kc$bathrooms=factor(ifelse(as.numeric(kc$bathrooms)<3.5,'Small',ifelse(as.numeric(kc$bathr
lot','Medium')),levels = c('Small','Medium','A lot'))
kc$waterfront=factor(kc$waterfront,ordered = FALSE)
kc$view=factor(kc$view,ordered = FALSE)
kc$condition=factor(kc$condition,ordered = FALSE)
kc$floors=factor(kc$floors,ordered = FALSE)
kc$bedrooms=factor(ifelse(as.numeric(kc$bedrooms)<3,'Small',ifelse(as.numeric(kc$bedrooms)
lot','Medium')),levels = c('Small','Medium','A lot'))
kc$grade=factor(ifelse(as.numeric(kc$grade)<8,'Low',ifelse(as.numeric(kc$grade)>10,'High',
= c('Low','Medium','High'))
data1=as.data.frame(kc)
data=as.data.frame(kc[,-14])
data[,c(1,4,5,11,12,13,14,15)]=scale(data[,c(1,4,5,11,12,13,14,15)])
#install.packages("dendextend")
```

```
#install.packages("circlize")
library(circlize)
library(dendextend)
library(cluster)
library(dplyr)
gower.dist <- daisy(data, metric = c("gower"))
hc.complete <- hclust(gower.dist,  method="complete")
cut_complete <- cutree(hc.complete, k = 3)
hc.complete_obj <- as.dendrogram(hc.complete)
hc.complete_final <- color_branches(hc.complete_obj, h = 3)
#Graph 2
plot(hc.complete_final)
#Graph 1
hc.complete.circle<- hc.complete %>%
  color_branches(k = 3) %>%
  color_labels(k = 3)
circlize_dendrogram(hc.complete.circle,
                    labels_track_height = NA,
                    dend_track_height = 0.5)

#######################################################
each_group <- mutate(data1, cluster = cut_complete)
split(each_group$zipcode,each_group$cluster)

cluster.list = list()
cluster.list = split(each_group$zipcode,each_group$cluster)
represent=NULL
level.represent=c("A","B","C","D")

represent = list()
for(i in 1:3){
  represent[[i]]=sort(table(cluster.list[[i]]),decreasing = T)[1:2]
}
represent
print(represent)


data.c1=data1[each_group$cluster==1,]
data.c2=data1[each_group$cluster==2,]
data.c3=data1[each_group$cluster==3,]

mean(as.numeric(data.c1$price))
mean(as.numeric(data.c2$price))
mean(as.numeric(data.c3$price))

################################################################
A=levels(as.factor(kc$zipcode))
  end.zip=NULL
  start.zip=NULL
  point.zip=NULL
  for(i in 0:4){
    start.zip[i+1]=A[1+14*i]
    end.zip[i+1]=A[14*(i+1)]
  }
```

~~~~
end.zip=end.zip[!is.na(end.zip)]
start.zip=end.zip[!is.na(start.zip)]
zip.list = NULL
  for(i in 1:5){
     zip.list[i] = seq(from=start.zip[i],to=end.zip[i],by=1)

  }

suggestion=function(x){
  if(x%in%zip.list[[1]]|x%in%zip.list[[2]]){
    y="Luxury"
  }else if (x%in%zip.list[[3]]){
    y="Middle Class"
  }else if (x%in%zip.list[[4]]|x%in%zip.list[[5]]){
    y="Economic"
  }
  return(y)
}


## for example house buyer's zipcode is 98060
house.anwser=suggestion(98060)
print(house.anwser)
```

## Appendix
```{r, ref.label=knitr::all_labels(),echo=TRUE,eval=FALSE}
```
~~~~