Final Project

Tenam lama

2022-06-23

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

## 'data.frame': 5000 obs. of 7 variables:  
## $ carat : num 0.41 0.5 1.03 1.1 1.51 0.3 0.87 1.05 1 2.01 ...  
## $ depth : num 62.3 62.8 65.2 62.1 63.3 62.1 61.4 63.3 64 63.8 ...  
## $ table : num 61 57 56 57 61 55 57 57 57 57 ...  
## $ width : num 4.72 5.05 6.42 6.6 7.24 4.3 6.17 6.45 6.29 7.95 ...  
## $ length: num 4.75 5.08 6.35 6.64 7.17 4.33 6.14 6.4 6.33 7.91 ...  
## $ height: num 2.95 3.18 4.16 4.11 4.56 2.68 3.78 4.07 4.04 5.06 ...  
## $ price : int 638 1402 3530 5037 13757 457 2321 5657 4372 13976 ...

## carat cut color clarity   
## Min. :0.2000 Length:53940 Length:53940 Length:53940   
## 1st Qu.:0.4000 Class :character Class :character Class :character   
## Median :0.7000 Mode :character Mode :character Mode :character   
## Mean :0.7979   
## 3rd Qu.:1.0400   
## Max. :5.0100   
## depth table width length   
## Min. :43.00 Min. :43.00 Min. : 0.000 Min. : 0.000   
## 1st Qu.:61.00 1st Qu.:56.00 1st Qu.: 4.710 1st Qu.: 4.720   
## Median :61.80 Median :57.00 Median : 5.700 Median : 5.710   
## Mean :61.75 Mean :57.46 Mean : 5.731 Mean : 5.735   
## 3rd Qu.:62.50 3rd Qu.:59.00 3rd Qu.: 6.540 3rd Qu.: 6.540   
## Max. :79.00 Max. :95.00 Max. :10.740 Max. :58.900   
## height price   
## Min. : 0.000 Min. : 326   
## 1st Qu.: 2.910 1st Qu.: 950   
## Median : 3.530 Median : 2401   
## Mean : 3.539 Mean : 3933   
## 3rd Qu.: 4.040 3rd Qu.: 5324   
## Max. :31.800 Max. :18823

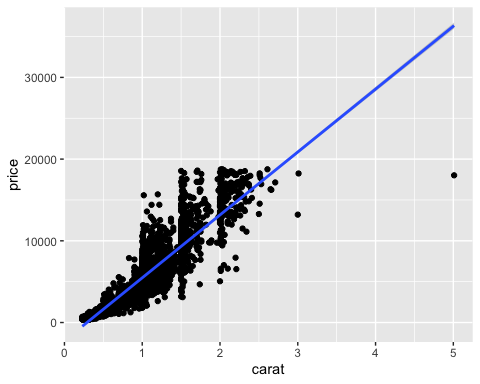
I found my data set from kaggle.com. It has about 55000 observation and 10 variables (3 categorical and 7 quantitative). I have choosen my quatitative variable - Price as my response variable for any supervised learning.

##   
## Fair Good Ideal Premium Very Good   
## 148 450 1916 1291 1195

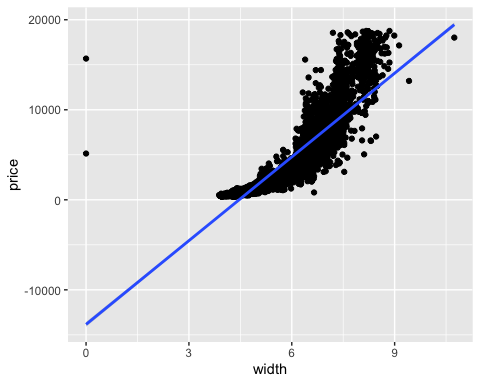
##   
## D E F G H I J   
## 611 904 859 1056 825 483 262

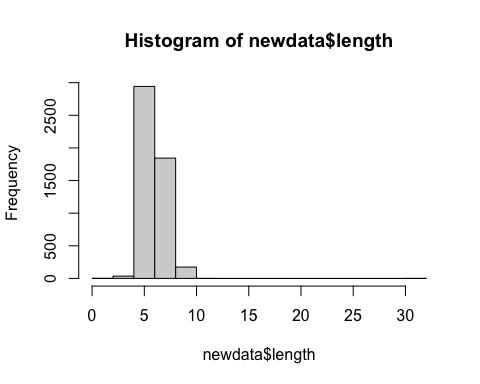
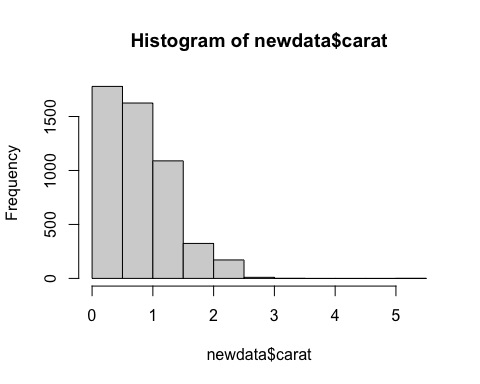
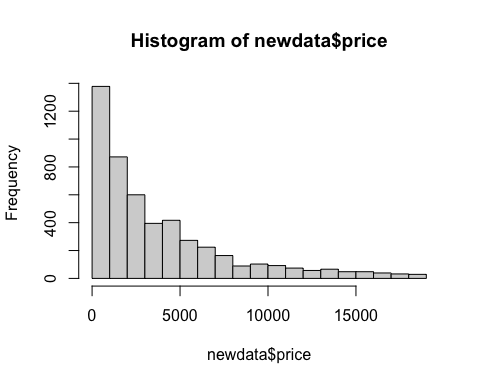
##   
## I1 IF SI1 SI2 VS1 VS2 VVS1 VVS2   
## 70 152 1193 860 764 1129 332 500

## `geom\_smooth()` using formula 'y ~ x'

 There seems to be a linear relationship between carat and price. As the carat increases the price of diamond seems to increase. There also seems to be an outlier. I have not removed the outlier just for time purposes but definitely in the future.

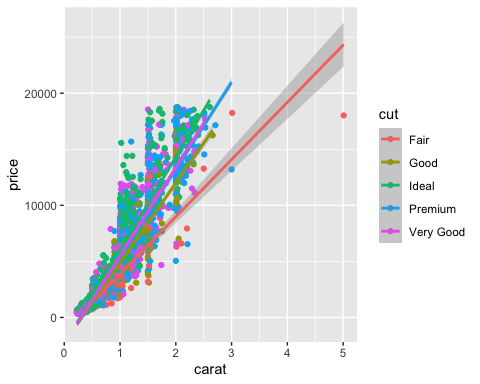
## `geom\_smooth()` using formula 'y ~ x'

 There seems to be a linear relationship between height and price. As the height increases the price of diamond seems to increase. There also seems to be an outlier. I have not removed the outlier just for time purposes but definitely in the future.

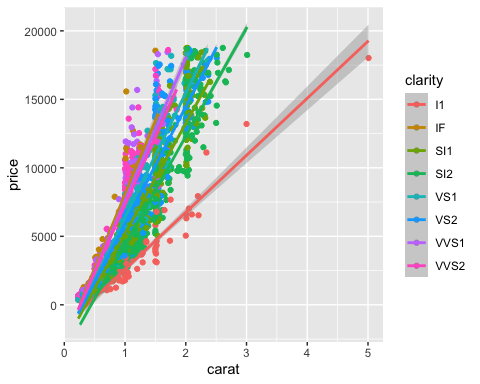
 Our data is pretty skewed to the right for both price and carat. This tells us that there are more number of diamonds that are inexpensive and have less carats than their are expensive with high carats. This tells us that they are dependent on one another, therefore positive correlation.

The histogram for length shows that there are more number of diamonds that have a length between 4 and 7.

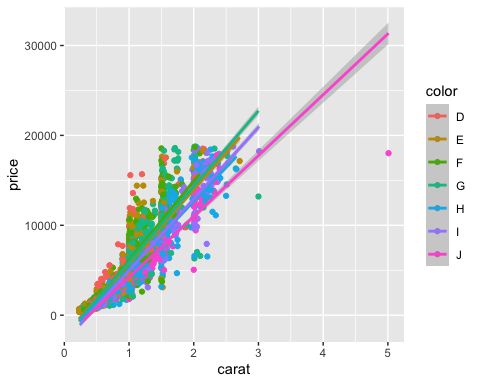
## `geom\_smooth()` using formula 'y ~ x'

 It looks like better the cut of the diamond higher the price. This shows a positive linear relationship between cut and price also price and carat.

## `geom\_smooth()` using formula 'y ~ x'

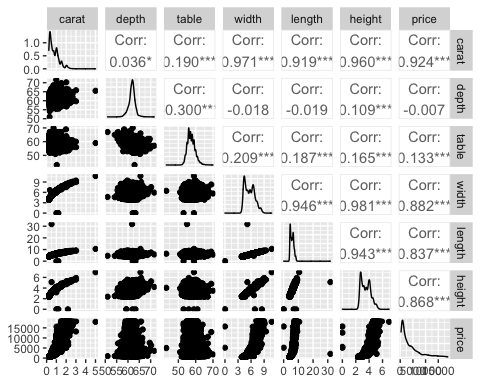
 This scatter plot shows that as clarity is better the price is higher.

## `geom\_smooth()` using formula 'y ~ x'



This scatterplot represents that color for a diamond is alphabetically arranged in which higher the alphabet lower the price. Therefore, negative correlation between price and color.

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

 Variables width and height has the largest positive correlation of 0.981. Variables table and depth has the largest negative correlation coefficient of -0.300. There seems to be highly correlated variables such as length, height, and width. This likely is showing there is overlap and the fact that change in any one of the three variables results to little effect on our model.

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## Standard deviation 2.1763 1.1360 0.82682 0.43217 0.25874 0.16359 0.09536  
## Proportion of Variance 0.6766 0.1844 0.09766 0.02668 0.00956 0.00382 0.00130  
## Cumulative Proportion 0.6766 0.8610 0.95863 0.98531 0.99488 0.99870 1.00000

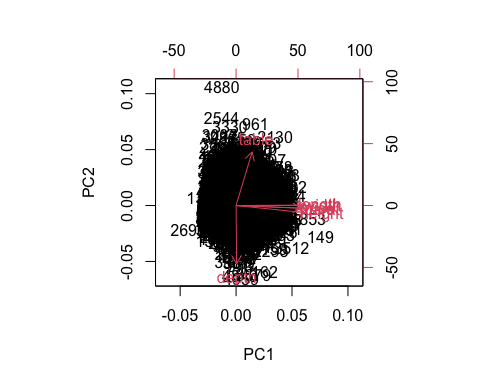
## PC1 PC2 PC3 PC4 PC5  
## carat 0.452571010 -0.0347417656 0.01056185 -0.13785434 0.34195521  
## depth 0.003671826 -0.7328475217 -0.66918152 -0.04226632 -0.06954696  
## table 0.105140273 0.6726498703 -0.72861229 -0.06296629 -0.03744078  
## width 0.453750638 0.0059819808 0.03844655 0.20547037 0.38055910  
## length 0.440437752 0.0001785218 0.05830084 0.46767480 -0.74709650  
## height 0.450290429 -0.0896560039 -0.04422833 0.25224097 0.29327596  
## price 0.425998822 -0.0345782681 0.11989748 -0.80664843 -0.29637619  
## PC6 PC7  
## carat -0.78567571 -0.2015872753  
## depth -0.02038929 0.0898786980  
## table 0.01565260 0.0004868328  
## width 0.19812443 0.7525289671  
## length -0.15873557 0.0233599116  
## height 0.50486626 -0.6186420208  
## price 0.25042512 0.0414809759

as our price value is way off we scale them by their standard deviation

The first PC accounts for about 67.66% of the variation in the data set. The first two PCs account for over 86.10% of the variation.

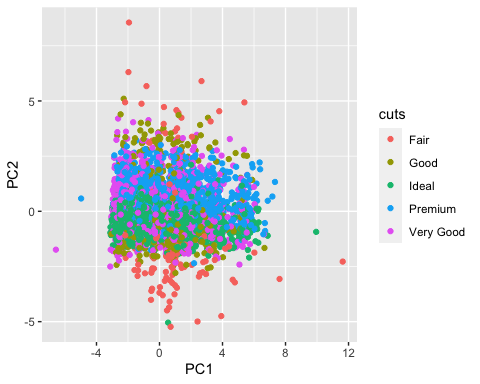
All variable have positive loading value for PC1. Whereas for PC2, variables table, width, and length got positive loading values and variables carat, depth, height, and price got negative loading values.

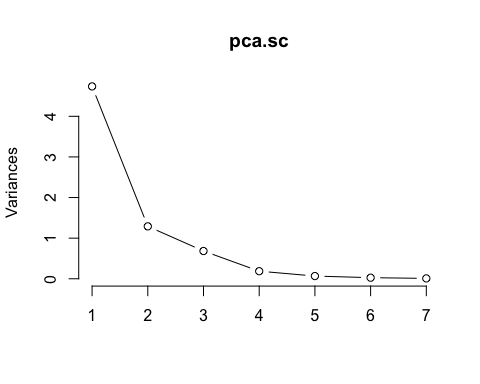
Variables carat, width, length, height, and price got the largest (in absolute value) loading values in PC1. Variables depth and table got the largest (in absolute value) loading values in PC2.

 We have all the variables with same sign (positive) in PC 1 (x-axis). That’s why all the variable go to the right. For PC2(y-axis), carat, depth, height, and price are all negative, and the rest are positive. That’s why all four of those variables point below the (imaginary) horizontal line at PC2 = 0. The rest point above that horizontal line. Yes, the arrows and relative lengths are as mentioned above from their loading values.

## carat depth table width length height price  
## 4880 0.30 51.0 67 4.67 4.62 2.37 945  
## 149 5.01 65.5 59 10.74 10.54 6.98 18018

Since 149 is to the right of the vertical line at PC1 = 0, a linear combination of all the variables (mostly just carat,width,length,height, and price) for these observations tend to be pretty different from observation 269.



 I would choose PC 4, as there is small decrease after PC 5 relative to the decrease from previous 4. 4 PCs explains about 98.531% of the variation.

## 'data.frame': 5000 obs. of 4 variables:  
## $ PC1: num -1.68 -1.3 1.05 1.46 3.72 ...  
## $ PC2: num 0.921 -0.5841 -2.2804 -0.4144 0.0268 ...  
## $ PC3: num -1.5675 -0.4501 -1.1116 0.0687 -1.5069 ...  
## $ PC4: num -0.122 0.069 0.547 0.413 -1.137 ...

## clusters2  
## 1 2   
## 2983 2017

## clusters3  
## 1 2 3   
## 2019 657 2324

## clusters4  
## 1 2 3 4   
## 633 1560 627 2180

## clusters5  
## 1 2 3 4 5   
## 1789 611 1339 553 708

## clusters6  
## 1 2 3 4 5 6   
## 1241 1075 551 563 517 1053

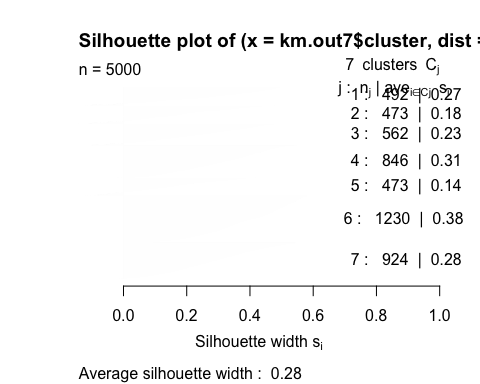
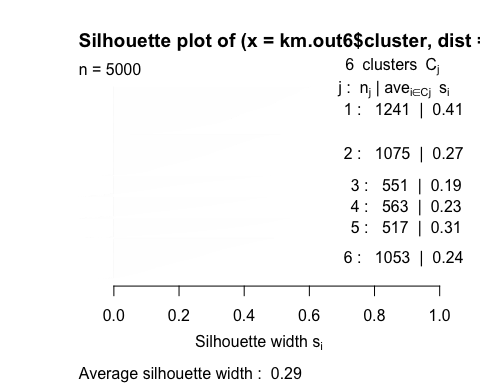
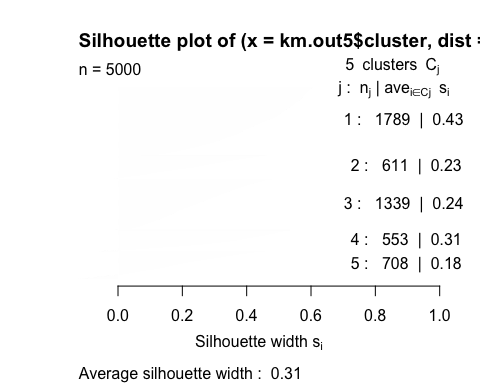
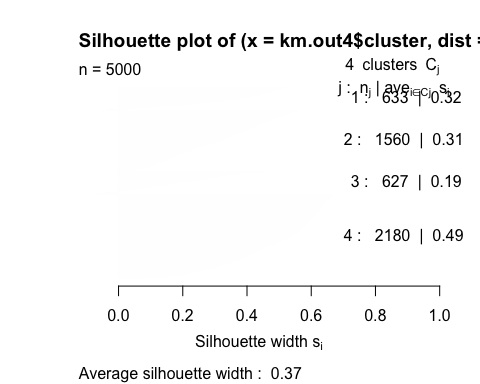
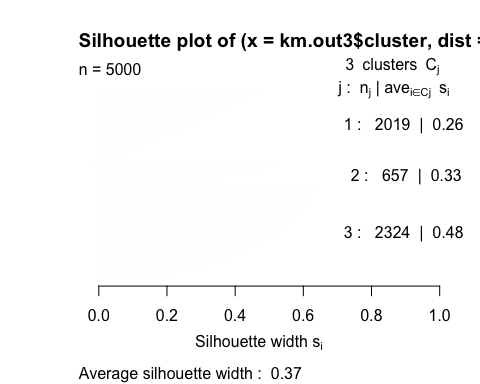
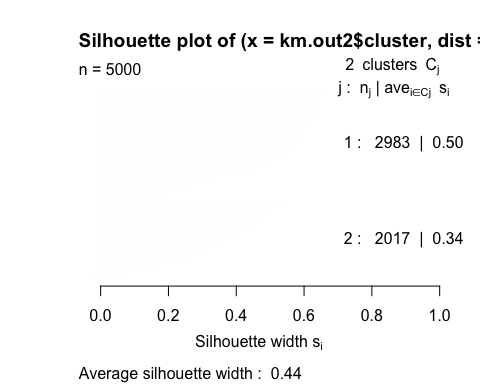
## clusters7  
## 1 2 3 4 5 6 7   
## 492 473 562 846 473 1230 924

## clusters8  
## 1 2 3 4 5 6 7 8   
## 904 720 931 824 319 442 398 462

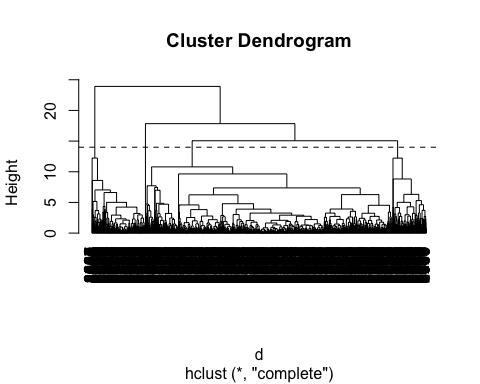
## clusters9  
## 1 2 3 4 5 6 7 8 9   
## 378 224 759 934 319 367 886 425 708

## clusters10  
## 1 10 2 3 4 5 6 7 8 9   
## 720 363 442 331 863 907 666 221 298 189

## Warning: package 'cluster' was built under R version 4.1.2



We choose K = 2 because it has the Silhouette width closest to 1.

 Using a cut height of 14, we get 4 clusters. The maximum distance between any two observations within the same cluster is 12.23235, which is less than the height at which we made our cut.

## clus.comp  
## 1 2 3 4   
## 3701 500 798 1

## [1] 10.79723

## [1] 12.22427

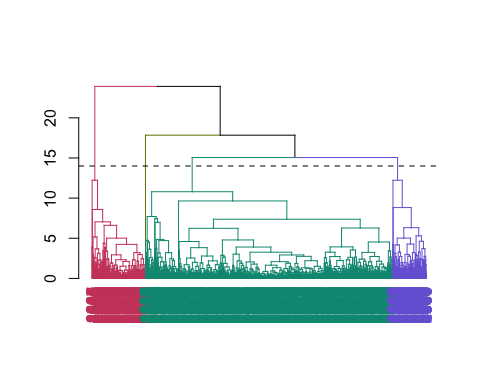
## [1] 12.23235

Ideal cut is the only category kept under cluster 4

##   
## ---------------------  
## Welcome to dendextend version 1.15.2  
## Type citation('dendextend') for how to cite the package.  
##   
## Type browseVignettes(package = 'dendextend') for the package vignette.  
## The github page is: https://github.com/talgalili/dendextend/  
##   
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues  
## You may ask questions at stackoverflow, use the r and dendextend tags:   
## https://stackoverflow.com/questions/tagged/dendextend  
##   
## To suppress this message use: suppressPackageStartupMessages(library(dendextend))  
## ---------------------

##   
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':  
##   
## cutree

 Here we can see the colors representing 4 different clusters. In which majority belongs to cluster green and only 1 belongs to cluster lime green.

## Warning: package 'caret' was built under R version 4.1.2

## Loading required package: lattice

##   
## Call:  
## lm(formula = .outcome ~ ., data = dat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -20171.9 -575.8 -178.3 395.2 8058.2   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3153.673 1365.747 2.309 0.020978 \*   
## carat 10775.924 140.775 76.547 < 2e-16 \*\*\*  
## cutGood 490.217 107.065 4.579 4.79e-06 \*\*\*  
## cutIdeal 891.308 106.706 8.353 < 2e-16 \*\*\*  
## cutPremium 818.524 102.998 7.947 2.35e-15 \*\*\*  
## `cutVery Good` 778.135 102.492 7.592 3.74e-14 \*\*\*  
## colorE -171.209 57.416 -2.982 0.002879 \*\*   
## colorF -228.132 58.304 -3.913 9.24e-05 \*\*\*  
## colorG -383.597 56.942 -6.737 1.80e-11 \*\*\*  
## colorH -934.412 59.554 -15.690 < 2e-16 \*\*\*  
## colorI -1312.167 68.619 -19.122 < 2e-16 \*\*\*  
## colorJ -2220.470 83.048 -26.737 < 2e-16 \*\*\*  
## clarityIF 5283.519 162.840 32.446 < 2e-16 \*\*\*  
## claritySI1 3767.375 137.263 27.446 < 2e-16 \*\*\*  
## claritySI2 2847.646 137.685 20.682 < 2e-16 \*\*\*  
## clarityVS1 4700.363 140.020 33.569 < 2e-16 \*\*\*  
## clarityVS2 4366.603 137.921 31.660 < 2e-16 \*\*\*  
## clarityVVS1 5077.313 148.641 34.158 < 2e-16 \*\*\*  
## clarityVVS2 5030.638 143.586 35.036 < 2e-16 \*\*\*  
## depth -84.946 16.301 -5.211 1.95e-07 \*\*\*  
## table -35.487 9.372 -3.786 0.000155 \*\*\*  
## width -956.072 111.665 -8.562 < 2e-16 \*\*\*  
## length -20.813 42.648 -0.488 0.625558   
## height 202.667 161.272 1.257 0.208929   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1093 on 4976 degrees of freedom  
## Multiple R-squared: 0.9246, Adjusted R-squared: 0.9242   
## F-statistic: 2652 on 23 and 4976 DF, p-value: < 2.2e-16

## Linear Regression   
##   
## 5000 samples  
## 9 predictor  
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 5000, 5000, 5000, 5000, 5000, 5000, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 1302.482 0.8914509 748.5153  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE

## Linear Regression   
##   
## 5000 samples  
## 9 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 4500, 4500, 4500, 4500, 4500, 4500, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 1112.698 0.9216218 729.3563  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE

## 1   
## 6053.88

We expect the price to increase by about 10775.924 unit for every one unit increase in carat, on average when all other variables are held constant.

Price is 490.217 higher on average for cut == Good, Price is 891.308 units higher on average for cut == Ideal, Price is 818.524 units higher on average for cut == Premium, and Price is 778.135 units higher on average for cut == Very good when holding all the other variables constant.

Price is 171.209 units lower on average for color == E, Price is 228.132 units lower on average for color == F, Price is 383.597 units lower on average for color == G, Price is 934.412 units lower on average for color == H, Price is 1312.167 units lower on average for color == I, and Price is 2220.470 units lower on average for color == J when holding all the other variables constant.

Price is 5283.519 units higher on average for clarity == IF, Price is 3767.375 units higher on average for clarity == SI1, Price is 2847.646 units higher on average for clarity == SI2, Price is 4700.363 units higher on average for clarity == VS1, Price is 4366.603 units higher on average for clarity == VS2, Price is 5077.313 units higher on average for clarity == VVS1, and Price is 5030 units higher on average for clarity == VVS2 when holding all the other variables constant.

We expect the price to decrease by about 84.946 unit for every one unit increase in depth, on average when all other variables are held constant.

We expect the price to decrease by about 35.487 unit for every one unit increase in table, on average when all other variables are held constant.

We expect the price to decrease by about 956.072 unit for every one unit increase in width, on average when all other variables are held constant.

We expect the price to decrease by about 20.813 unit for every one unit increase in length, on average when all other variables are held constant.

We expect the price to increase by about 202.667 unit for every one unit increase in height, on average when all other variables are held constant.

DUMMY VARIABLES ARE ARRANGED IN ALPHABETICAL ORDER. Our test RMSE is 1112.698.

When predicting the price of diamond with given descriptions (carat = 1, cut = "Ideal", color ="I", clarity = "VVS1", depth = 62, table = 63, width = 6, length = 5, height = 4 ) we get $6053.88

## k-Nearest Neighbors   
##   
## 5000 samples  
## 9 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 4500, 4500, 4500, 4500, 4500, 4500, ...   
## Resampling results across tuning parameters:  
##   
## k RMSE Rsquared MAE   
## 1 1628.429 0.8333856 902.0552  
## 3 1365.549 0.8832345 757.9656  
## 5 1331.222 0.8923463 734.3735  
## 7 1323.197 0.8952139 728.3290  
## 9 1323.155 0.8973306 718.2109  
## 11 1318.958 0.8986176 718.0017  
## 13 1317.807 0.8993923 717.0025  
## 15 1321.600 0.8995483 721.3069  
## 17 1322.795 0.9000728 721.4926  
## 19 1324.665 0.9004483 722.7665  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was k = 13.

## [1] 3198.308

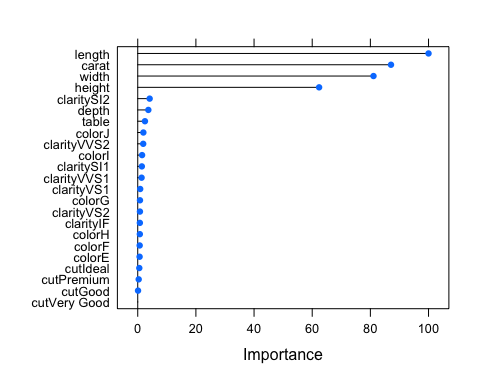
Our final model used was K = 13, in which RMSE value is 1317.807. We also used our model to predict the price of the diamond when inputted these values (carat = 1, cut = “Ideal”, color =“I”, clarity = “VVS1”, depth = 62, table = 63, width = 6, length = 5, height = 4 ), we get $3198.308. When centering and scaling, we actually don’t do a good job with our model. Therefore, we use without centering and scaling. We still don’t do a good job with our model than linear regression model.

## Random Forest   
##   
## 5000 samples  
## 9 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 4500, 4500, 4500, 4500, 4500, 4500, ...   
## Resampling results across tuning parameters:  
##   
## mtry RMSE Rsquared MAE   
## 2 1160.6855 0.9306338 692.8170  
## 3 955.6816 0.9434891 488.2799  
## 4 913.2948 0.9473364 448.5679  
## 5 897.9391 0.9488828 435.4641  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was mtry = 5.

## 1   
## 4670.545

Since we have 9 potential predictors, we can set m = 5. Trying values between 2 and 5. Our smallest RMSE is 897.9391, which is the lowest among the model used. Therefore, a better model.

## rf variable importance  
##   
## only 20 most important variables shown (out of 23)  
##   
## Overall  
## length 100.0000  
## carat 87.0947  
## width 81.0782  
## height 62.3751  
## claritySI2 4.1006  
## depth 3.6472  
## table 2.4803  
## colorJ 1.9343  
## clarityVVS2 1.8820  
## colorI 1.4921  
## claritySI1 1.4096  
## clarityVVS1 1.3200  
## clarityVS1 0.8448  
## colorG 0.7710  
## clarityVS2 0.7555  
## clarityIF 0.7308  
## colorH 0.7210  
## colorF 0.6689  
## colorE 0.6401  
## cutIdeal 0.5365



We do a much better job of predicting the price of the diamond using Random forest model than linear model or K-Nearest Neighbor model because RMSE is the lowest for Random forest model. Since, outliers were not dealt with in the preprocessing phase we don’t take the RMSE but instead take median error ipn this case for a better prediction as shown earlier using the histogram which were heavily skewed to the right. Therefore, when predicting the price of diamond with given descriptions (carat = 1, cut = "Ideal", color ="I", clarity = "VVS1", depth = 62, table = 63, width = 6, length = 5, height = 4 ) we get $4670.545 -

The most important variable was length and the least important variable was cut = very good. Method of calculation for regression problem is by measuring root mean square error.

Future work - I would use all of the 55000 observation. Play around with my categorical variable as my response variable and see which would make the most sense. Therefore, classification model will be used. I did mention about outliers, so in the grand scheme I would analyze deeper if they need to be taken out or not. I would want to know as to how those clusters were separated and as to why. There could a meaningful reason behind it.

Overall, it was definitely a fun experience using the tools to do unsupervised as well as supervised learning. I think it was a lot simple and straight forward. I plan on using this experience for any future work. Thanks