Laptop Price Amongst Varying Suppliers

Arthur Starodynov (7751472)

June 07, 2022

Contents

Intro

Intro

The purpose of this report is to figure out which brand of laptops provides the best laptop for the cheapest price.

Why might our predictive model be helpful?

The reason behind making our model useful and helpful is to guide customers to make an educated decision on the best laptop for their specific needs. Some customers might be intrigued and unsatisfied when a predictive model can show that some companies are not showing the fine print or real performance of the laptop.

```
#Read in the data
laptop.data <- read.csv('C:/Users/arthu/Dropbox/My PC (DESKTOP-9BV8I37)/Documents/PSTAT131Final/Cleaned_
set.seed(46)
```

Data

Introduction into the data set

As we all may know there are various laptop brands out there, with some of the most popular being Dell, Mac, and Lenovo. However, some of these popular laptop brands can be cheaping out customers based on their reputation that they have already built up. For example due to Apple's brand recognition and amazing customer support, the best performance laptop is mainly focused on performance features such as RAM, the Graphics Card, a Solid State Drive, and other factors. In addition, a factor that may persuade customers to buy a specific laptop would be based on online reviews and generic ratings that professionals may include. Our goal through using this data set found on Kaggle, is to be able to figure out which laptop brand really bring the most high performing laptop with the best price. The data set below has 896 observations and 23 variables. Just by looking at the column names all of them are self explanatory where they state what is being described in each column.

Here we looked at what the values of our columns are within our data set.

```
laptop.data$ram_gb<-gsub(" GB GB","",as.character(laptop.data$ram_gb))
laptop.data$ssd<-gsub(" GB","",as.character(laptop.data$ssd))
laptop.data$hdd<-gsub(" GB","",as.character(laptop.data$hdd))
laptop.data$os_bit<-gsub("-bit","",as.character(laptop.data$os_bit))

print("Table view of all categorical variables")
```

```
## [1] "Table view of all categorical variables"
```

```
## [1] "Table view for column:  brand"
##
##      acer ALIENWARE     APPLE      ASUS     Avita      DELL        HP     iball
##        58         4        28       254        18       154       142         1
##   Infinix    lenovo    Lenovo        LG        Mi MICROSOFT       MSI     Nokia
##         4         3       148         5         2         3        52         4
##    realme RedmiBook   SAMSUNG  Smartron      Vaio
##         4         3         1         3         5
## [1] "Table view for column:  model"
##
##         14a         14s          15 15-ec1105AX         15q         15s
##           1           5           3           1           3          12
##         250      250-G6        3000        3511         430     A6-9225
##           1           1           1           1           1           1
##       Alpha         AMD         APU      Aspire        Asus        ASUS
##           1           1           3          24           1          13
##      Athlon      B50-70        Book  Book(Slim)       Bravo     Celeron
##           1           1           2           2           1           3
##  Chromebook  Commercial    CompBook    ConceptD      Cosmos     Creator
##          13           1           1           1           1           1
##          DA        DELL       Delta        Dual           E     EeeBook
##           1           2           1           1           4           4
##        Envy  ExpertBook     Extensa         F17         G15          G3
##           6          16           1           1           3           1
##          G5          G7      Galaxy      GAMING        GE76        GF63
##           2           1           1           1           1           6
##        GF65        GP65        GP76        Gram          GS        GS66
##           2           1           1           5           1           1
##          HP     Ideapad     IdeaPad     IDEAPAD      INBook     Inpiron
##           6          32          37           1           3           1
##    Inspiron    INSPIRON    Insprion       Intel      Katana      Legion
##          77           2           1           4           4          11
##      Lenovo       Liber     MacBook     Missing      Modern       Nitro
##           1          13          28          95          10           5
##    Notebook        Omen        OMEN    Pavilion     Pentium    Predator
##           3           2           5          38           5           9
##    Prestige         Pro       Pulse        PURA    PureBook         Rog
```

```
##           5           4           3           4           4           1
##         ROG       Ryzen          SE     Spectre        Spin     Stealth
##          31          35           1          12           1           2
##      Summit     Surface       Swift       Sword      t.book   Thinkbook
##           1           3           8           2           3           4
##   ThinkBook    Thinkpad    ThinkPad     Thinpad  Travelmate         TUF
##           4           1          10           1           3          10
##         v15         V15        Vivo    VivoBook  VivoBook14      Vostro
##           1           1           1          89           1          33
##        WF65          X1        x360        X390         XPS        Yoga
##           1           1           2           1           5          14
##     Zenbook     ZenBook    Zephyrus
##           7          22           5
## [1] "Table view for column:  processor_brand"
##
##      AMD    Intel       M1 MediaTek Qualcomm
##      208      660       24        3        1
## [1] "Table view for column:  processor_name"
##
## A6-9225 Processor          APU Dual       Athlon Dual      Celeron Dual
##                 1                 7                 2                24
##              Core           Core i3           Core i5           Core i7
##                 1               170               312               112
##           Core i9           Core m3         Dual Core   Ever Screenpad
##                 8                 1                 3                 2
##       GeForce GTX       GeForce RTX       GEFORCE RTX   Genuine Windows
##                 2                 4                 1                 3
##         Hexa Core                M1 MediaTek Kompanio      Pentium Quad
##                 2                24                 3                14
##    Pentium Silver              Quad             Ryzen           Ryzen 3
##                 2                 1                 1                26
##           Ryzen 5           Ryzen 7           Ryzen 9     Snapdragon 7c
##                85                58                26                 1
## [1] "Table view for column:  processor_gnrtn"
##
##    10th    11th    12th     4th     7th     8th     9th Missing
##     246     346       3       1      12      43       6     239
## [1] "Table view for column:  ram_gb"
##
##  16  32   4   8
## 180   3 259 454
## [1] "Table view for column:  ram_type"
##
##    DDR3    DDR4    DDR5  LPDDR3  LPDDR4 LPDDR4X
##      12     760       8      14      36      66
## [1] "Table view for column:  ssd"
##
##    0 1024  128 2048  256 3072   32  512
##  151  111   12    2  201    1    1  417
## [1] "Table view for column:  hdd"
##
##    0 1024 2048  512
##  666  164    1   65
## [1] "Table view for column:  os"
```

```
##
##     DOS     Mac Windows
##      36      28     832
## [1] "Table view for column:  os_bit"
##
##  32  64
## 135 761
## [1] "Table view for column:  graphic_card_gb"
##
##   0   2   4   6   8
## 631  69 138  40  18
## [1] "Table view for column:  weight"
##
##     Casual     Gaming ThinNlight
##        566         39        291
## [1] "Table view for column:  display_size"
##
##    12.2      13    13.3    13.4      14    14.1    14.2    14.9   14.96      15
##       2       4      40       1     131       6       3       1       7       3
##    15.6      16    16.1    16.2    17.3 Missing
##     218     135       1       3       9     332
## [1] "Table view for column:  warranty"
##
##   0   1   2   3
## 332 521  30  13
## [1] "Table view for column:  Touchscreen"
##
##  No Yes
## 793 103
## [1] "Table view for column:  msoffice"
##
##  No Yes
## 606 290
```

```
##           brand           model processor_brand  processor_name processor_gnrtn
##               0              95               0               0             239
##          ram_gb        ram_type             ssd             hdd              os
##               0               0               0               0               0
##          os_bit graphic_card_gb          weight    display_size        warranty
##               0               0               0             332               0
##     Touchscreen        msoffice    latest_price       old_price        discount
##               0               0               0               0               0
##     star_rating         ratings         reviews
##               0               0               0
```


```
##           brand           model processor_brand  processor_name processor_gnrtn
##               0              95               0               0             239
##          ram_gb        ram_type             ssd             hdd              os
##               0               0               0               0               0
##          os_bit graphic_card_gb          weight    display_size        warranty
##               0               0               0             332               0
##     Touchscreen        msoffice    latest_price       old_price        discount
##               0               0               0               0               0
##     star_rating         ratings         reviews
```

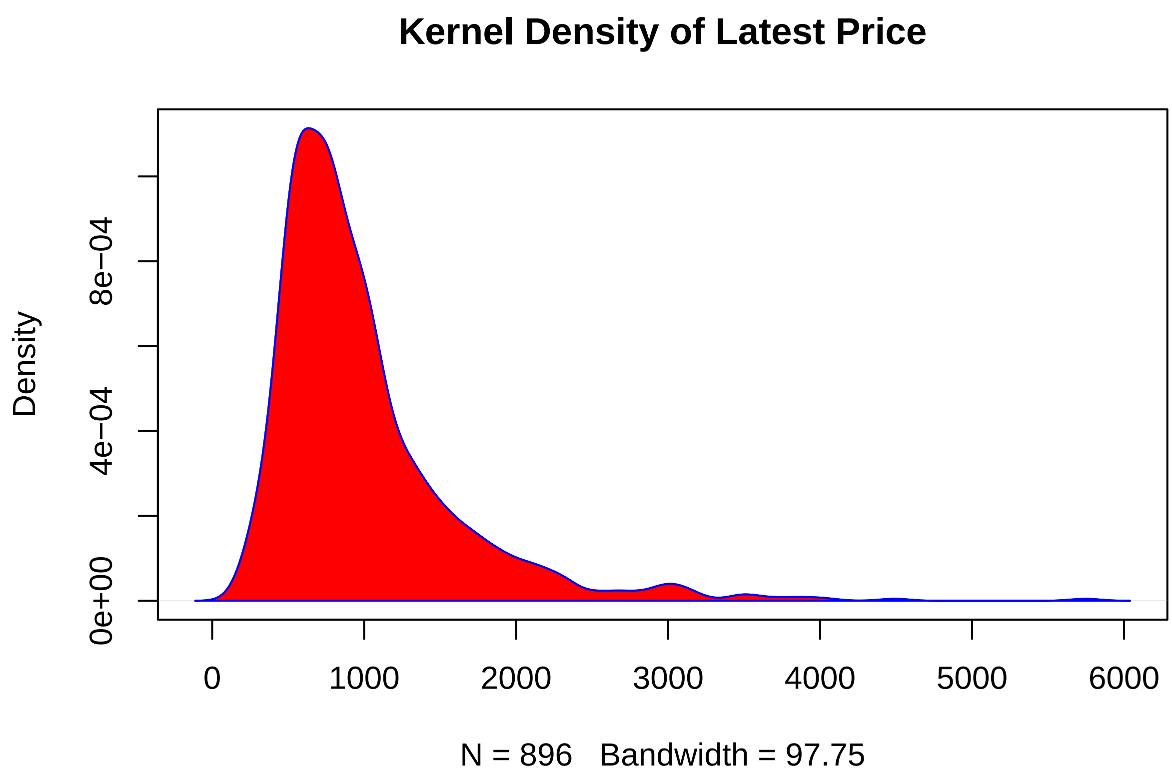
```
##                      0                  0                 0
```

Visualization

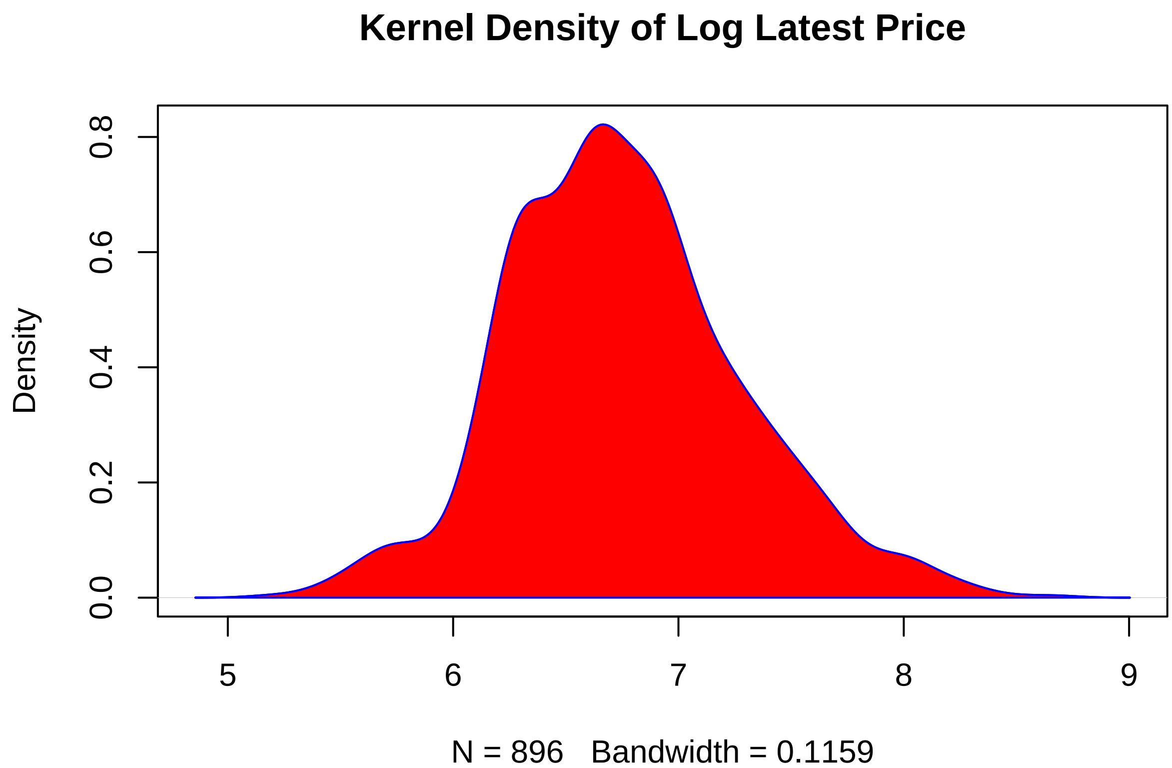
First Visualization

We first look at our target variable, Latest Price, convert it into USD $ to make it easier to visualize and notice that the variable is right-skewed. Hence we apply a log transform and make a new target variable which is the log_transform of latest price.

```
laptop.data$latest_price <- laptop.data$latest_price*0.013
laptop.data$old_price <- laptop.data$old_price*0.013
d <- density(laptop.data$latest_price)
plot(d, main="Kernel Density of Latest Price")
polygon(d, col="red", border="blue")
```




```
laptop.data$log_latest_price <- log(laptop.data$latest_price)
d <- density(laptop.data$log_latest_price)
plot(d, main="Kernel Density of Log Latest Price")
polygon(d, col="red", border="blue")
```



Cleaning of the data

We noticed that there were a few variables within our data set that had missing observations, so we needed to create a new column that would indicate if display size was given, and gave it either a "Yes or a "No value. In addition we wanted to use all the available data we had so any missing records were replaced with an average of the column, which were eventually grouped to make some levels, furthering the easiness of modeling later. In the clean data, we wanted to drop many of the columns that included N/A which were seen as processor_gnrtn, discount, model, old_price. Also we wanted to only maintain the same currency so we removed the discount, and old_price columns to not get clustered. With all the cleaning we made a new dataframe.

```
laptop.data["display_size_given"] <- laptop.data$display_size
laptop.data$display_size_given[!is.na(laptop.data$display_size_given)] <- "Yes"
laptop.data$display_size_given[is.na(laptop.data$display_size_given)] <- "No"
laptop.data$display_size <- as.numeric(laptop.data$display_size)
laptop.data$display_size <- round(na.aggregate(laptop.data$display_size),2)

laptop.data$brand <- tolower(laptop.data$brand)

laptop.data$brand <- ifelse(laptop.data$brand %in% c("acer",
                                                    "apple", "asus", "dell",
                                                    "hp","lennovo", "msi"),
                            laptop.data$brand,"other")
```

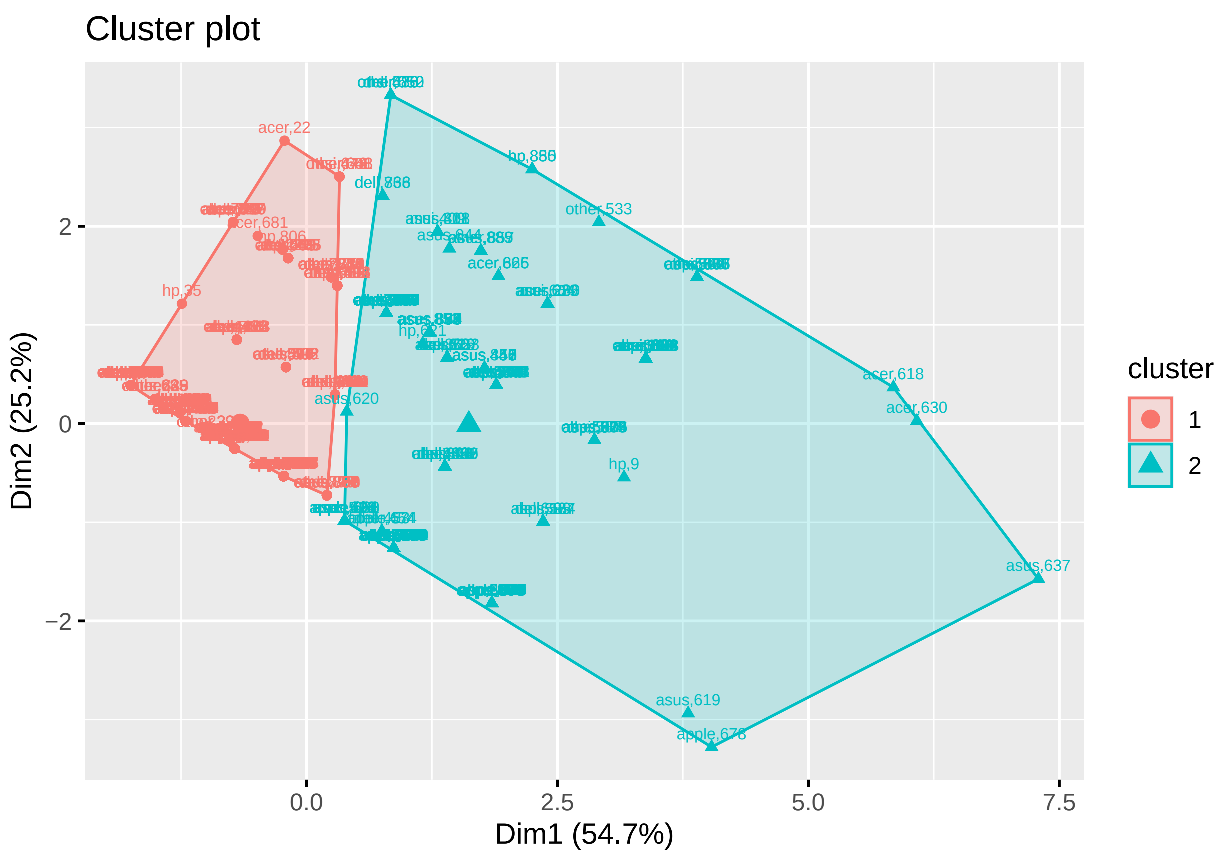
```r
laptop.data$processor_brand <- tolower(laptop.data$processor_brand)

laptop.data$processor_brand <- ifelse(laptop.data$processor_brand %in% c("amd",
                                                                        "intel"),
                            laptop.data$processor_brand,"other")

laptop.data$ram_gb_cat <- ifelse(as.numeric(laptop.data$ram_gb)<=8,
                            "less_than_8","greater_than_8")

laptop.data$ssd<-as.numeric(laptop.data$ssd)
laptop.data$ssd_cat <- ifelse(laptop.data$ssd < 1024.0,
                            "less_than_1gb","greater_than_1gb")

laptop.data$hdd<-as.numeric(laptop.data$hdd)
laptop.data$hdd_cat <- ifelse(laptop.data$hdd < 1024.0,
                            "low_hdd","high_hdd")

laptop.data$processor_name <- tolower(laptop.data$processor_name)

laptop.data$processor_name <- ifelse(laptop.data$processor_name %in% c("celeron dual",
                                                                      "core i3", "core i5", "core i7",
                                                                      "m1","pentium quad",
                                                                      "ryzen 3","ryzen 5",
                                                                      "ryzen 7","ryzen 9"),
                            laptop.data$processor_name,"other")
laptop.data.clean <-laptop.data[c(1,3,4,7,10,11,12,13,14,15,16,17,21,22,23,24,25,26,27,28)]
```

Modeling

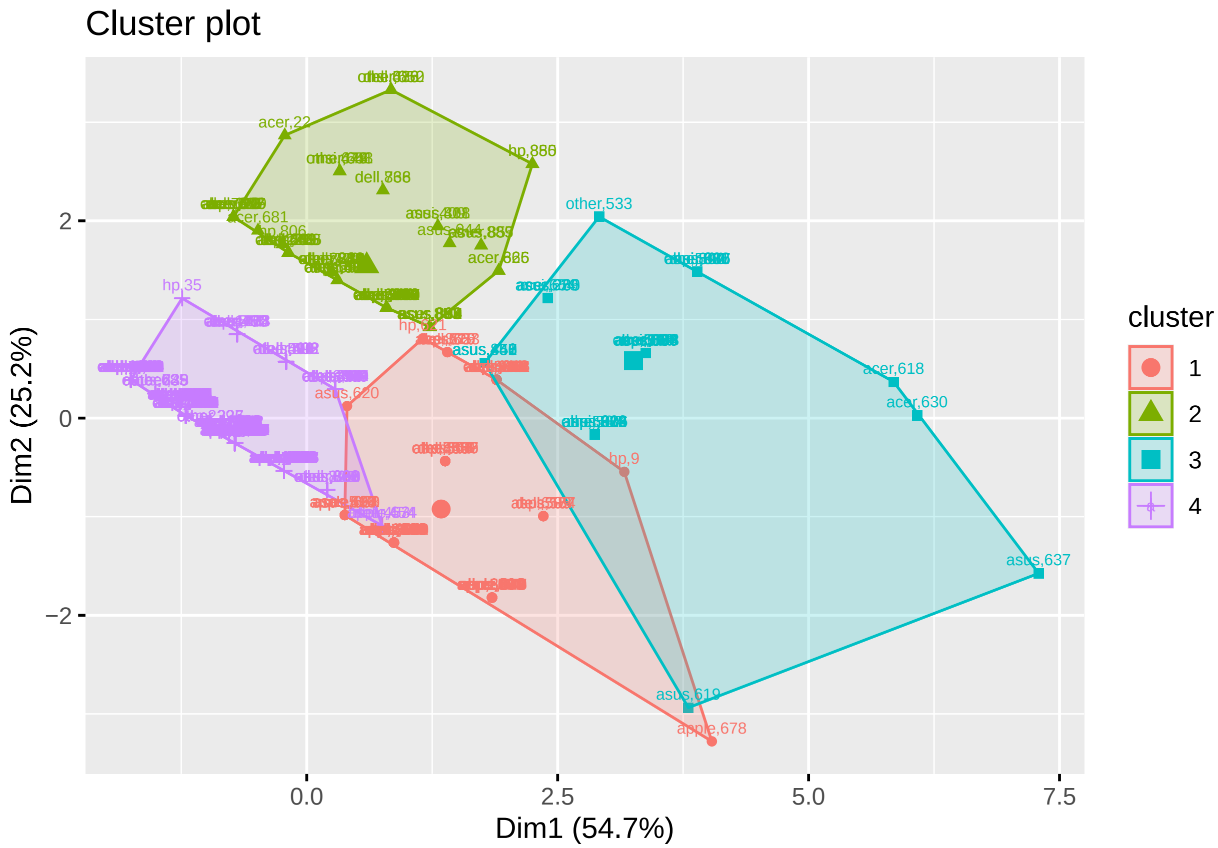
Clustering of Data

Instead of using k folding technique we thought it would be more beneficial to be able to analyze using clustering analysis on the data set. Looking at the data set, we know that RAM, SSD, and the graphic card memory are performance attributes that customers will consider when they are in the market of buying a laptop. Hence, we decided to cluster upon these three columns.

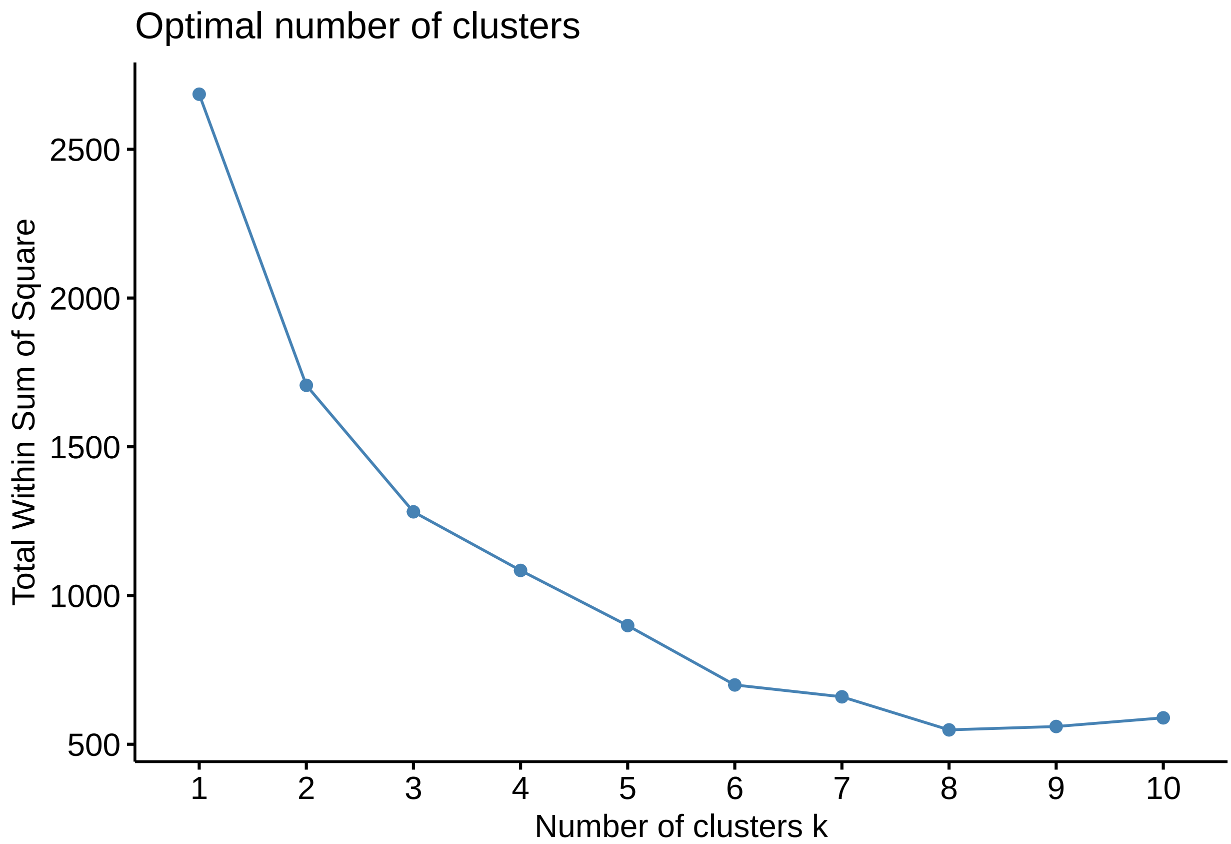
```r
tmp.laptop.data.clean <- laptop.data[,c(6,8,12)]
tmp.laptop.data.clean$ram_gb<- as.numeric(tmp.laptop.data.clean$ram_gb)
tmp.laptop.data.clean$ssd<- as.numeric(tmp.laptop.data.clean$ssd)
tmp.laptop.data.clean$graphic_card_gb<- as.numeric(
  tmp.laptop.data.clean$graphic_card_gb)

tmp.laptop.data.clean <- scale(tmp.laptop.data.clean)
rownames(tmp.laptop.data.clean) <- paste(laptop.data.clean$brand,",",
                              c(1:length(laptop.data.clean$brand)),sep="")

k1 <- kmeans(tmp.laptop.data.clean, centers = 2, nstart = 25)
fviz_cluster(k1, data = tmp.laptop.data.clean, labelsize=6)
```



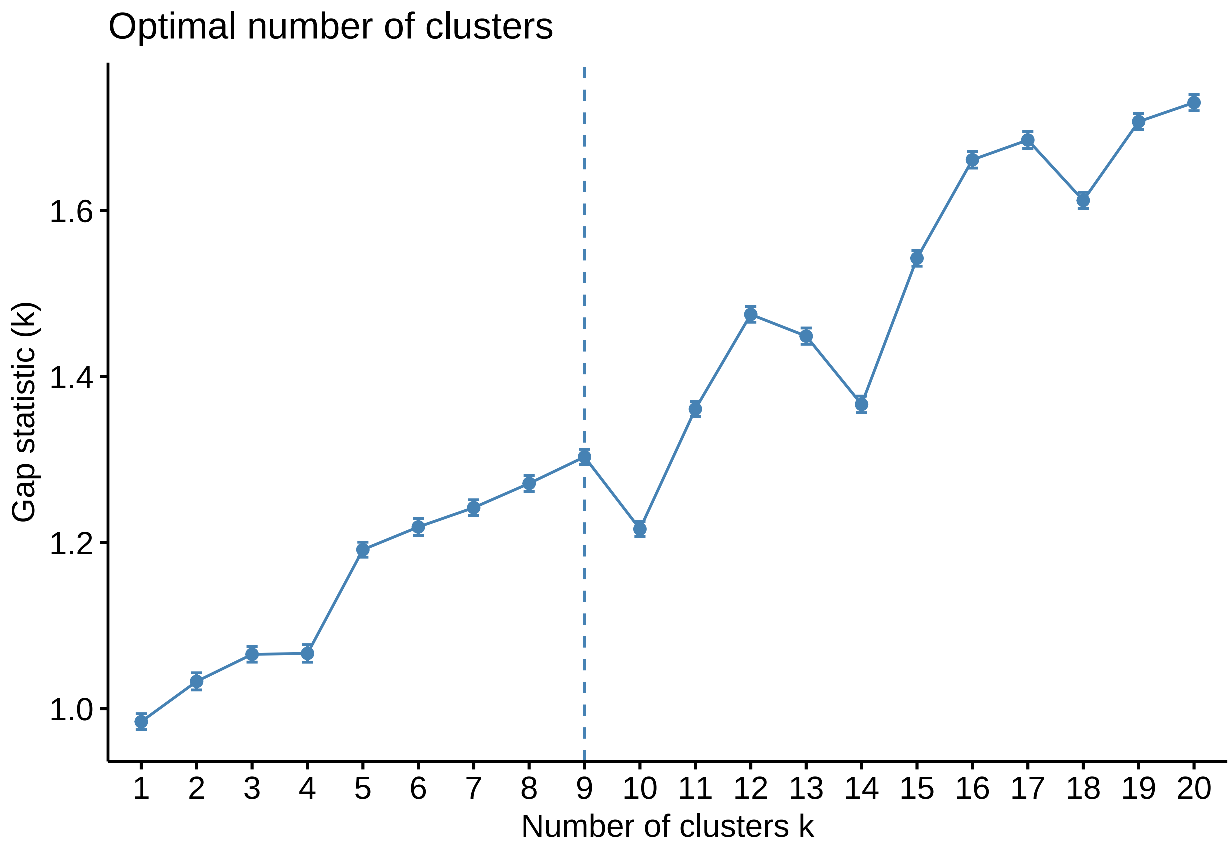

```
k2 <- kmeans(tmp.laptop.data.clean, centers = 4, nstart = 25)
fviz_cluster(k2, data = tmp.laptop.data.clean, labelsize=6)
```




```r
fviz_nbclust(tmp.laptop.data.clean, kmeans, method = "wss")
```

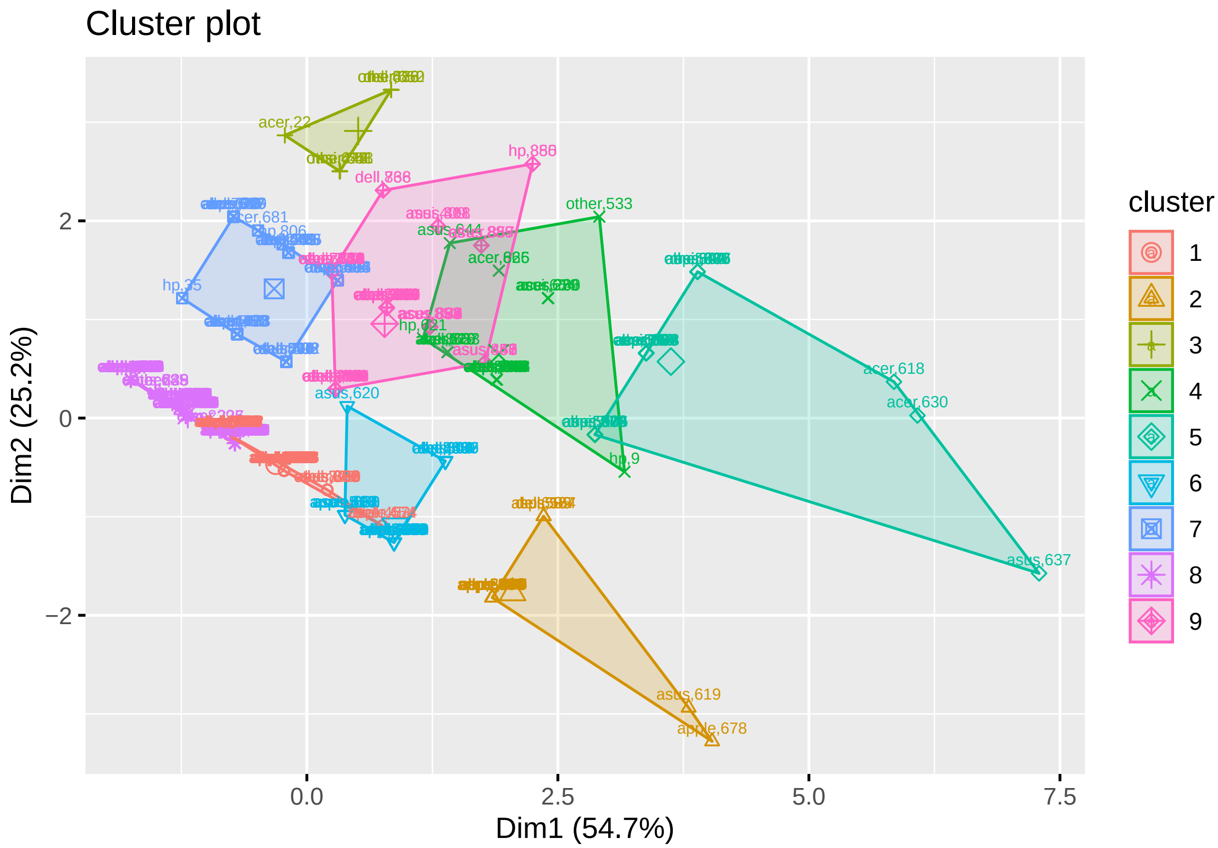



```
gap_stat <- clusGap(tmp.laptop.data.clean, FUN = kmeans, nstart = 25,
                    K.max = 20, B = 50)
fviz_gap_stat(gap_stat)
```



From the above results we notice that we were successful in clustering our laptop variables into separable clusters. Using a classic clustering method to find the optimal amount of them known as the elbow method, we figured out that the optimal amount of clusters is 9, so using 9 centers and the k means method we visualized these variables.

```
k3 <- kmeans(tmp.laptop.data.clean, centers = 9, nstart = 25)
fviz_cluster(k3, data = tmp.laptop.data.clean, labelsize=6)
```



Below we printed the number of observations in each cluster which we will use to further visualize laptop price.

```
laptop.data.clean['cluster'] <- as.factor( k3$cluster)
table(laptop.data.clean$cluster)
```

```
##
##   1   2   3   4   5   6   7   8   9
## 256  33  11  37  37  76  51 283 112
```

Training the Data

Here we wanted to to explore the data further so we implemented the technique to be able to fix the data into a training and testing data set.

```
train.indices <- sample(nrow(laptop.data.clean), floor(nrow(laptop.data.clean)/1.5), replace = FALSE)
validation.indices <- seq(nrow(laptop.data.clean))[-train.indices]
pred.laptop.train <- laptop.data.clean[train.indices,]
pred.laptop.train <- pred.laptop.train[,c(16,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,17,18,19,20,21)]
pred.laptop.validation <- laptop.data.clean[validation.indices,]
pred.laptop.validation <- pred.laptop.validation[,c(16,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,17,18,19,20,
```

Model Peformance

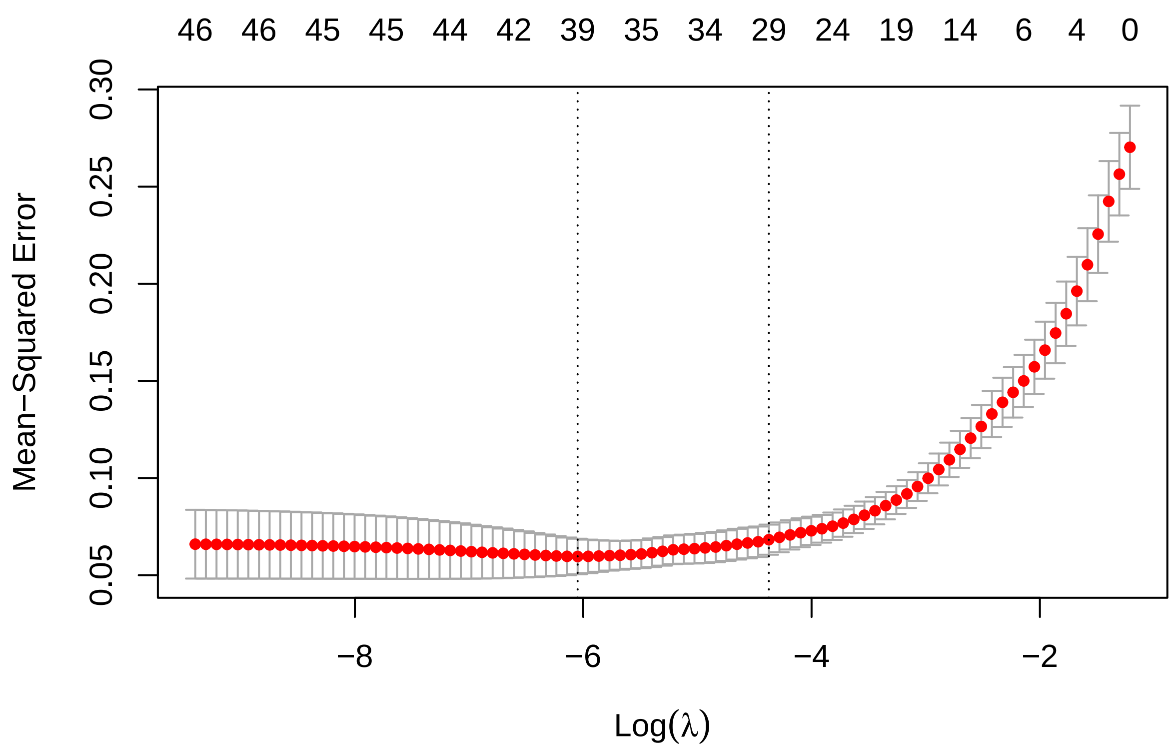
Analysis

Below we create our "recipe" or formula which we will use to fit our training data set. We wanted to use lasso regression to be able make the model have fewer features. We also applied a stepwise reggression approach onto our model and compared the two.

```
glmnet.formula <- as.formula(log_latest_price ~ .)
glmnet.design.matrix <- model.matrix(glmnet.formula, data = pred.laptop.train)
dim(glmnet.design.matrix)
```

```
## [1] 597  49
```

```
glmnet.cv.laptop.out <- cv.glmnet(glmnet.design.matrix,
                        y = pred.laptop.train$log_latest_price,
                        family = c("gaussian"),
                        type.measure="mse", # the model selection criteria
                        alpha = 1) # The Lasso regression


plot(glmnet.cv.laptop.out)
```



```r
saved.coef <- coef(glmnet.cv.laptop.out, s=c("lambda.1se"))

dim(saved.coef)
```

```
## [1] 50  1
```

```r
chosen.vars <- data.frame(name = saved.coef@Dimnames[[1]][saved.coef@i + 1],
                          coefficient = saved.coef@x)


print(paste("The lasso regression chose", dim(chosen.vars)[1]-1,
            "variables and 1 intercept"))
```

```
## [1] "The lasso regression chose 29 variables and 1 intercept"
```

```r
print(saved.coef)
```

```
## 50 x 1 sparse Matrix of class "dgCMatrix"
##                                         s1
## (Intercept)                   6.7416819323
## (Intercept)                   .
## brandapple                    0.5558859411
## brandasus                    -0.0020370338
## branddell                     0.0048670322
## brandhp                       .
## brandmsi                      .
## brandother                    .
## processor_brandintel          .
## processor_brandother          .
## processor_namecore i3        -0.0097100993
## processor_namecore i5         0.1985940376
## processor_namecore i7         0.2869211681
## processor_namem1              0.1365601575
## processor_nameother           .
## processor_namepentium quad -0.2503282289
## processor_nameryzen 3        -0.0362703686
## processor_nameryzen 5         0.0907451217
## processor_nameryzen 7         0.1362783961
## processor_nameryzen 9         0.3215072423
## ram_typeDDR4                  .
## ram_typeDDR5                  0.3005891972
## ram_typeLPDDR3                0.3802361275
## ram_typeLPDDR4                .
## ram_typeLPDDR4X               .
## osMac                         0.0012034446
## osWindows                    -0.2671547136
## os_bit64                      .
## graphic_card_gb               0.0448758096
## weightGaming                  .
## weightThinNlight              0.0083628155
## display_size                  0.0332215507
## warranty                      0.0263151957
```

```
## TouchscreenYes              0.2750043592
## msofficeYes                 .
## star_rating                -0.0281194638
## ratings                    -0.0000391211
## reviews                     .
## display_size_givenYes       .
## ram_gb_catless_than_8      -0.1557618406
## ssd_catless_than_1gb       -0.2561305868
## hdd_catlow_hdd              .
## cluster2                    .
## cluster3                    0.5303553466
## cluster4                    .
## cluster5                    0.1315500117
## cluster6                    0.0068286880
## cluster7                    .
## cluster8                   -0.2059538924
## cluster9                    .
```

```
glmnet.formula2 <- as.formula(log_latest_price ~ .)
glmnet.design.matrix.validation <- model.matrix(glmnet.formula2,
                                                data = pred.laptop.validation)

validation.preds.regularizedreg <- predict(glmnet.cv.laptop.out,
                                  newx = glmnet.design.matrix.validation,
                                  type = c("response"))
```

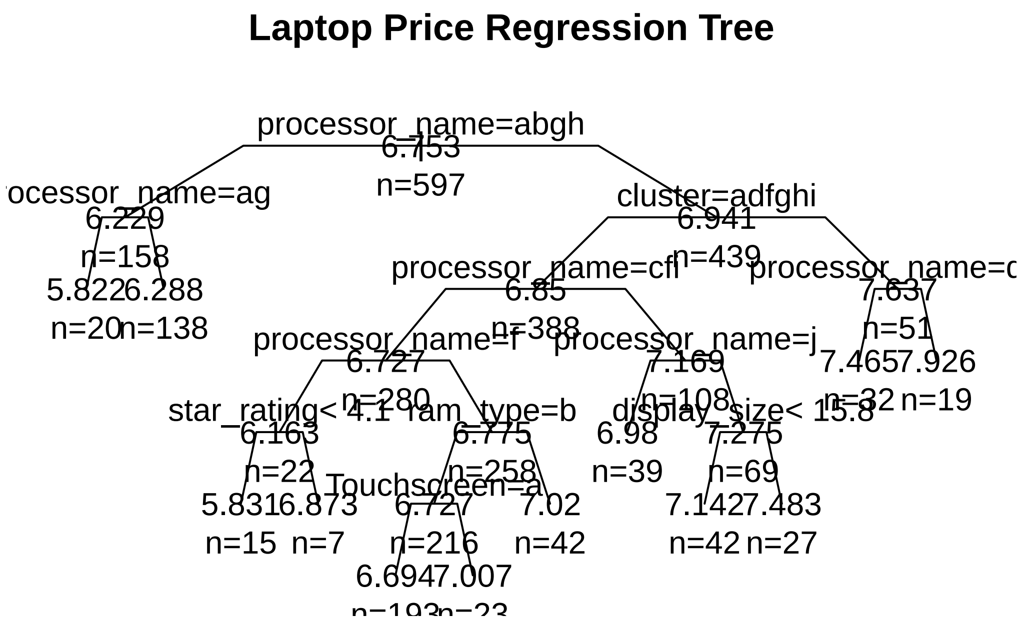
We can clearly see that the lasso model was fairly accurate, the stepwise regression model was still superior.

Decision Tree

We wanted further visualization of the laptop price variable, hence we made a regression tree to visualize this.

```
tree.out.1 <-rpart(log_latest_price ~ ., data = pred.laptop.train,
                   parms  = list(split="information"),
                   control = rpart.control(minsplit=20))
```

```
#Create a plot of the classification tree.
#Code to plot the tree.
plot(tree.out.1, uniform=TRUE, branch=0.6, margin=0.05)
text(tree.out.1, all=TRUE, use.n=TRUE)
title("Laptop Price Regression Tree")
```



Looking at the results we were able to see that we got an R-squared value of around .68 for this model

Random Forest Model

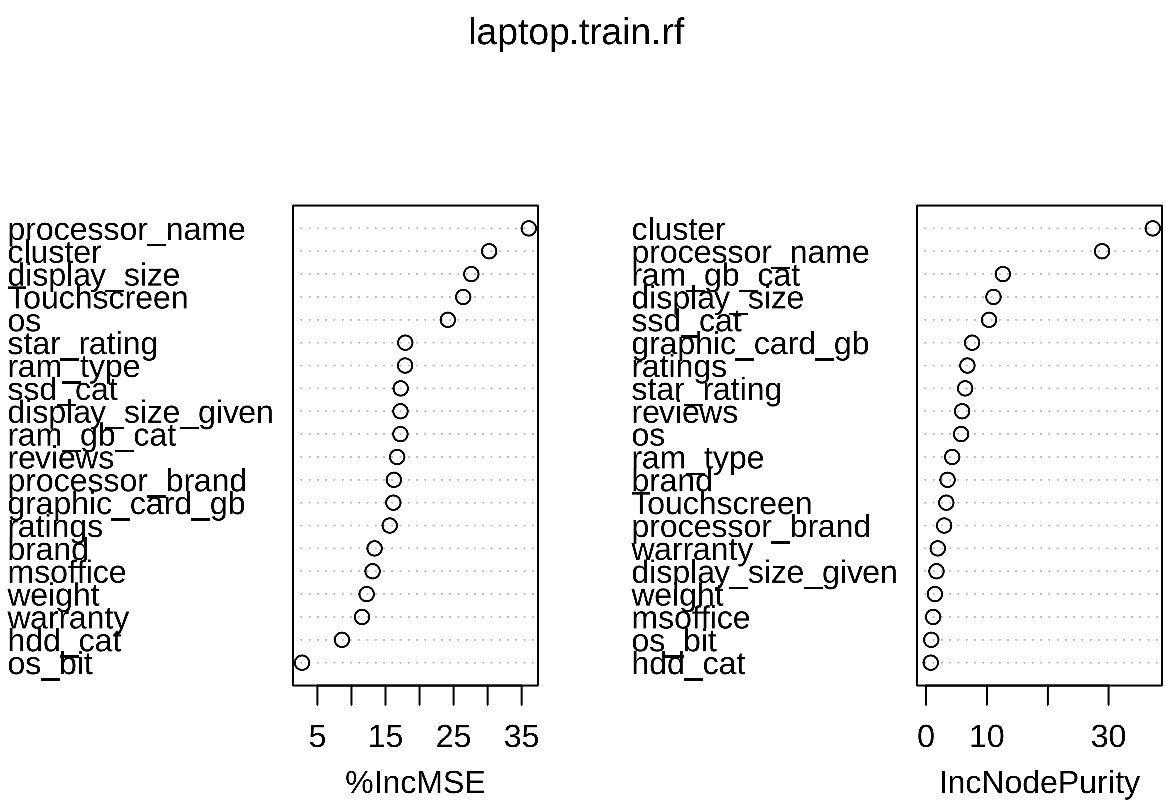
We then used the random forest model to see if this would be a better model, and in fact was.

```
laptop.train.rf <- randomForest(log_latest_price ~ .,
                      data = pred.laptop.train,
                      importance=TRUE)

print(laptop.train.rf$importance)
```

```
##                       %IncMSE IncNodePurity
## brand             0.0059684506     3.5434391
## processor_brand   0.0153051988     2.9744342
## processor_name    0.0923899754    28.9346092
## ram_type          0.0087423660     4.3039938
## os                0.0216309357     5.7705393
## os_bit            0.0006633697     0.8664901
## graphic_card_gb   0.0204087236     7.5760371
## weight            0.0042694026     1.4592317
## display_size      0.0275332441    11.0879834
## warranty          0.0036213301     1.9348522
## Touchscreen       0.0095114446     3.3301613
## msoffice          0.0028293931     1.1686366
```

```
## star_rating        0.0149630137     6.4254561
## ratings            0.0206146725     6.7990646
## reviews            0.0157708219     5.9301584
## display_size_given 0.0064951843     1.7197173
## ram_gb_cat         0.0225982440    12.6229860
## ssd_cat            0.0214271808    10.3615372
## hdd_cat            0.0017975182     0.7782949
## cluster            0.0872089169    37.2697937
```

```
varImpPlot(laptop.train.rf)
```



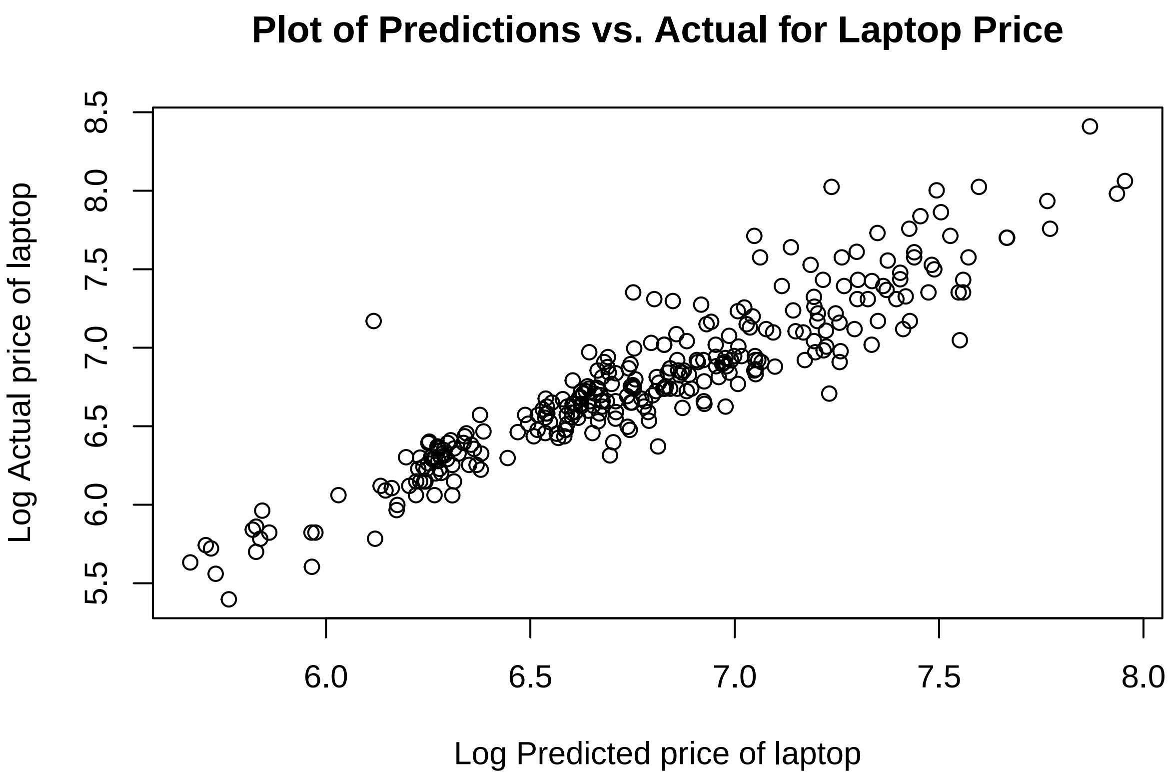

```
optimum <- which.max(laptop.train.rf$importance[,"%IncMSE"])
opt.var <- laptop.train.rf$importance[optimum,0,drop=FALSE]

print("The most predictive variable with regard to price is:")
```

```
## [1] "The most predictive variable with regard to price is:"
```

```
print(opt.var)
```

```
##
## processor_name
```

```
val.preds.rf <- predict(laptop.train.rf, # The forest
                 newdata = pred.laptop.validation, # The values of x to do prediction at
                 type = c("response")
                 )

# Code to plot the predictions against the actual values

plot(val.preds.rf, pred.laptop.validation$log_latest_price,
     main = "Plot of Predictions vs. Actual for Laptop Price",
     xlab = "Log Predicted price of laptop",
     ylab = "Log Actual price of laptop")
```



Looking at the summary of the model we were able to get an R-squared value of .78

Best Fitting Model

Through the models created it was clear that the Random Forest provided us with the best analysis of the laptop price data. This was seen as it had the highest R-squared value of around .78. It was seen that processor name variables followed by cluster and ssd_cat were the most predictive towards laptop price.

Conclusion

Conclusion

This data set was interesting to visualize and to implement a clustering methodolgy seemed to be more practice than using cross folding. Seeing our best fitting moel was the random forest model, was great, however, was still not as accurate we would have liked it to be. Being able to model this data set and see some correlation of variables to predict laptop price could be used for consumers to have a second thought of which laptop brand they should consider purchasing. In the future if we wanted to expand on the data set, we could instead try using cross folding to mode the data, as well as using an XGboost model to look further at the prediction of laptop price.