Diamond Price Analysis
Harvard Data Science Capstone Project

Dania Zhu

June 30 2021

```
library(tidyverse)  # basic
library(tidyr)      # basic
library(caret)      # models
library(corrplot)  # correlation
library(data.table)
library(dplyr)
library(kableExtra)
library(ggplot2) # Plot
library(gridExtra)  # Plot
library(forcats)
library(matrixStats)  # Matrix
library(rpart)
library(stringr)
library(ggcorrplot)
options(digits = 3, warn = -1)
```

1. Introduction

This dataset contains price and other attributes of almost 54,000 diamonds downloaded from Kaggle beginner dataset. There are 10 attributes included in the dataset, here is Feature description:

Price: in US dollars ($326–$18,823), it is the target column.

The 4 Cs of Diamonds:

Carat (0.2 – 5.01) is the diamond’s physical weight measured in metric carats. One carat equals 1/5 gram and is subdivided into 100 points. Carat weight is the most objective grade of the 4Cs.

Cut (Fair, Good, Very Good, Premium, Ideal) In determining the quality of the cut, the diamond grader evaluates the cutter’s skill in the fashioning of the diamond. The more precise the diamond is cut, the more captivating the diamond is to the eye.

Color, from J (worst) to D (best). The colour of gem-quality diamonds occurs in the range from colourless to light yellow or light brown. Colourless diamonds are the rarest. Other natural colours (blue, red, pink) are known as "fancy," and their colour grading is different than from white colorless diamonds.

Clarity (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best)), Diamonds can have internal characteristics known as inclusions or external characteristics known as blemishes. Diamonds without inclusions or blemishes are rare; however, most characteristics can only be seen with magnification.

Dimensions:
x length in mm (0 – 10.74)
y width in mm (0 – 58.9)
z depth in mm (0 – 31.8)

Depth: this is Diamonds height (in mm) measured from the culet (bottom tip) to the table (flat, top surface).

Table width: the top of the diamond relative to widest point (43 – 95)

A diamonds table refers to the flat facet of the diamond seen when the stone is face up. The main purpose of a diamond table is to refract entering light rays and allow reflected light rays from within the diamond to meet the observer's eye. The ideal table cut diamond will give the diamond stunning fire and brilliance.

```
dat <- read.csv("diamonds.csv", header=TRUE, sep=",")
```

```
str(dat)
```

```
## 'data.frame':    53940 obs. of  11 variables:
##  $ X      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
```

```
##  $ cut     : chr  "Ideal" "Premium" "Good" "Premium" ...
##  $ color   : chr  "E" "E" "E" "I" ...
##  $ clarity: chr  "SI2" "SI1" "VS1" "VS2" ...
##  $ depth   : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table   : num  55 61 65 58 58 57 57 55 61 61 ...
##  $ price   : int  326 326 327 334 335 336 336 337 337 338 ...
##  $ x       : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y       : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z       : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

```
dim(dat)
```

```
## [1] 53940    11
```

2. Data description

We first look at the distinct cut, color, clarity for types and numbers. There are 5 color types, 7 cut types, and 8 clarities. Diamonds sold at price from lowest $326 to highest $18823, with 1st - 3rd Quantile from $950 to $5324, price below $950 is Low.

```
dat %>% select (cut, color, clarity) %>% summarize
(number_distinct_cuts=n_distinct(dat$cut),
number_distinct_color= n_distinct(dat$color),
number_distinct_clarity = n_distinct(dat$clarity))
```

```
##   number_distinct_cuts number_distinct_color number_distinct_clarity
## 1                    5                     7                       8
```

```
# Price summary, 326 - 18823
summary(dat$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     326     950    2401    3933    5324   18823
```

```
# Carat summary, 0.2 - 5.01
summary(dat$carat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.20    0.40    0.70    0.80    1.04    5.01
```

One Diamond sold at $18823, 12 sold at lowest price. Diamond weight is from smallest 0.21 carat to largest 5.1 carat.

```
# Diamond sold at highest price
dat %>% filter(price==18823) %>% nrow()
```

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

```r
# Diamond sold at lowest price
dat %>% filter(carat==0.2) %>% nrow()
```

```
## [1] 12
```

Next we like to see for the diamond sold at max price, how the 4C is (carat, cut, color, clarity), the data showed it is not very big, just the average 2.29, the cut is Premium, top-level cutting, suggests the buyer like the luminous quality, and this could be sold at Sale-event or other unknown factors drove price high.

```r
# check Max sold at Price high, but Avg carat, avg color, good in cutting
& clarity
dat %>% filter(price==18823) %>% select (carat, price, cut, clarity, color
)
```

```
##   carat price     cut clarity color
## 1  2.29 18823 Premium     VS2     I
```

Next we like to see counts in each category of Diamonds cut, color, clarity and distribution. We find the majority is the Upper 3 classes of cutting: Ideal, Premium, Very Good, together they are around 85%.

```r
t1<-dat %>% group_by(cut)%>%
  summarise(count_cut=n(),percent=n()/length(dat$cut) * 100) %>%
  arrange(desc(percent))
print.data.frame(t1)
```

```
##         cut count_cut percent
## 1     Ideal     21551   39.95
## 2   Premium     13791   25.57
## 3 Very Good     12082   22.40
## 4      Good      4906    9.10
## 5      Fair      1610    2.98
```

4 colors G, E, F, H consist about 70% of total percentage, Best color D is 12%, color distribution is more variant than cut.

```r
t2<-dat %>% group_by(color)%>% summarise(count_color=n(),
percent=n()/length(dat$color) *100) %>% arrange(desc(percent))
print.data.frame(t2)
```

```
##   color count_color percent
## 1     G       11292   20.93
## 2     E        9797   18.16
## 3     F        9542   17.69
## 4     H        8304   15.39
## 5     D        6775   12.56
```

```
## 6     I          5422    10.05
## 7     J          2808     5.21
```

For clarity, around 45% of the Diamonds are in middle class of clarity (SI1, VS2), the worst clarity (I1) is 1.3%, best clarity (IF) is 3.3%, both worst and best are low in their percentage. VVS1, VVS2 (2nd Best) they together are about 15%

```
t3<-dat %>% group_by(cut)%>% summarise(count_cut=n(),
percent=n()/length(dat$cut) * 100) %>% arrange(desc(percent))
print.data.frame(t3)
```

```
##           cut count_cut percent
## 1       Ideal     21551   39.95
## 2     Premium     13791   25.57
## 3 Very Good     12082   22.40
## 4        Good      4906    9.10
## 5        Fair      1610    2.98
```

3. Data Cleaning

Before we start data analysis and prediction, we need to get rid of Invalid data (e.g: NULLS, zeros, etc). There are rows in Zero Value in dimension x,y,z, probably due to data entry or measure error, we need to remove them. 2nd we need to remove outliers in dataset.

```
# remove zero rows in x, y, z
dat %>% filter(x==0 | y==0 | z==0) %>% nrow()
```

```
## [1] 20
```

```
dat <- dat %>% filter(dat$x !=0 & dat$y!=0 & dat$z!=0)
dim(dat)
```

```
## [1] 53920    11
```

```
# remove outliers
dat <- dat %>% filter(x>=3.73 & x<=10.74) %>%
  filter(y>=3.7 & y<=58.9) %>%
  filter(z>=1.1 & z<=31.8) %>%
  filter(depth>=43.0 & depth<=79.0) %>%
  filter(table>=43.0 & table<=95.0)
dim(dat)
```

```
## [1] 53918    11
```

3rd, we need to convert 3 categorical data to numeric for regression analysis, here is number setup:
cut(worst to best): Fair 1, Good 2, Ideal 3, Premium 4, Very Good 5
color(J worst to D best): J 1, I 2, H 3, G 4, F 5, E 6, D 7
clarity(I1 worst, SI2, SI1, VS2, VS1, VVS2, VVS1, IF best):

```
tmp <- dat %>%
  mutate(
    cut_num = case_when (
      cut=="Fair" ~ 1,
      cut=="Good" ~ 2,
      cut=="Ideal" ~3,
      cut=="Premium" ~ 4,
      cut=="Very Good" ~ 5
    ), color_num = case_when (
      color=="J" ~ 1,
      color=="I" ~ 2,
      color=="H" ~3,
      color=="G" ~ 4,
      color=="F" ~ 5,
      color=="E" ~ 6,
      color=="D" ~ 7
    ), clarity_num = case_when (
      clarity=="I1" ~ 1,
      clarity=="SI2" ~ 2,
      clarity=="SI1" ~3,
      clarity=="VS2" ~ 4,
      clarity=="VS1" ~ 5,
      clarity=="VVS2" ~ 6,
      clarity=="VVS1" ~ 7,
      clarity=="IF" ~ 8
    )
  )
my_dat<- tmp %>%
select(carat,cut_num,color_num,clarity_num,depth,table,price,x,y,z)
```

now we have a dataset all variables numeric

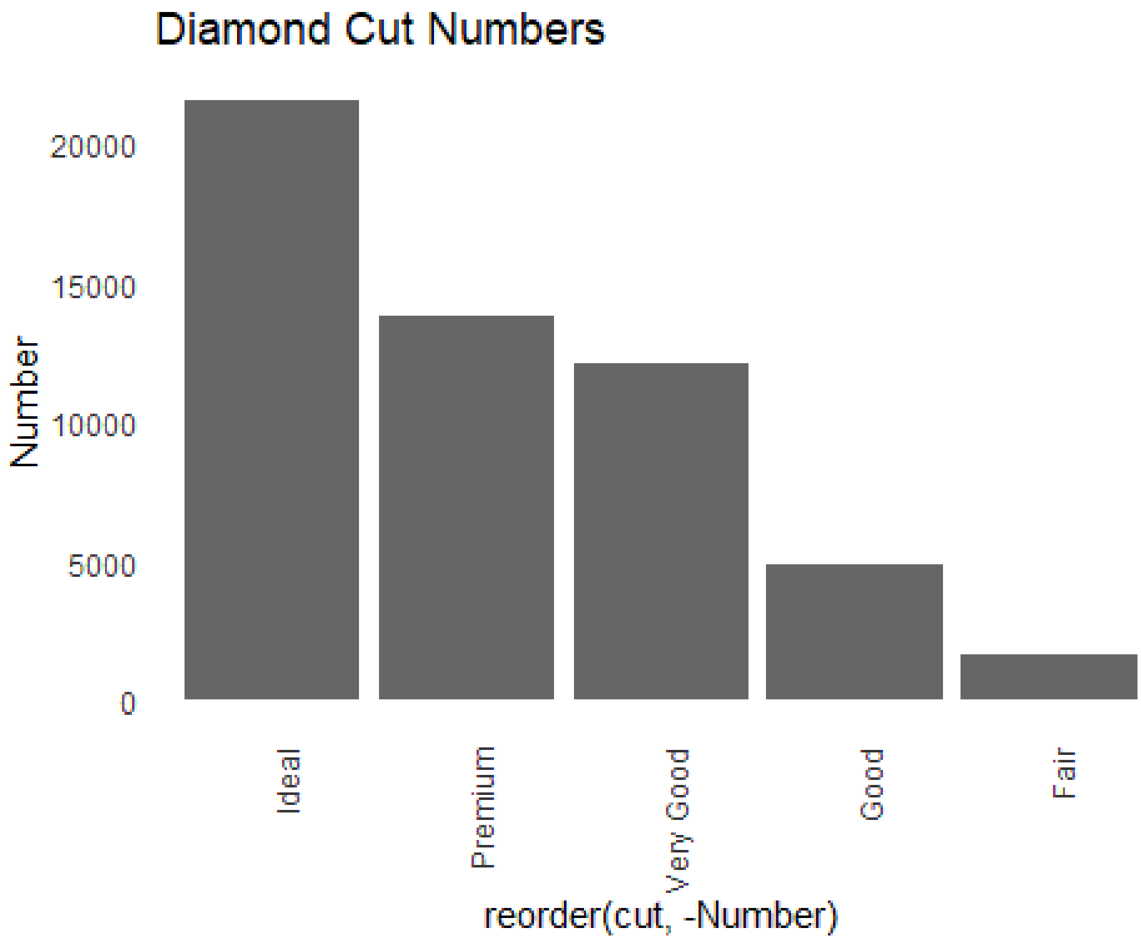
```
glimpse(my_dat)
## Rows: 53,918
## Columns: 10
## $ carat       <dbl> 0.23, 0.21, 0.23, 0.29, 0.31, 0.24, 0.24, 0.26,
0.22, 0.23~
## $ cut_num     <dbl> 3, 4, 2, 4, 2, 5, 5, 5, 1, 5, 2, 3, 4, 3, 4, 4, 3,
2, 2, 5~
## $ color_num   <dbl> 6, 6, 6, 2, 1, 1, 2, 3, 6, 3, 1, 1, 5, 1, 6, 6, 2,
1, 1, 1~
## $ clarity_num <dbl> 2, 3, 5, 4, 2, 6, 7, 3, 4, 5, 3, 5, 3, 2, 2, 1, 2,
3, 3, 3~
## $ depth       <dbl> 61.5, 59.8, 56.9, 62.4, 63.3, 62.8, 62.3, 61.9,
65.1, 59.4~
```

```
## $ table       <dbl> 55, 61, 65, 58, 58, 57, 57, 55, 61, 61, 55, 56, 61,
54, 62~
## $ price       <int> 326, 326, 327, 334, 335, 336, 336, 337, 337, 338,
339, 340~
## $ x           <dbl> 3.95, 3.89, 4.05, 4.20, 4.34, 3.94, 3.95, 4.07,
3.87, 4.00~
## $ y           <dbl> 3.98, 3.84, 4.07, 4.23, 4.35, 3.96, 3.98, 4.11,
3.78, 4.05~
## $ z           <dbl> 2.43, 2.31, 2.31, 2.63, 2.75, 2.48, 2.47, 2.53,
2.49, 2.39~
```

4. Data Visualization

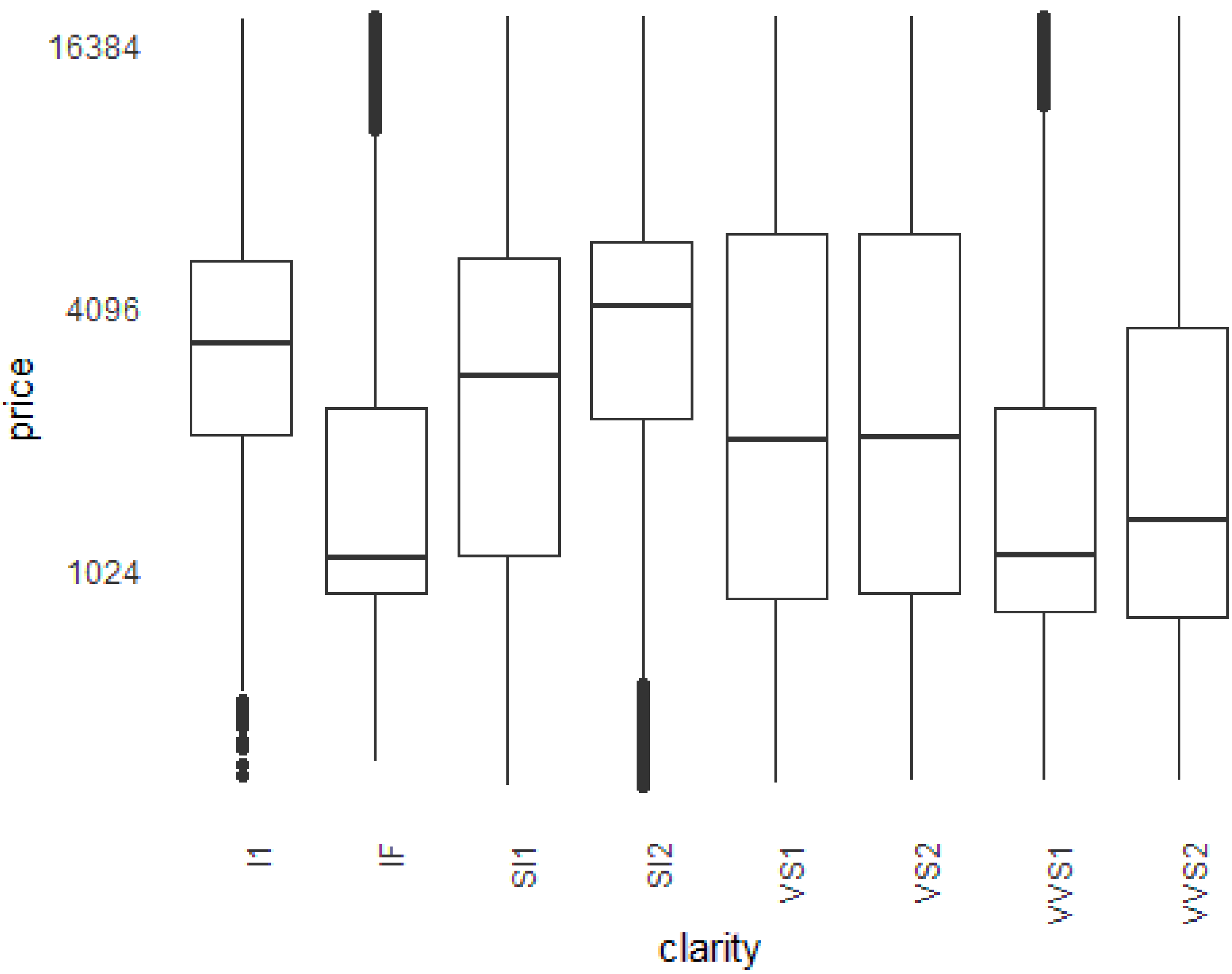
4.1 Plot Diamond Cut by Numbers to see cut distribution

We see 3 major cuttings: Ideal, Premium, Very Good

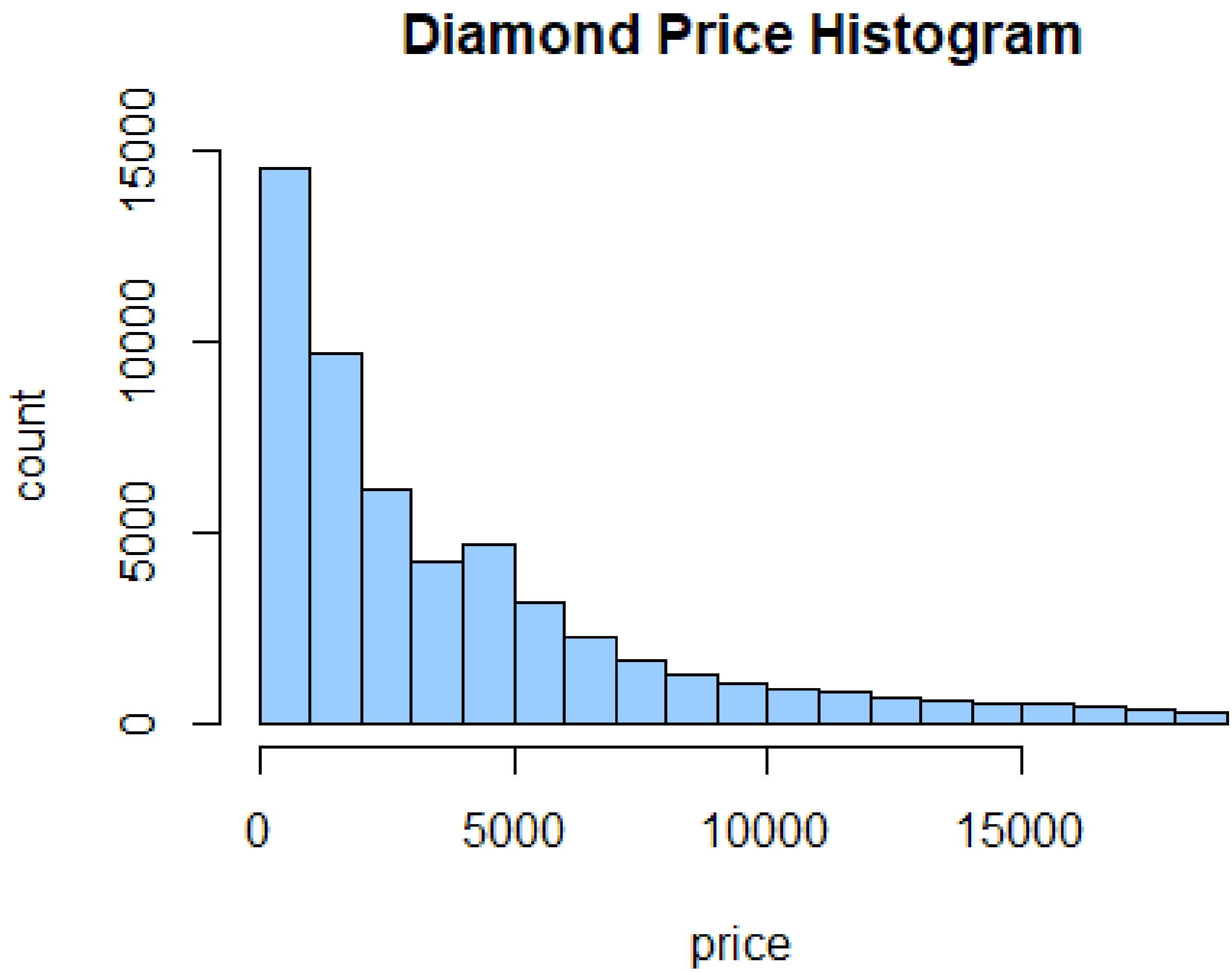


4.2 Question: is the Best Clarity sold at highest price?

From plot below it shows the highest median price is SI2, the average clarity, with many outliers in lower range. Best clarity (IF, VVS1) median price is not high, but more outliers in upper price range, suggest they are more sold expensively probably due to good quality. The Worst 2 clarity (SI2, I1) has more outliers in lower range, suggest their poor quality could drag down price also.



4.3 Diamond price distribution, the majority is less than 10K -15K



4.4 Let's look into Diamonds price for those less than $950 vs higher than $5323, we find low-end price stay about $600 - 800, high-end is more cantered around 5500-10K

4.5 Very expensive diamonds cut percent

1 Ideal 33.98
2 Premium 32.36
3 Very Good 22.98
4 Good 7.77
5 Fair 2.91

Very expensive diamonds color percent

1 H 23.30
2 G 22.33
3 F 18.12
4 I 15.86
5 E 8.41
6 J 6.15
7 D 5.83

It's an interesting observation that best color D only 5.8%, suggest color is not a main factor to attract more buyers compared with clarity or others.

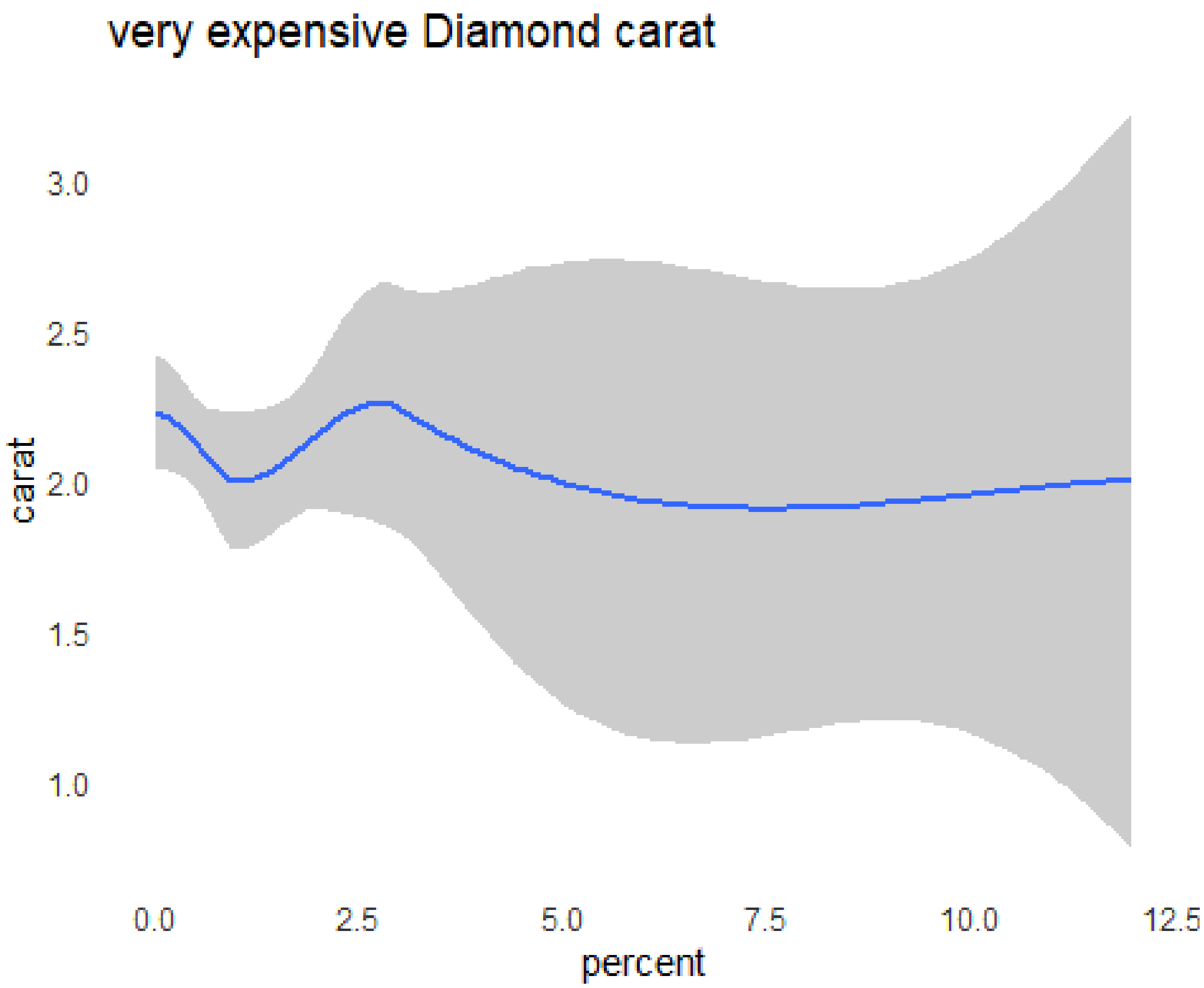
Very expensive diamonds carat distribution: a lot of one percent

```
# 45 Diamonds in one percent group
```r
total<-length(!is.na(t$carat))

t%>%group_by(carat)%>%summarise(percent=round(n()/total * 100))%>%
filter(percent>=1) %>% nrow()

## [1] 45
```

4.6 Plot the distribution of carat, min - max from 1.04 to 4.5, mid-range around 2


```

4.7 Some small diamonds are sold more than $2075. Question: what's their carat? What's their cutting quality, clarity?

```
##   price carat       cut color clarity
## 1  2160  0.34      Fair     F      I1
## 2  2287  0.34     Ideal     D      IF
## 3  2287  0.34     Ideal     D      IF
## 4  2346  0.34     Ideal     D      IF
## 5  2346  0.34     Ideal     D      IF
## 6  2366  0.30 Very Good     D    VVS2
```

Above data shows their carat is around 0.34, cut is normal, but many have Best Color (D) and Best clarity (IF). It suggests buyers love these diamond's crystal quality even if they are small, and buyers may not be able to tell the difference between cut grades due to their lack of professional knowledge in cutting techniques.

4.8 I add a new column VolGrp to get full size, after calculation the data shows cutting point for size small is < 65.19, large is >170.84, those in-between is Medium.

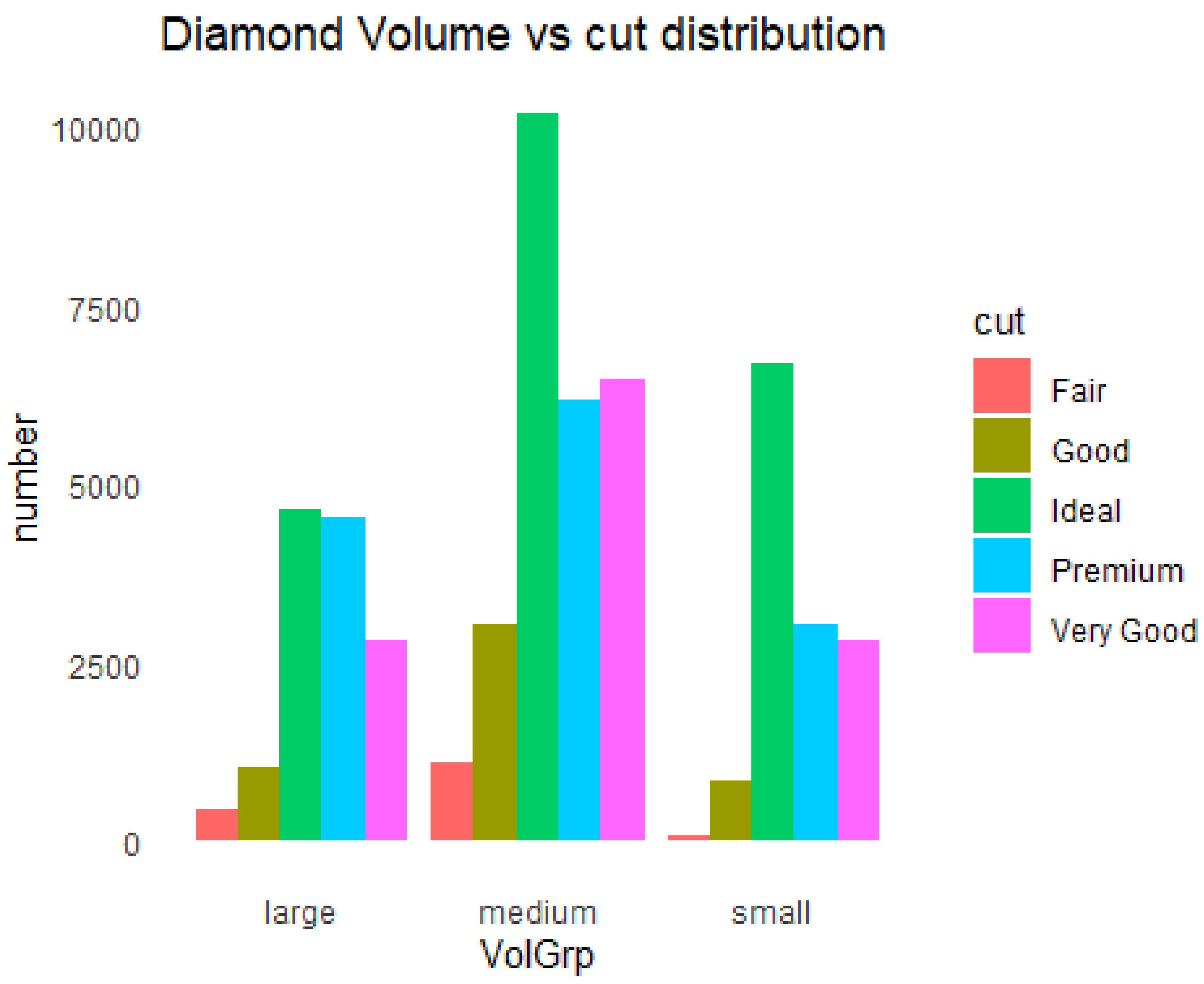
```
tmp<-dat %>% select(price,x,y,z,carat,cut,color,clarity) %>%
  mutate(
  Vol = round(x*y*z, digits = 2),
  VolGrp = case_when(
    Vol <65.19 ~ "small",
    Vol >170.84 ~ "large",
    TRUE ~ "medium"
  )
)
```

```
# The over-lap of price in Large, Medium, Small group
tmp %>% group_by(VolGrp) %>%
  summarise(min_price=min(price), max_price=max(price))
```

```
## # A tibble: 3 x 3
##   VolGrp min_price max_price
##   <chr>      <int>     <int>
## 1 large       1970     18823
## 2 medium       452     17590
## 3 small        326      2366
```

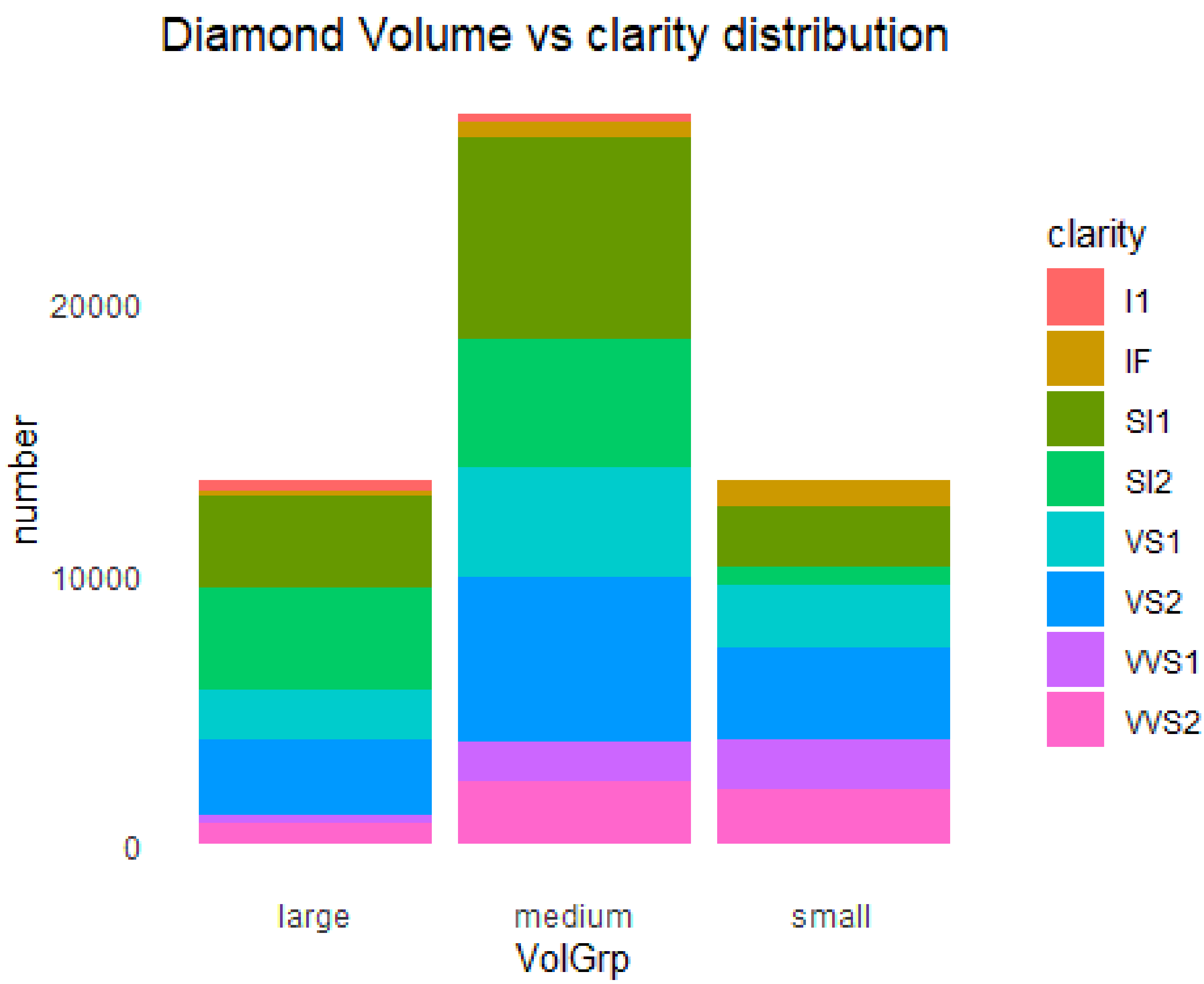
Since we have Diamond VolGrp data, we like to see cut and clarity by group. The following plot shows the 3 types of cuttings Very Good, Premium, Ideal occurs in all groups, however, the Worst type (Fair) percentage at small group is relative much lower than other 2 groups.

```
## `summarise()` has grouped output by 'VolGrp'. You can override using
the `.groups` argument.
```



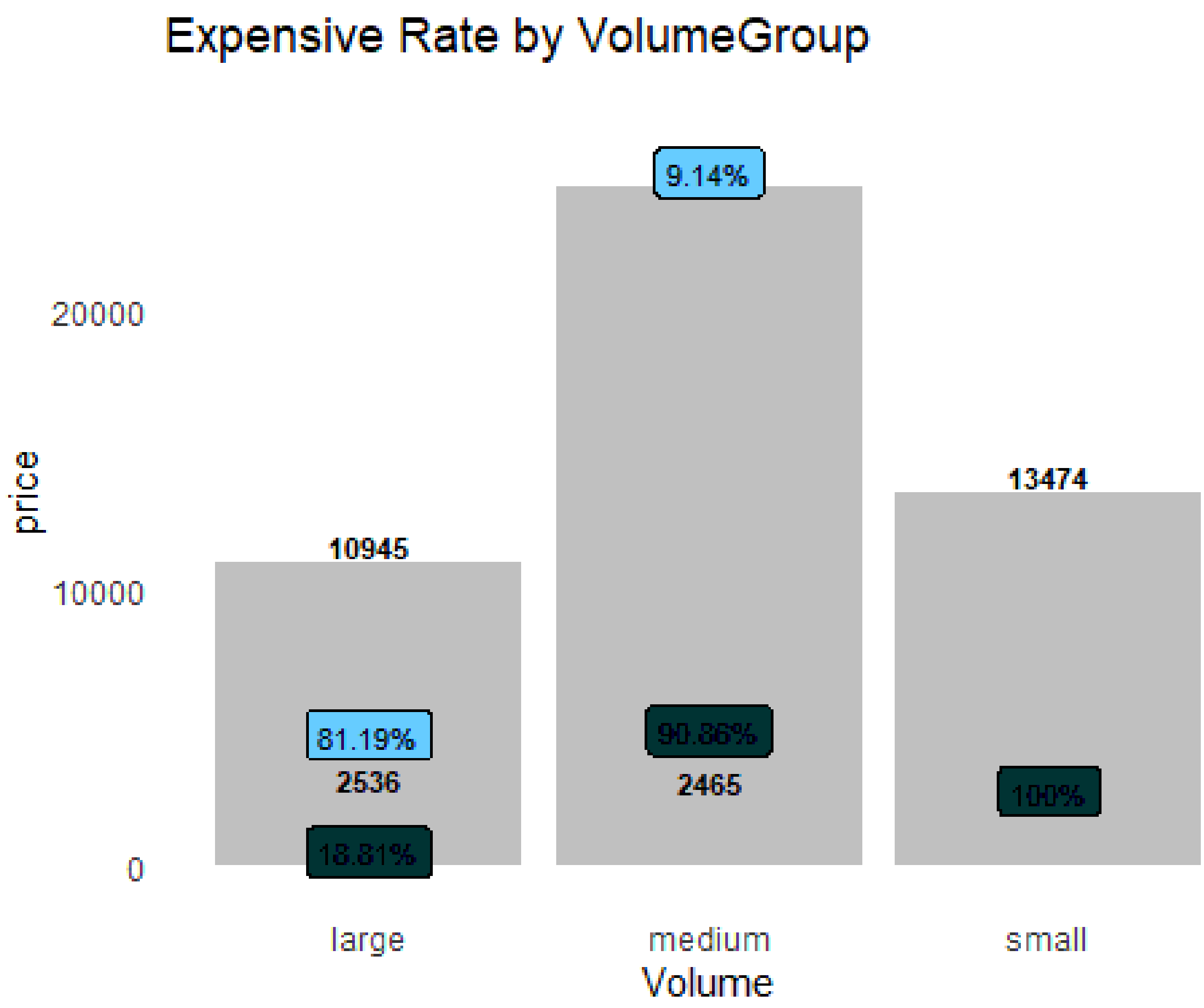
The following plot shows all groups have all clarity, with large group percentage for Best clarity (VVS1, IF) is lower than other 2 groups.

```
## `summarise()` has grouped output by 'VolGrp'. You can override using
the `.groups` argument.
```



4.9 Now we like to see Diamonds price expensiveness vs VolGrp, I add another new column Expensive, price > $5350 expensive is 1 (true). The observation tells us all small VolGrp diamonds are not expensive, zero percentage (none sold over $5350); medium size Diamonds only has around 9% expensiveness; large Diamonds is over 80% expensive. The size of a diamond does drive price higher.

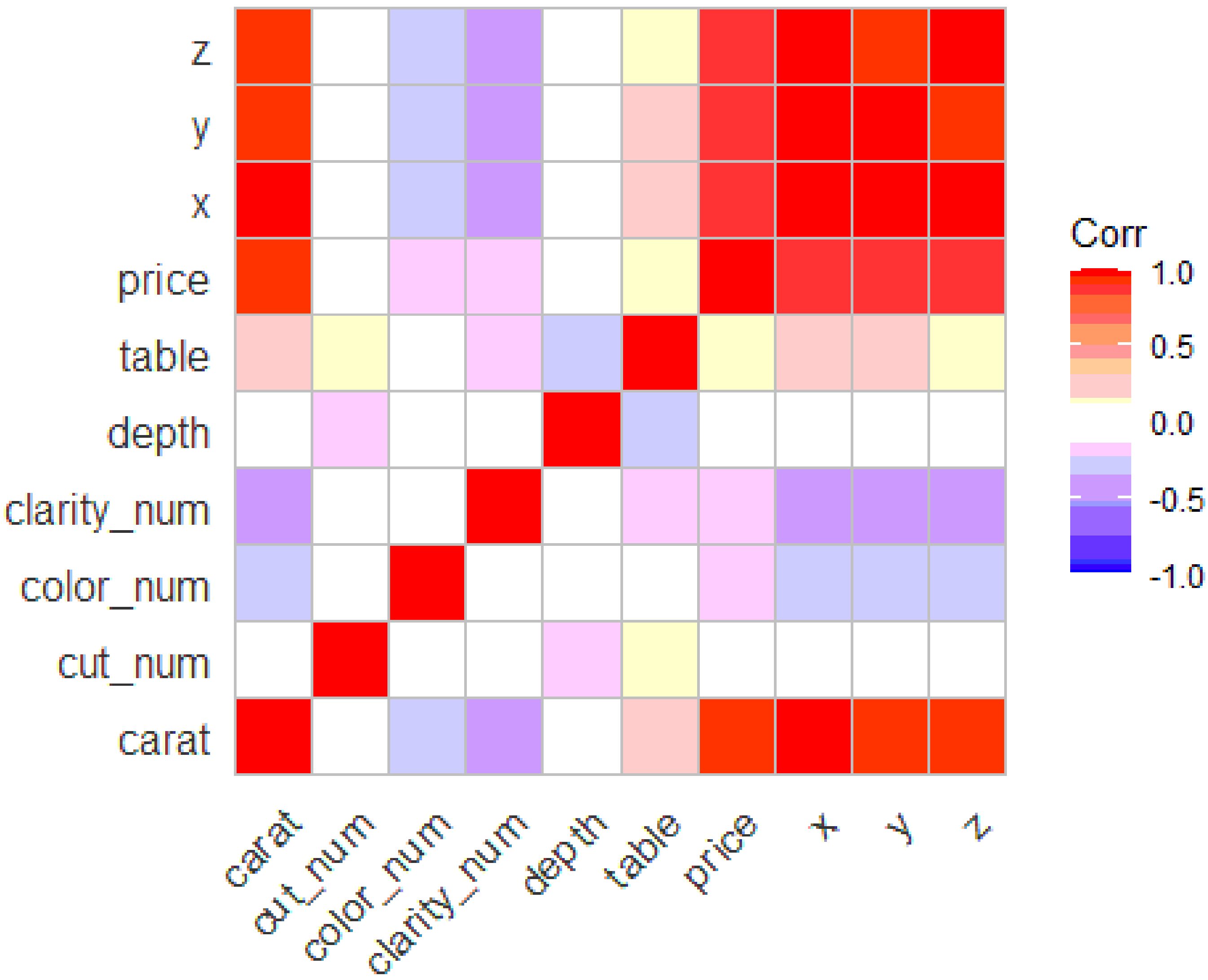
```
tmp4<-tmp %>% select(VolGrp,price) %>% mutate(
  Expensive=case_when(
    price >=5350 ~ 1,
    price <5350 ~ 0
  ))
```


Expensive Rate by VolumeGroup
9.14%
20000
price
13474
10945
10000
81.19%
90.86%
2536
2465
100%
18.81%
0
large
medium
small
Volume

5. Data Analysis

5.1. Correlation

First check correlations of all numeric variables, find price, carat, x,y,z are strong correlated. Cut, color, clarity does not impact price strongly. There are some correlation between table & depth, cut & color.



4 variables have score more than 0.5 to price, shows they are strong correlated

```
my_dat2 <- my_dat %>% select(carat,x,y,z,price)

cor(my_dat2$price, my_dat2$carat)
```

```
## [1] 0.922
```

```
cor(my_dat2$price, my_dat2$x)
## [1] 0.887
cor(my_dat2$price, my_dat2$y)
## [1] 0.868
cor(my_dat2$price, my_dat2$z)
## [1] 0.868
```

“x”, “y”, “z” show a higher correlation to price column. “depth”, “cut” and “table” show low correlation. We could consider dropping in regression analysis

5.2 Test the Models

1. Split into train 0.75 and test set 0.25 of dataset

```
set.seed(123)
test_index <- createDataPartition(my_dat2$price, times = 1, p = 0.25, list
= FALSE)
test <- my_dat2[test_index,]
train <- my_dat2 [-test_index,]

# The RMSE function that will be used
RMSE <- function(true_ratings = NULL, predicted_ratings = NULL) {
  round(sqrt(mean((true_ratings - predicted_ratings)^2)), digits = 4)
}

mu<- mean(train$price) #3935.8

# Base-line RMSE, 1.127
rmse_output <- RMSE(log(test$price), log(mu))
RMSE_table <- tibble(Method = "Base-line RMSE",
                      RMSE = rmse_output)
RMSE_table
```

```
## # A tibble: 1 x 2
##   Method          RMSE
##   <chr>          <dbl>
## 1 Base-line RMSE  1.12
```

2. Test models

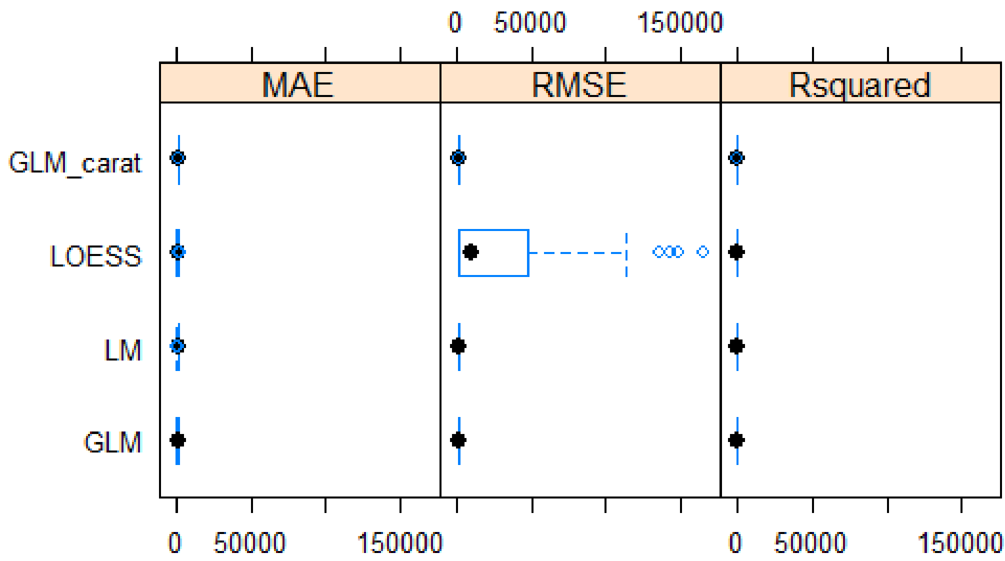
```
set.seed(123)
lm_model <- train(price ~ x + y + z + carat, data=train, method="lm")
glm_model <- train(price ~ x + y + z + carat, data=train, method="glm")
glm_carat_model <- train(price ~ carat, data=train, method="glm")
loess_model<- train(price ~ x + y + z + carat, data=train,
method="gamLoess")
```

glm gives the best in Rsquare, RMSE, MAE

```
results <- resamples(list(LM=lm_model, GLM=glm_model,
GLM_carat=glm_carat_model, LOESS=loess_model))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: LM, GLM, GLM_carat, LOESS
## Number of resamples: 25
##
## MAE
##            Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
## LM          876     880    887  894     906  917    0
## GLM         874     881    888  893     904  925    0
## GLM_carat   988    1001   1004 1004    1007 1016    0
## LOESS       819     834    839 1082    1494 1839    0
##
## RMSE
##            Min. 1st Qu. Median  Mean 3rd Qu.   Max. NA's
## LM         1488    1521   1538  1703    2038   2212    0
## GLM        1496    1519   1537  1715    2017   2139    0
## GLM_carat  1528    1542   1552  1553    1563   1585    0
## LOESS      1375    1409   1422 31425   80590 123529    0
##
## Rsquared
##               Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
## LM         0.71854 0.75518  0.851 0.819   0.856 0.861    0
## GLM        0.73526 0.75818  0.852 0.817   0.856 0.861    0
## GLM_carat  0.84196 0.84767  0.849 0.849   0.850 0.851    0
## LOESS      0.00204 0.00361  0.873 0.596   0.875 0.880    0
```

```
bwplot(results)
```



3. Pick GLM model

```r
set.seed(123)
# glm with carat only
fit_glm_carat <- glm(price ~ carat, data=train)
pred<-predict(fit_glm_carat, test)
pred<-abs(pred)
rmse_output <- RMSE(log(test$price), log(pred))

RMSE_table <- rbind(RMSE_table,
                    tibble(Method = "Regression model by carat",
                            RMSE = rmse_output))

# glm regression model with 4 predictors x, y, z, carat
set.seed(123)
fit_glm_4 <- glm(price ~ x + y + z + carat, data=train)
pred<-predict(fit_glm_4, test)
pred <- abs(pred)
rmse_output <- RMSE(log(test$price), log(pred))

RMSE_table <- rbind(RMSE_table,
                    tibble(Method = "Regression model by 4 predictors",
                            RMSE = rmse_output))


# coefficient

fit_glm_carat$coefficients
```

```
## (Intercept)       carat
##       -2245        7737
fit_glm_4$coefficients
```

```
## (Intercept)           x           y           z       carat
##        2698        -161         276       -2243       10668
```

```
RMSE_table
```

```
## # A tibble: 3 x 2
##   Method                          RMSE
##   <chr>                          <dbl>
## 1 Base-line RMSE                 1.12
## 2 Regression model by carat      0.790
## 3 Regression model by 4 predictors 0.361
```

Next we like to tune for data partitioning percentage

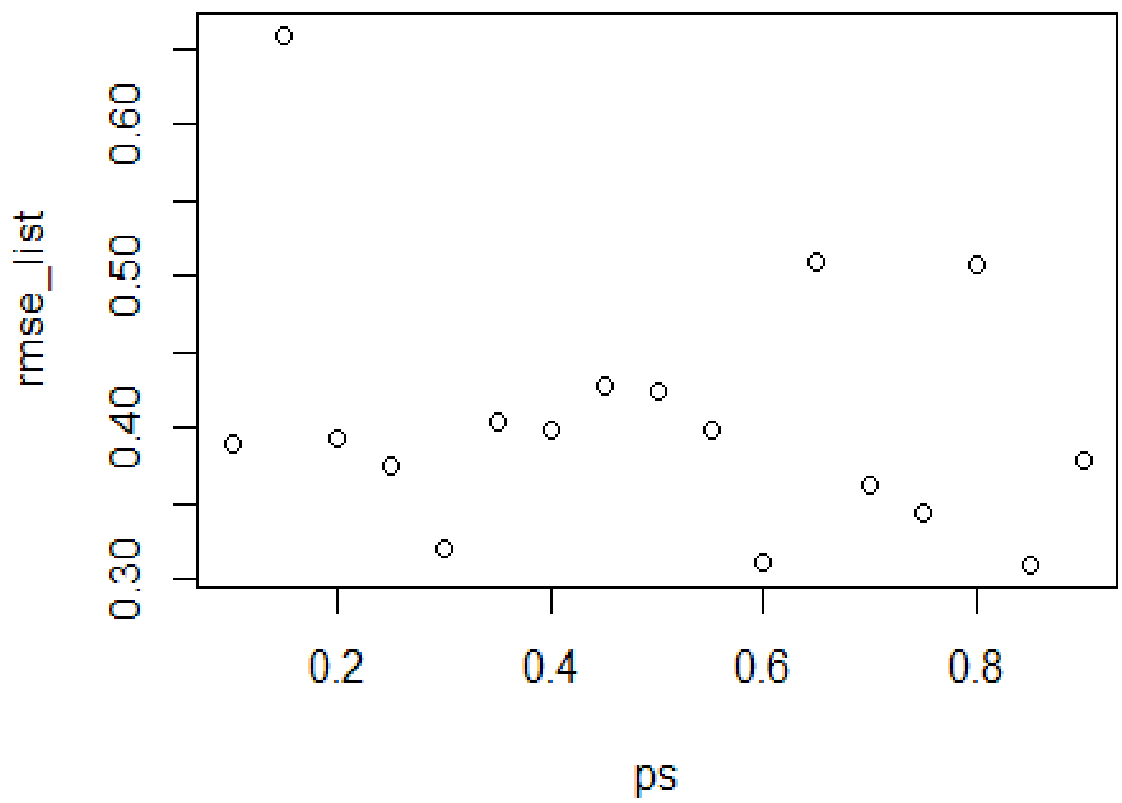
```
ps <- seq(from=.10, to=.90, by=.05)

rmse_list <- sapply(ps, function(p){
  train_index <- createDataPartition(my_dat2$price,
                                     times=1,
                                     p=ps,
                                     list=FALSE)
  train <- my_dat2[train_index,]
  test <- my_dat2[-train_index,]
  fit <- glm(price ~ x+y+z+carat, data = train)
  test <- test %>%
    mutate(pred_price = abs(predict.glm(fit, newdata=test)))
  RMSE(log(test$price), log(test$pred_price))
})

# Pick suggested percentage 60%
min(rmse_list)
```

```
## [1] 0.309
```

```
ps[which.min(rmse_list)] #0.6
```

```
## [1] 0.6
```

```
plot(ps, rmse_list) # no clear winner
```



Go back using p in our data split

```
set.seed(123)
train_index <- createDataPartition(my_dat2$price, times = 1, p = 0.6, list
= FALSE)
train <- my_dat2 [train_index,]
test <- my_dat2[-train_index,]


fit_glm_carat_final <- glm(price ~ carat, data=train)
fit_glm_4_final <- glm(price ~ x + y + z + carat, data=train)
pred_3<-predict(fit_glm_carat_final, test)
pred_3<-abs(pred_3)
pred_4<-predict(fit_glm_4_final, test)
pred_4<-abs(pred_4)

rmse_output_3 <- RMSE(log(test$price), log(pred_3))
rmse_output_4 <- RMSE(log(test$price), log(pred_4))


RMSE_table <- rbind(RMSE_table,
                    tibble(Method = "Regression model using carat + better
```

```
partition",
                           RMSE = rmse_output_3),
                    tibble(Method = "Regression model using 4 predictors +
better partition",
                           RMSE = rmse_output_4))

RMSE_table
```

```
## # A tibble: 5 x 2
##   Method                                              RMSE
##   <chr>                                              <dbl>
## 1 Base-line RMSE                                     1.12
## 2 Regression model by carat                          0.790
## 3 Regression model by 4 predictors                   0.361
## 4 Regression model using carat + better partition    0.758
## 5 Regression model using 4 predictors + better partition 0.352
```

```
# final model formula
# Y_hat = intercept + b1*x +b2*y +b3*z
fit_glm_4_final$coefficients
```

```
## (Intercept)           x           y           z       carat
##        2825       -1090         108        -532       10791
```

6. Conclusion

Diamond price dataset is a good exercise for beginners like me. Before I took this course with Harvard, I never used R and have zero knowledge in ML! During the course and by implementing this project, I learnt a lot: not only in R but also in RMarkdown, RStudio, ML algorithm, and how to understand the data.

My laptop is 4G Windows 10, I tried USB 'Readyboost' feature as instructor said in discussion, but not successful; algorithm using a lot of MEM(e.g: knn, randomForst) runs very slow, session dead after run overnight and crashed my RStudio, so I didn't use them in final report. Techniques in R such as how to clean mem, hide warning message, and plot tips such as arrange two graphs at one page, adjust figure width are also good exercise for beginners like me.

This report is limited in the amount of data in training and test. For the dataset insight, its many numeric variables good for regression model, and it teaches me the knowledge about diamond’s price; it’s also interesting to drilldown special category of small vs large diamonds to see why their price different. The final RMSE is below 1 and reduced as each round of different predictors and split percentage. I believe other algorithms will outperform the GLM if I have a chance to test. . Also the negative pred value in some output could be a hole for further check, it may due to some influential variables not included in this dataset, or a larger dataset will give a more fitted pred curve.

It’s a great learning process for me! I appreciate a lot to edx and Harvard to joint offer this course. It opens a door of opportunity, I’m more prepared and stronger, for the world of data science! Thank you very much again!