HW1

Yaqi Li

3/30/2019

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
library(ggplot2)
library(ggmap)
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  2.0.1       v purrr   0.3.0
## v tidyr   0.8.3       v dplyr   0.8.0.1
## v readr   1.3.1       v stringr 1.3.1
## v tibble  2.0.1       v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(GGally)
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

Q1.1

```
hsb2 = read.csv(file = 'hsb2.csv',header = TRUE, sep = ',')
str(hsb2)
```

```
## 'data.frame':    200 obs. of  11 variables:
##  $ id     : int  70 121 86 141 172 113 50 11 84 48 ...
##  $ female : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ race   : int  4 4 4 4 4 4 3 1 4 3 ...
##  $ ses    : int  1 2 3 3 2 2 2 2 2 2 ...
##  $ schtyp : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ prog   : int  1 3 1 3 2 2 1 2 1 2 ...
##  $ read   : int  57 68 44 63 47 44 50 34 63 57 ...
##  $ write  : int  52 59 33 44 52 52 59 46 57 55 ...
##  $ math   : int  41 53 54 47 57 51 42 45 54 52 ...
##  $ science: int  47 63 58 53 53 63 53 39 58 50 ...
##  $ socst  : int  57 61 31 56 61 61 61 36 51 51 ...
```

```
meanscore = rowMeans(hsb2[,7:11])
hsb2$meanscore = meanscore
```

Q1.2

```
hsb2$meanscore[hsb2$meanscore == 99] = NA
hsb2 = within(hsb2,{
              meancat = NA
              meancat[meanscore <45] = 'Low'
              meancat[meanscore >45 & meanscore <60] = 'Middle'
              meancat[meanscore >60 ] = 'High'

})
```

Q1.3

```
newdata = hsb2[order(hsb2$meanscore,decreasing = TRUE),]
sum(is.na(newdata))
```

```
## [1] 3
```

Q1.4

```
newdata2 = na.omit(newdata)
sum(is.na(newdata2))
```

```
## [1] 0
```

Q1.5

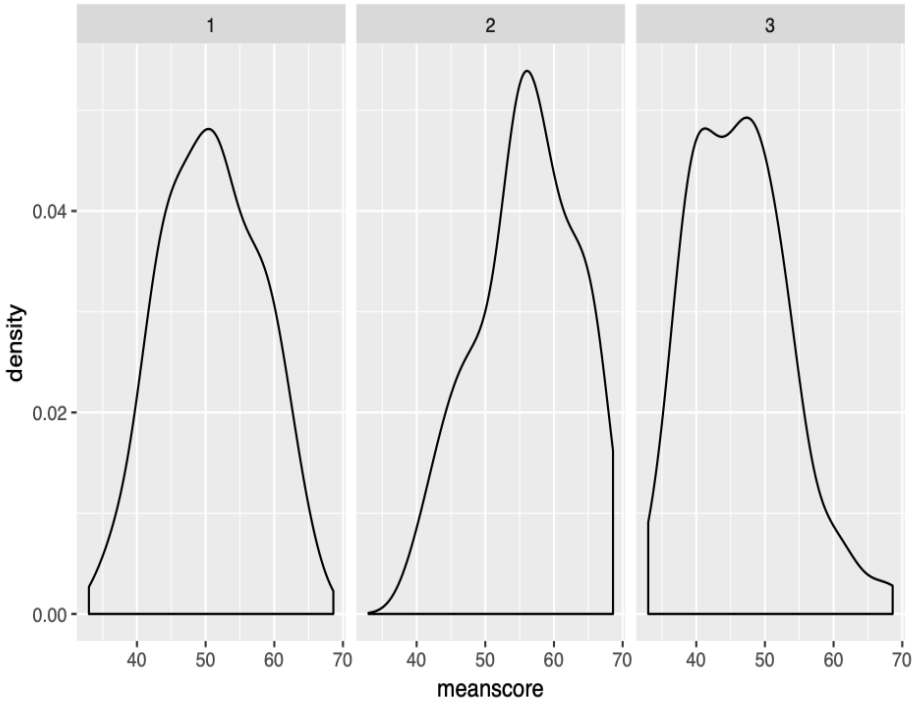
```
index = c('read','write',"math","science","socst","meanscore","meancat")
newdata3 = newdata2[index]
str(newdata3)
```

```
## 'data.frame':    197 obs. of  7 variables:
##  $ read     : int  73 73 63 63 73 71 65 76 71 68 ...
##  $ write    : int  62 67 63 65 60 65 67 63 65 54 ...
```

```
##  $ math     : int  73 71 75 71 71 69 63 60 72 75 ...
##  $ science  : int  69 63 72 69 61 58 66 67 66 66 ...
##  $ socst    : int  66 66 66 71 71 71 71 66 56 66 ...
##  $ meanscore: num  68.6 68 67.8 67.8 67.2 66.8 66.4 66.4 66 65.8 ...
##  $ meancat  : chr  "High" "High" "High" "High" ...
```

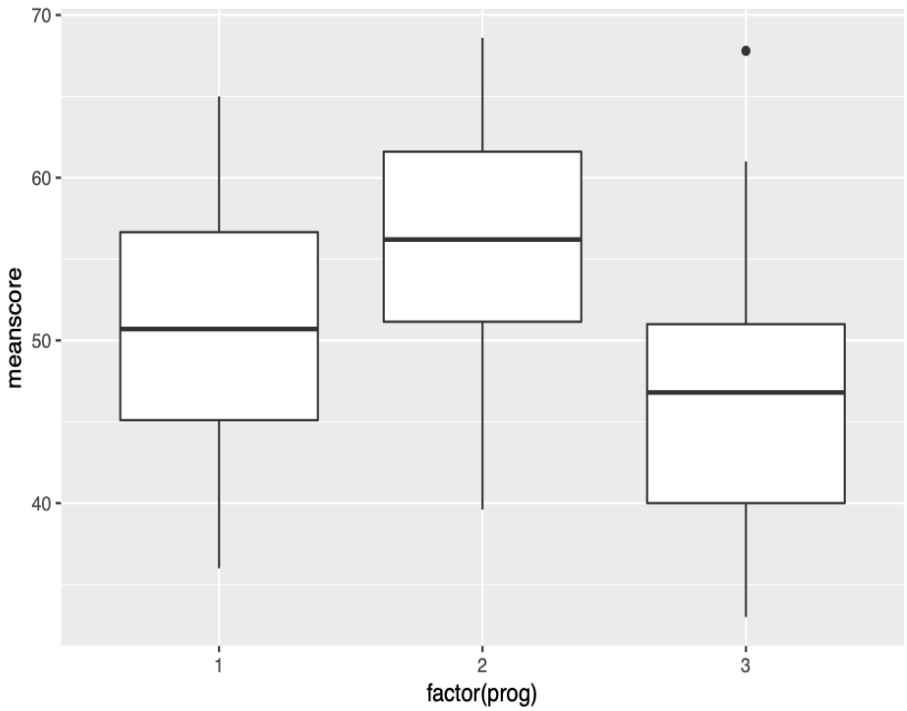
Q2.1

```
den = ggplot(hsb2, aes(x=meanscore)) + geom_density() + facet_grid(~prog) +xlab('meanscore') +ylab('den
den
```



Q2.2

```
newdata2$prog = as.factor(newdata2$prog)
ggplot(newdata2, aes(x=prog, y = meanscore)) + geom_boxplot() +xlab("factor(prog)")
```



##Q2.3

```
ta = table(newdata2$meancat)
ta
```

```
##
##   High    Low Middle
##     38     42    117
```

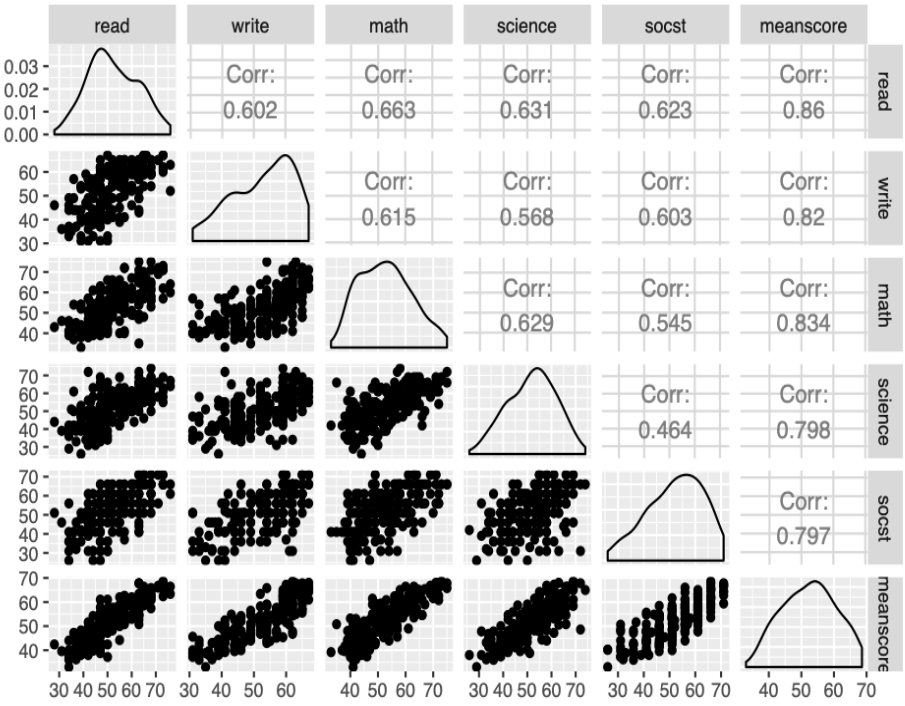
```
sum(ta)
```

```
## [1] 197
```

The total is 197 because three NA are deleted from the dataset.

Q2.4

```
ggpairs(newdata2[,7:12])
```



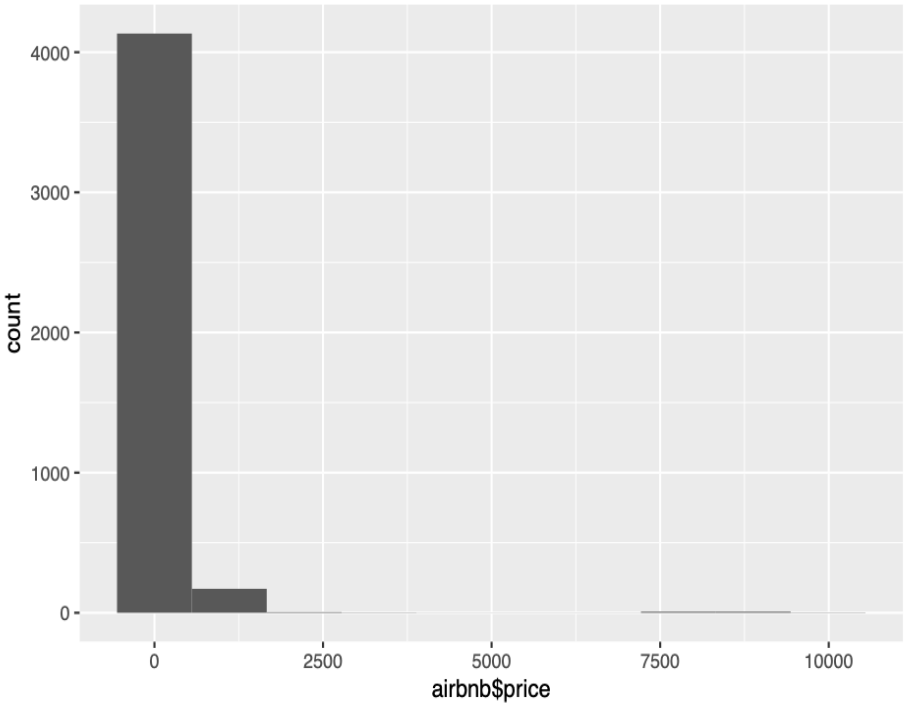
Q3.1

```
airbnb = read.csv(file = 'NOLAlistingsJune2016.csv',header = TRUE, sep = ",")
summary(airbnb$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.0    90.0   132.0   195.5   200.0 10000.0
```

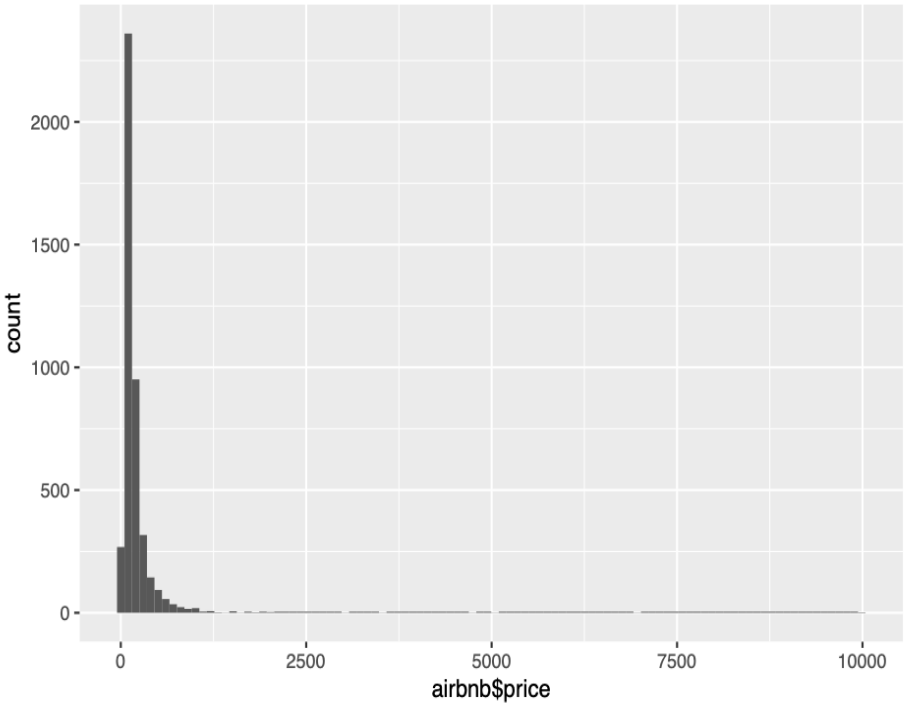
The histogram of the price would be skewd to left very much and has very long right tail. ##Q3.2

```
ggplot(airbnb,aes(airbnb$price)) + geom_histogram(bins = 10)
```



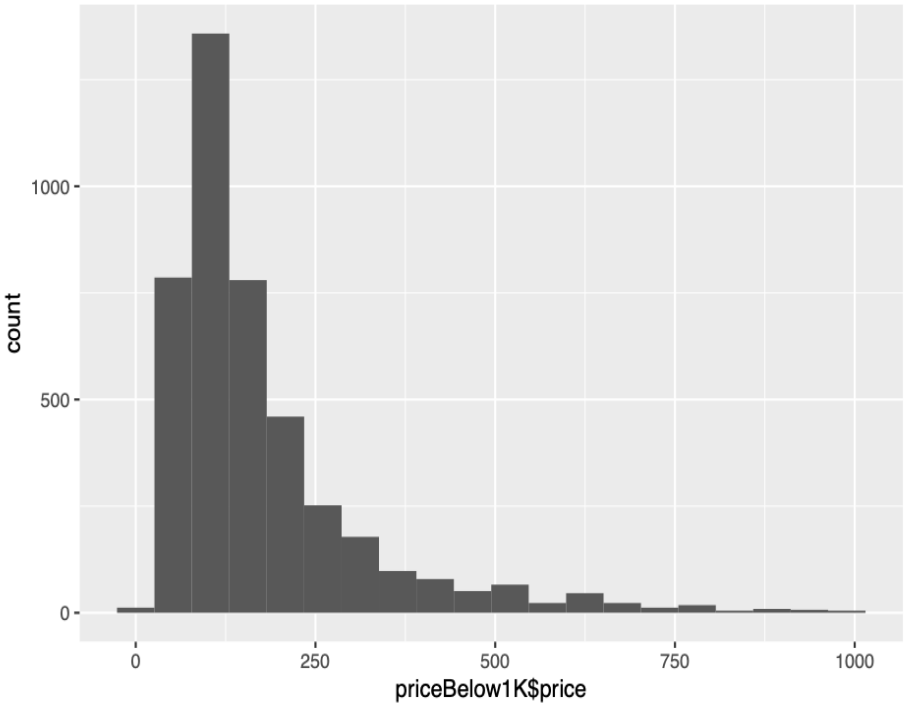
The result of the visualization is not useful because the data is highly left skewed and the granularity of the important part of the data is too coarse. ##Q3.3

```
ggplot(airbnb,aes(airbnb$price)) + geom_histogram(bins = 100)
```



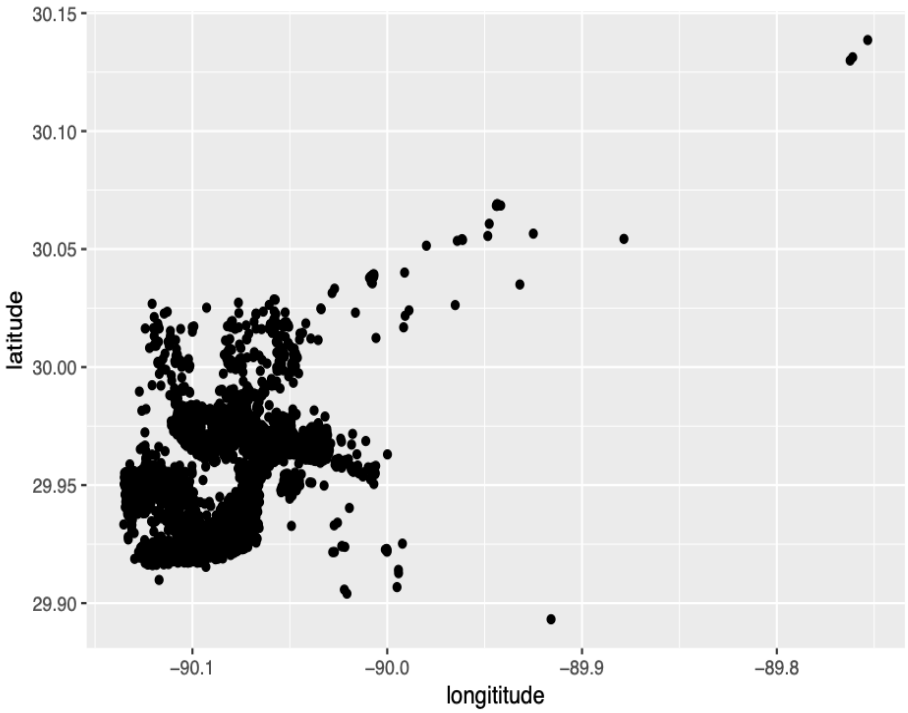
The result is much better than the previous one. ##Q3.4

```
priceBelow1K = subset(airbnb,price <1000)
ggplot(priceBelow1K,aes(priceBelow1K$price)) + geom_histogram(bins = 20)
```



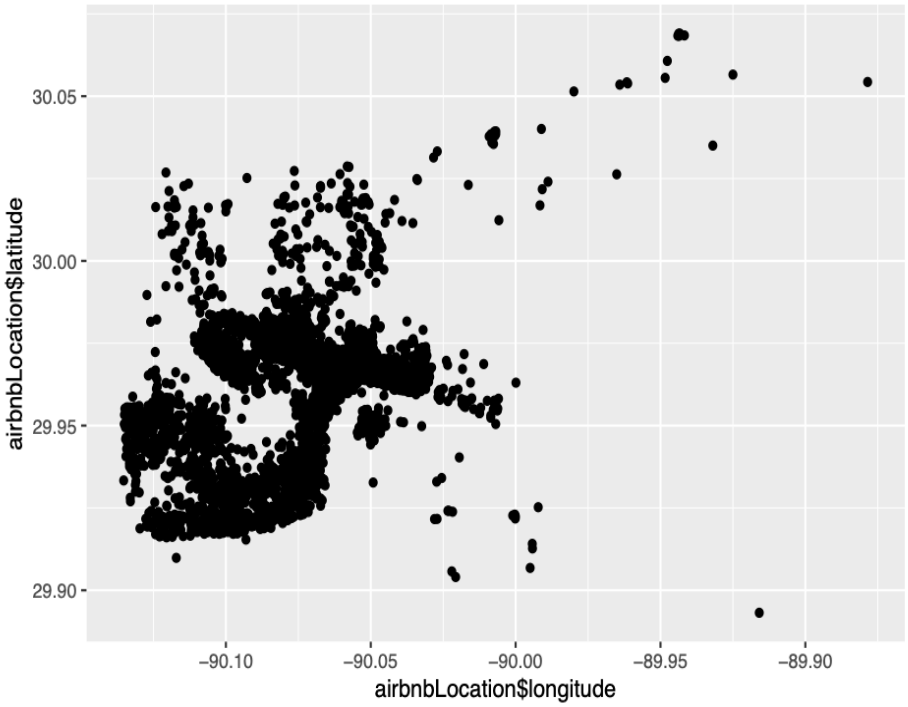
##Q3.5

```
ggplot(airbnb,aes(x=airbnb$longitude,y=airbnb$latitude))+geom_point()+xlab("longititude")+ylab("latitud
```



##3.6 There seems to be several outliers on the right top of the chart

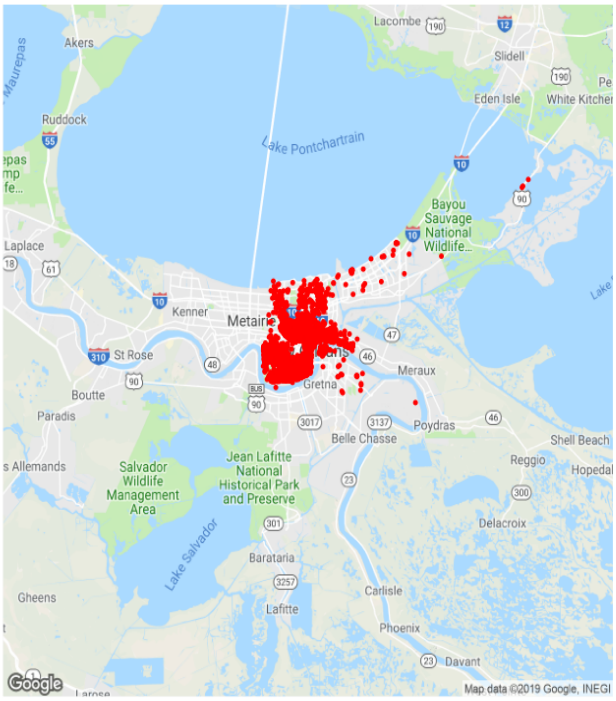
```
airbnbLocation = subset(airbnb,longitude<=-89.8)
ggplot(airbnbLocation,aes(x=airbnbLocation$longitude,y=airbnbLocation$latitude))+geom_point()
```



##Q3.7

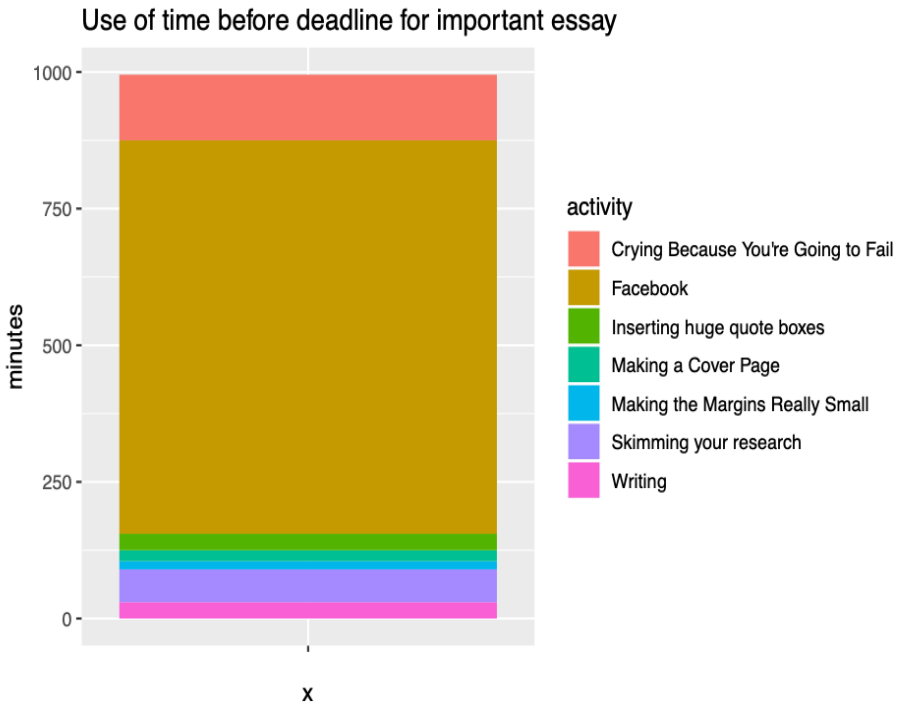
```
NewOrleans= get_map(location = 'new orleans')
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=new%20orleans&zoom=10&size=640x640&sc
## Source : https://maps.googleapis.com/maps/api/geocode/json?address=new+orleans&key=xxx-BhjZ0rF2FxLzC
ggmap(NewOrleans) + geom_point(data = airbnb, aes(x= longitude, y = latitude), size = 0.5, color = "red
```

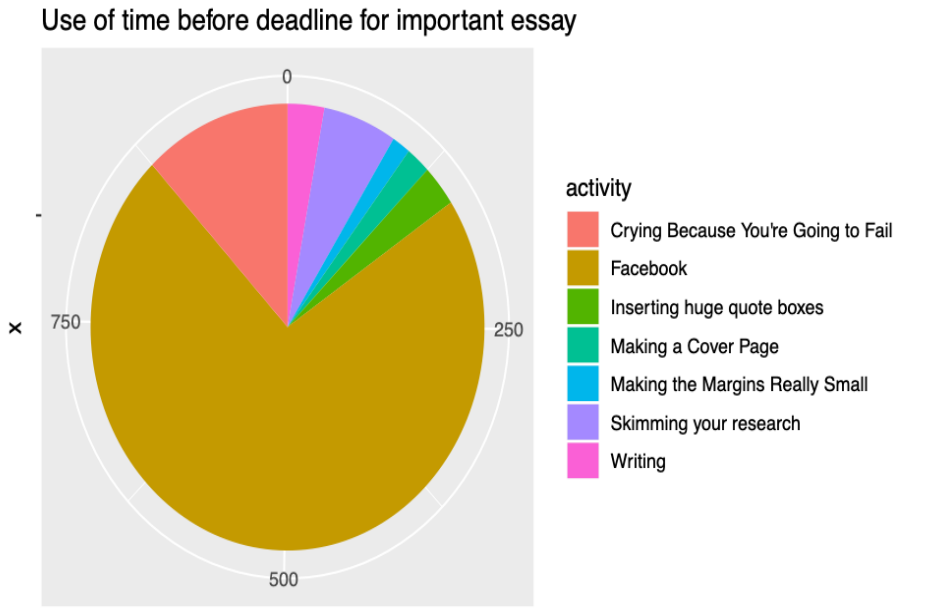


Q4

```
df <- data.frame(activity = c("Writing", "Making the Margins Really Small", "Making a Cover Page", "Ins

bp <- ggplot(df, aes(x="", y=minutes, fill=activity)) + geom_bar(width = 1, stat = "identity") + ggtitl

bp
```



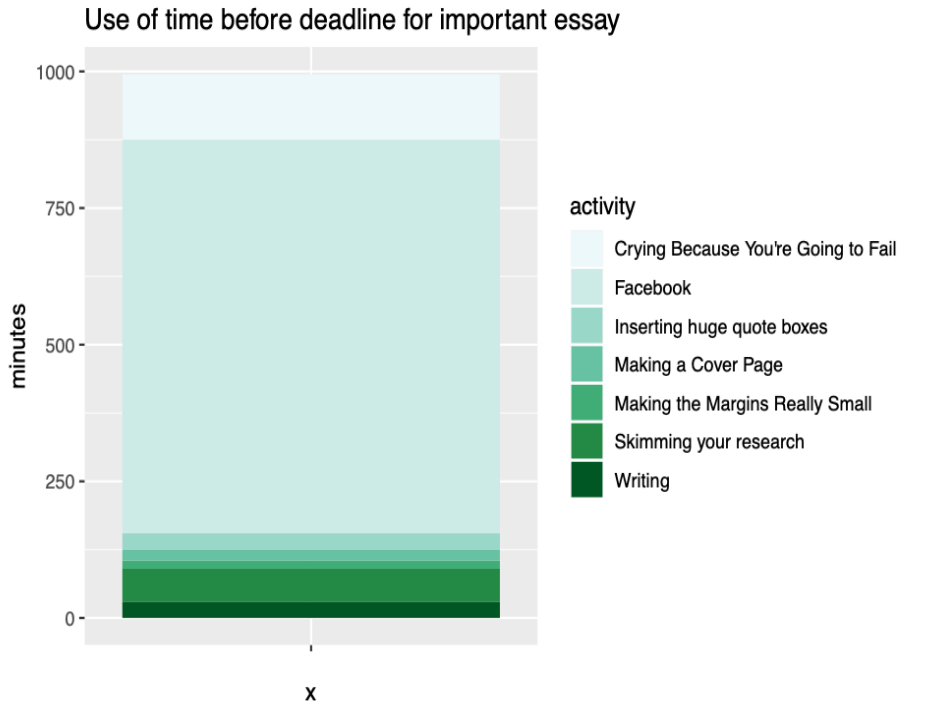

```
pie <- bp + coord_polar("y", start=0)

pie
```



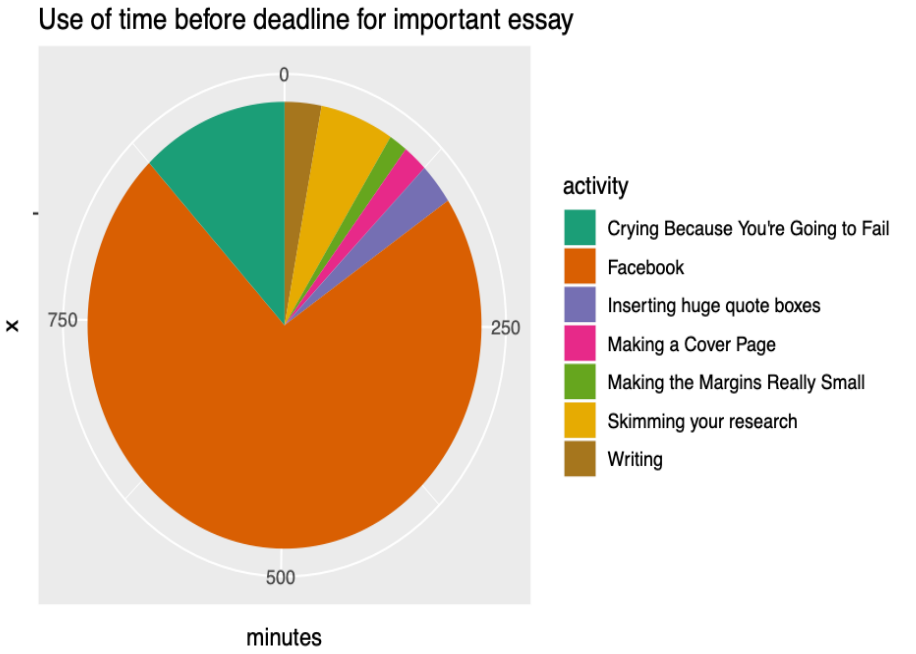
Use

`scale_fill_brewer` to adjust colore scheme of the plot

```
bp +  scale_fill_brewer(palette="BuGn")
```



```
pie + scale_fill_brewer(palette = "Dark2")
```



##Q5 Imagine you work for a company which, for a current project, needs to analyze a shipment of diamonds which was recently delivered. Now imaging you are charged with the task of delivering a brief presentation to your team describing the diamond delivery. How do some variables change with others? What are the best visual ways to express the structure of this data? (Maybe a bar graph, maybe a density plot?) Use the ggplot package to generate a few plots to help you describe the new diamond delivery to your team.

```
data(diamonds)
attach(diamonds)
str(diamonds)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    53940 obs. of  10 variables:
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ 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 ...
```

There are three factor variables in the dataset, which are cut, color, and clarity. Price and carat are continuous variables. x, y, z are number variables and depth variable is deduced from the three. First of all, we inspect whether there are any missing values in the dataset.

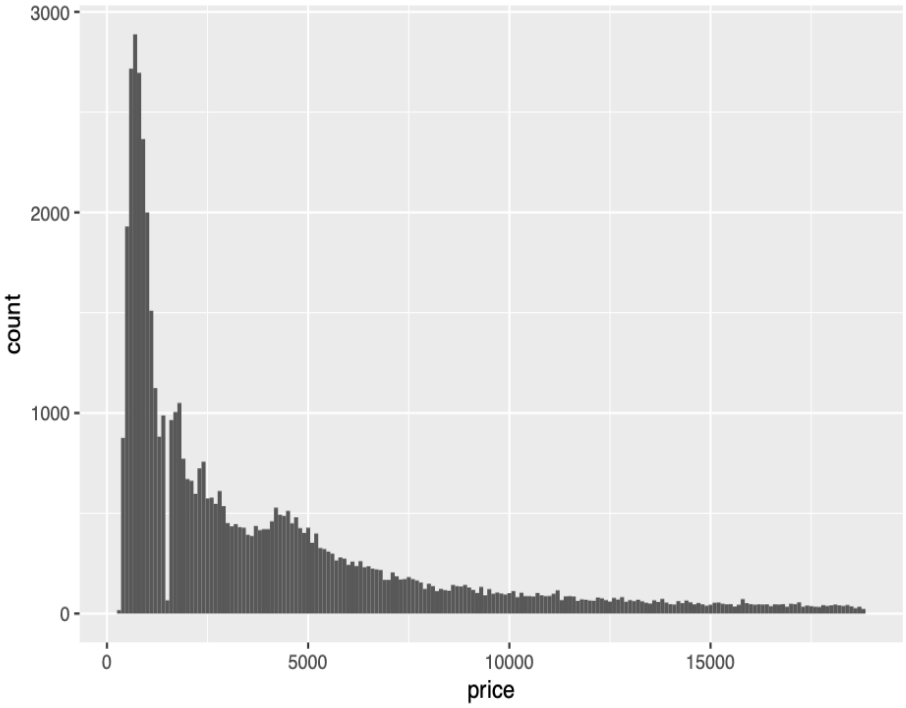
```
sum(is.na(diamonds))
```

```
## [1] 0
```

There are no missing values in the dataset.

Let's first take a look at the distribution of prices of these pricey carbon stones.

```
ggplot(diamonds, aes(x= price))  + geom_histogram(binwidth = 100)
```

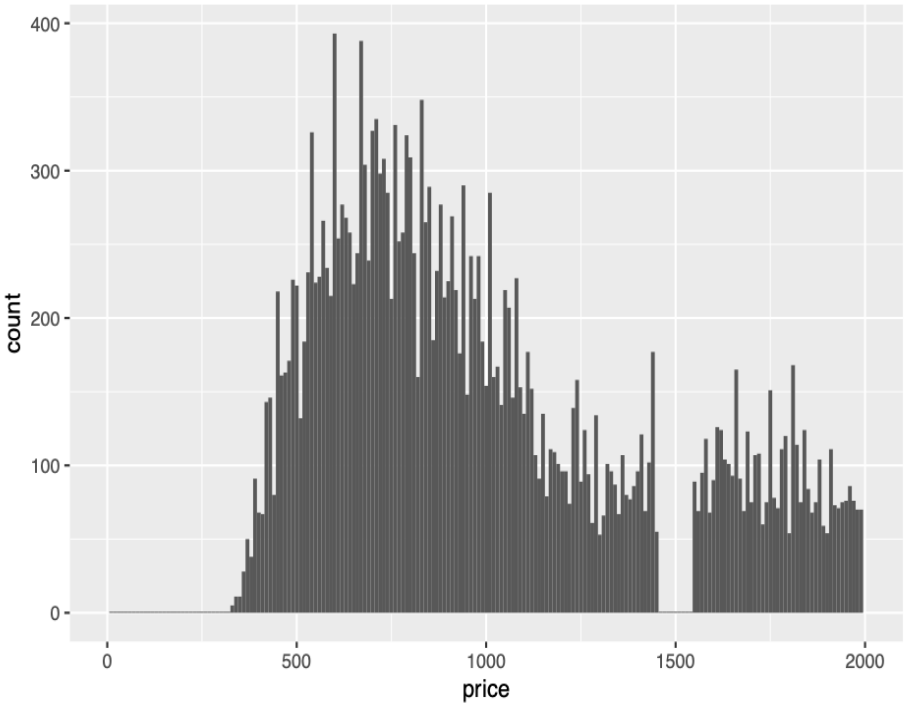


Looks just like the distribution of the wealth of the society! Most people can just afford a small diamond while there are always rich people who can afford high priced diamonds. One thing worth noting is the huge blank strip on the left. Let's take a closer look at that area to find out what's going on.

```
ggplot(diamonds, aes(x= price))  + geom_histogram(binwidth = 10) + xlim(c(1,2000))
```

```
## Warning: Removed 29733 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

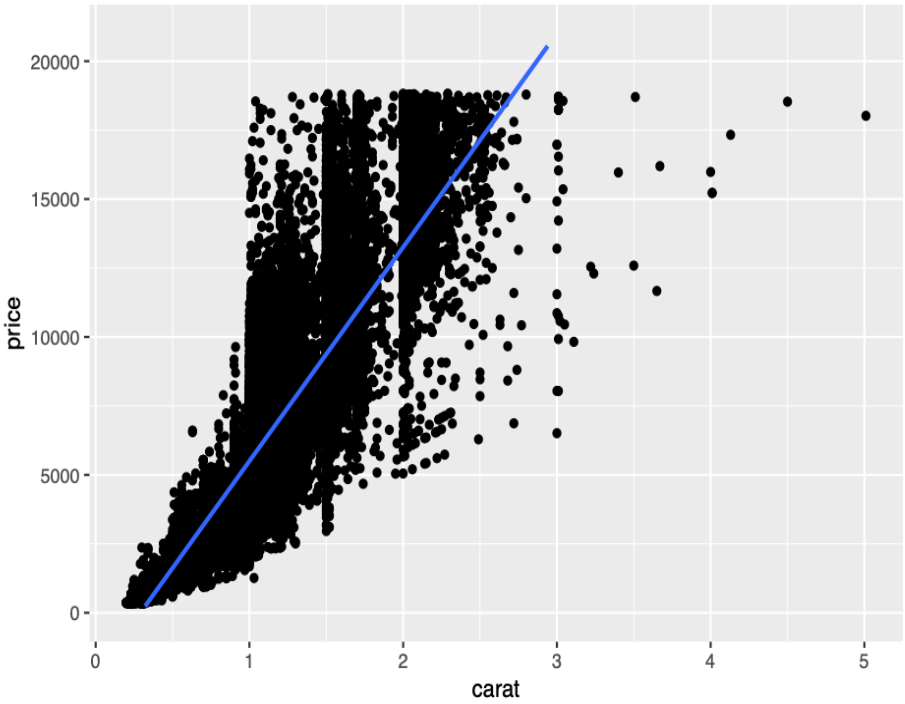


Lots of price data have gone missing around 1500. This looks like someone intentionally deletes the chunk of data around 1500.

Next, we want to check the relationship between carat and price, which should show a positive correlation. Bigger the diamond, more money for it!

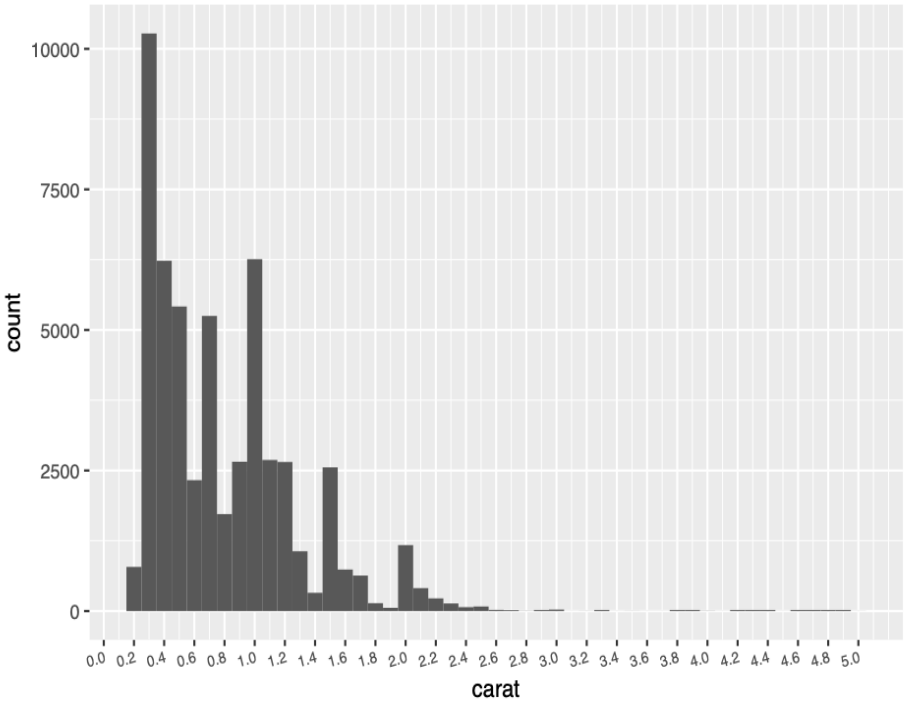
```
ggplot(diamonds,aes(x = carat,y = price)) + geom_point() + geom_smooth(method = "lm") + ylim(0,21000)
```

```
## Warning: Removed 36 rows containing missing values (geom_smooth).
```



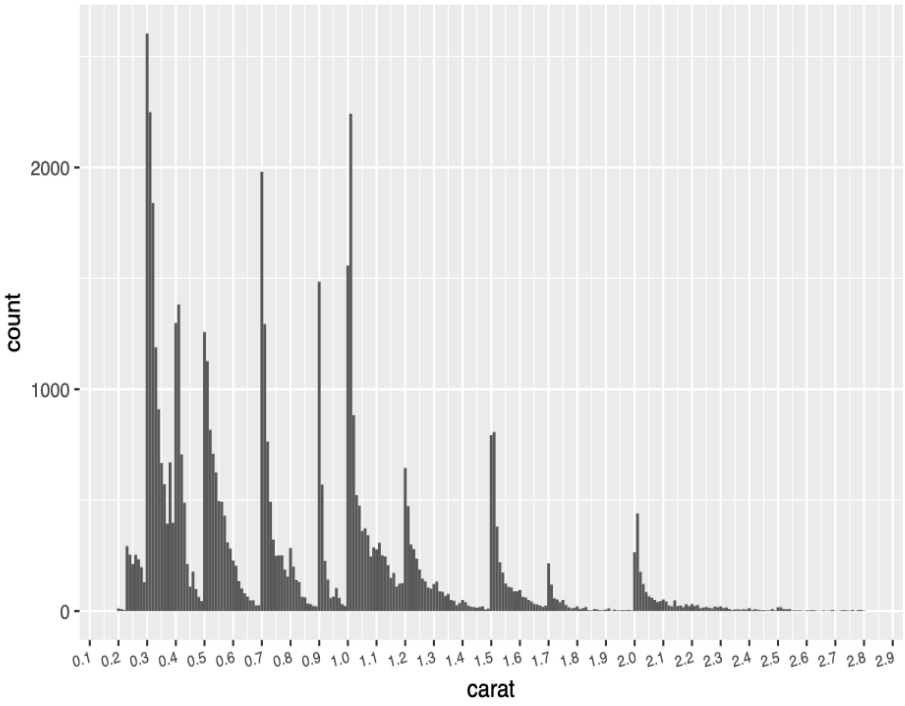
The graph shows clearly the positive relationship between carat and price. But the relationship is not a simple linear relationship. Firstly, there are much fewer diamonds heavier than 3 carats. Secondly, it appears that more diamonds are concentrated around certain carats such as 1,1.5, and 2. We can plot the histogram of the weight variable.

```
ggplot(diamonds, aes(x = carat)) + geom_histogram(binwidth = 0.1 ) + scale_x_continuous( breaks = seq (
```



The histogram shows the same pattern we have observed from the last plot. Let's take a more detailed look at diamonds whose weights are less than 3 carats.

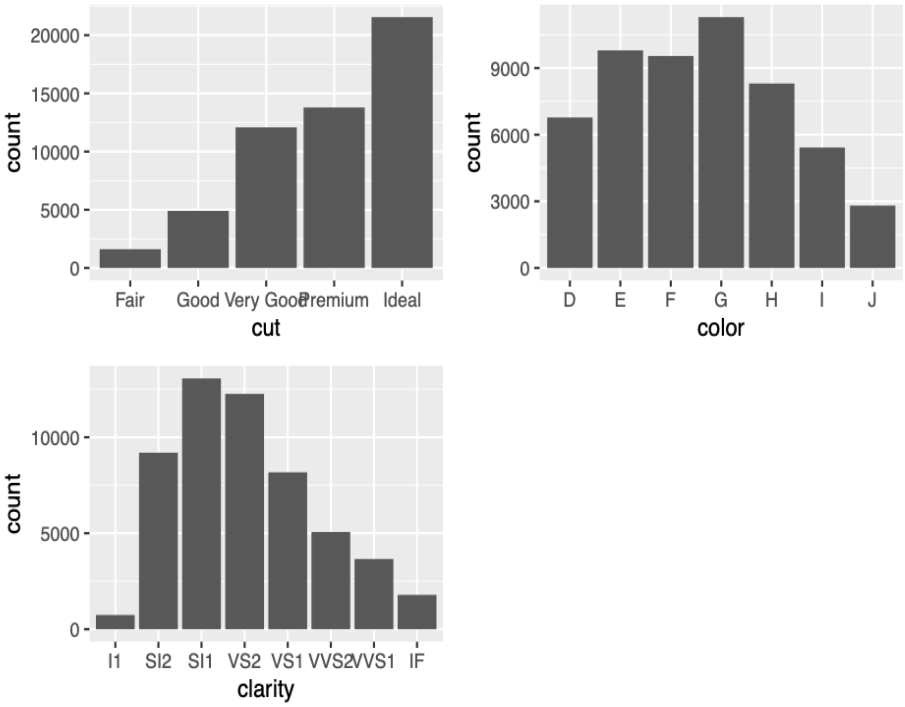
```
ggplot(diamonds %>% filter(carat<3), aes(x = carat)) + geom_histogram(binwidth = 0.01 ) + scale_x_contii
```



Apparently, people like diamonds of 0.3,0.4,0.5,0.7,0.9 carat levels of diamond. In each cluster, it shows the declining trend until reaching the next peak. When the diamond becomes bigger, people favor diamonds weighing 1, 1.2, 1.5, then 2 carats.

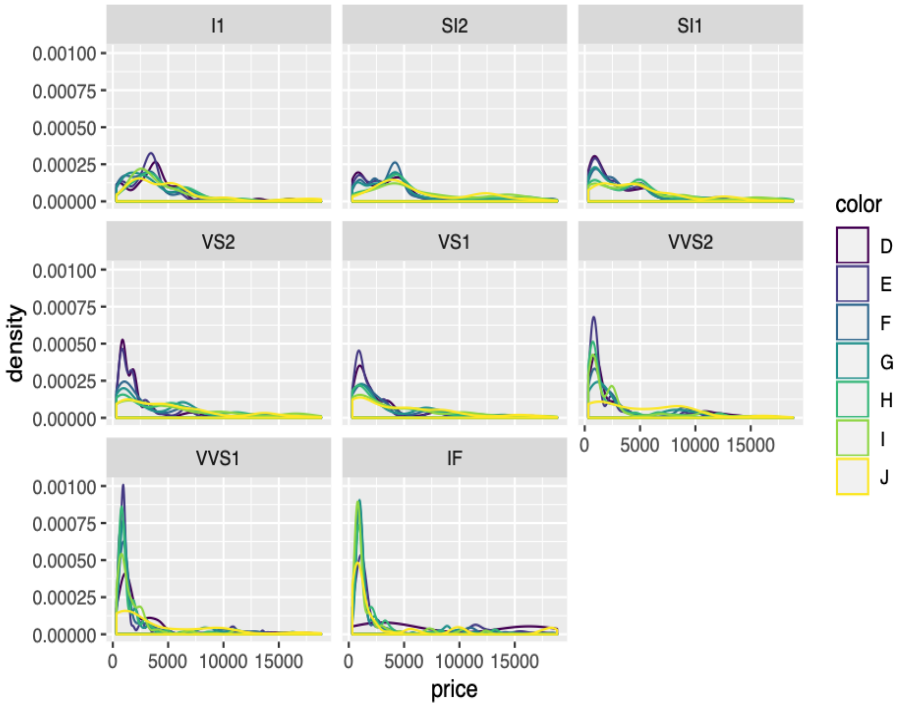
Then we can take a look at how different criterias of diamonds(cut, color, and clarity) are distributed.

```
g1= ggplot(data = diamonds, aes(x=cut)) + geom_bar()
g2=ggplot(data = diamonds, aes(x=color)) + geom_bar()
g3= ggplot(data = diamonds, aes(x=clarity)) + geom_bar()
grid.arrange(g1,g2,g3,ncol=2)
```



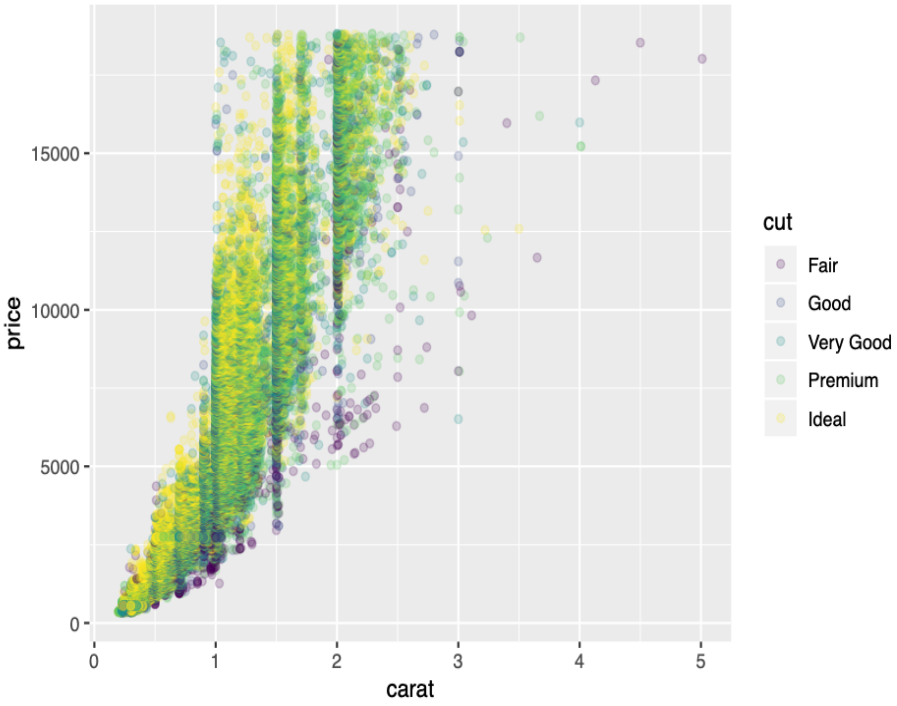
Most diamonds are cutted in an ideal condition and the colors. Next, we are interested in how the price of diamonds are affected by these criteria.

```
ggplot(diamonds, aes(x= price, color = color))  + geom_density() + scale_fill_brewer(palette = "Dark2")
```

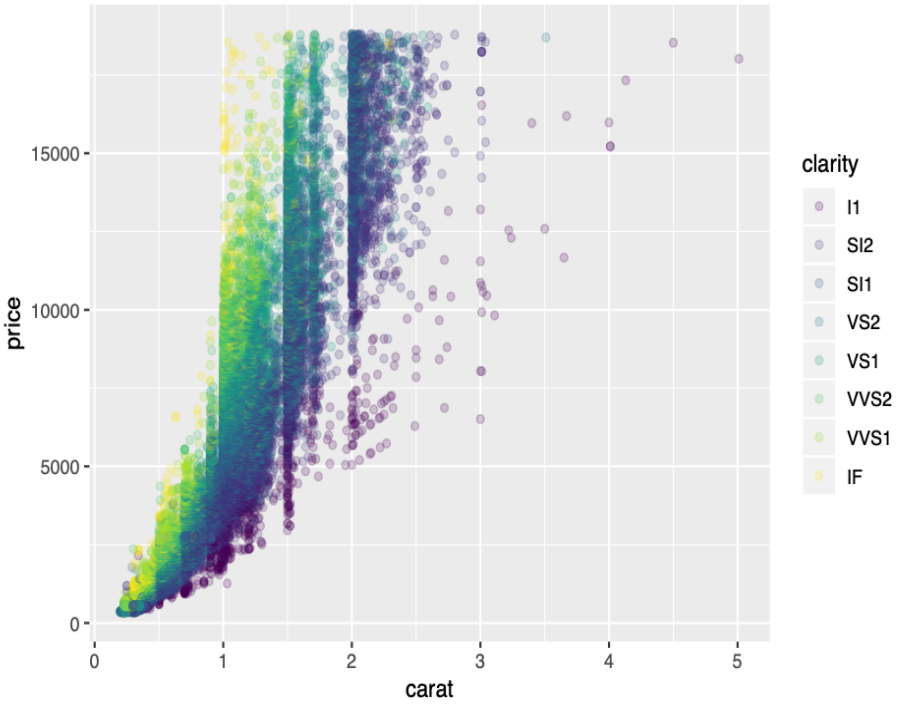


Each subplot is the price density distribution of different clarities while different colors mean different color gradings of diamonds. One thing peculiar here is the top color class diamonds density are accumulated in a relatively low price range, because the carat of a diamond is a very important factor. There are generally more small diamonds so this density plot may cause confusion about the dataset and we need another visualization of how these criterias affect the price given the same weight.

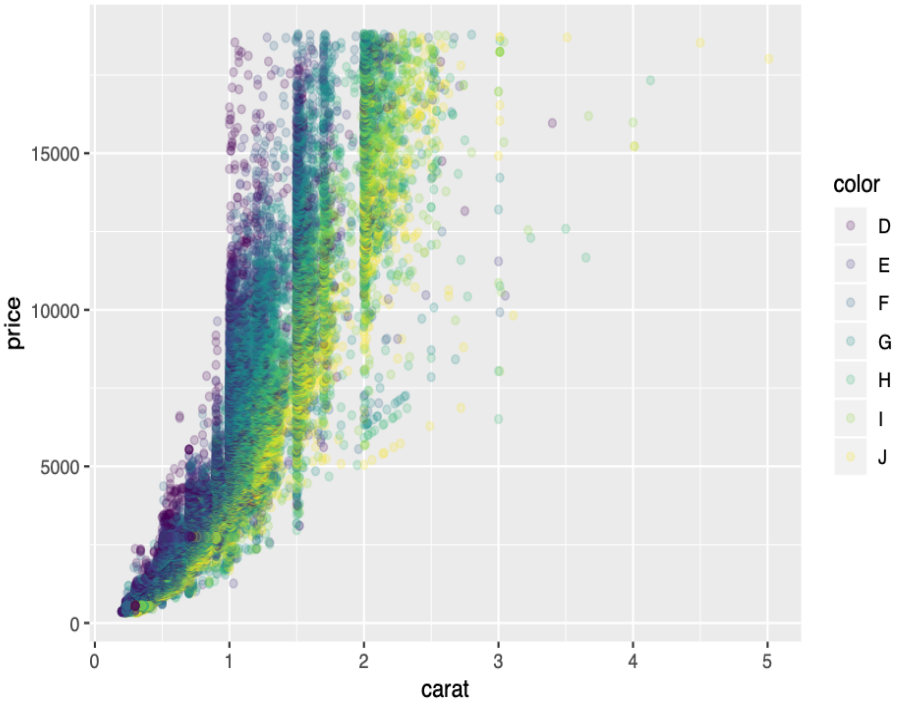
```
ggplot(diamonds,aes(x = carat,y = price, color = cut)) + geom_point(alpha=0.2)
```




```
ggplot(diamonds,aes(x = carat,y = price, color = clarity)) + geom_point(alpha=0.2)
```

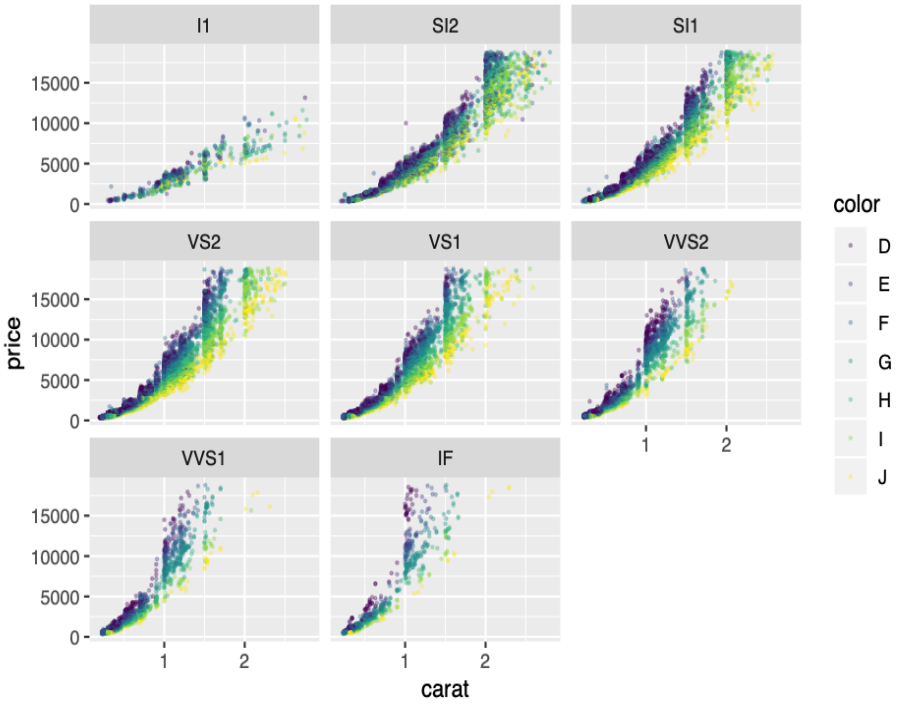



```
ggplot(diamonds,aes(x = carat,y = price, color = color)) + geom_point(alpha=0.2)
```



From the first plot, we do see **fair** cut diamonds generally have lower price compared to other cutting levels but other cutting conditions are mingled together, which means bad cutting would affect the price badly while good cutting does not necessarily bump up the price. Clarity seems to be an important factor of the price because we can see apparent level separation from the second plot. Also, as color level improces from D to J, the price decreases. Actually, diamond color is the second most important factor and D means the best color level.

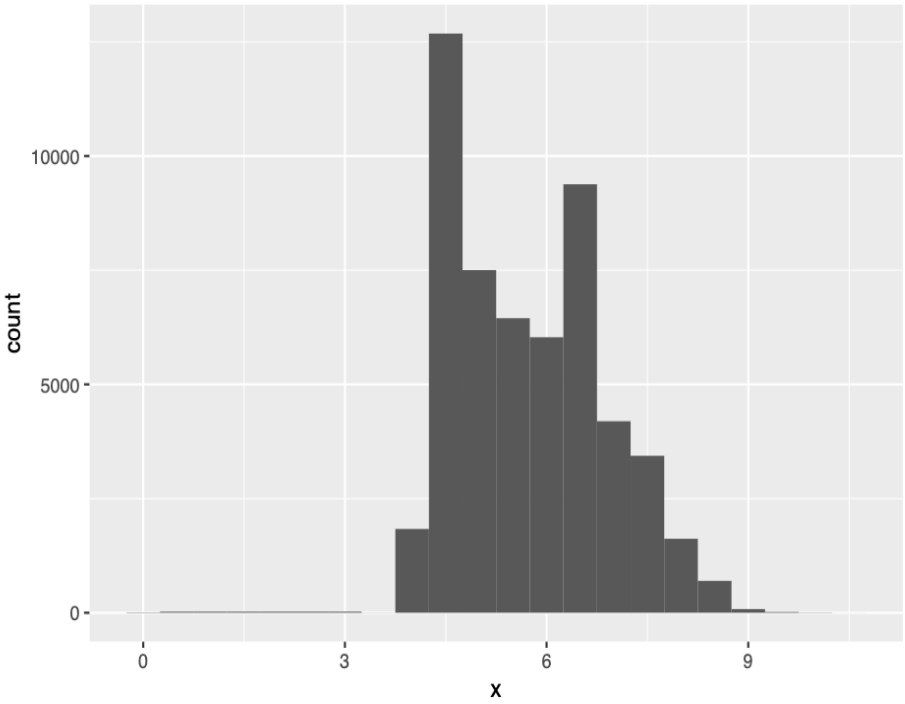
```
ggplot(diamonds %>% filter(carat < 3), aes(x= carat, y = price, colour = color))  + geom_point(alpha = (
```



This plot shows the effect of color and clarity on the price. As clarity becomes better, the slope of data points increases, which suggests positive relationship. Also, the darker points in each subplot, which indicates better color quality of diamonds, generally have higher price compared to others.

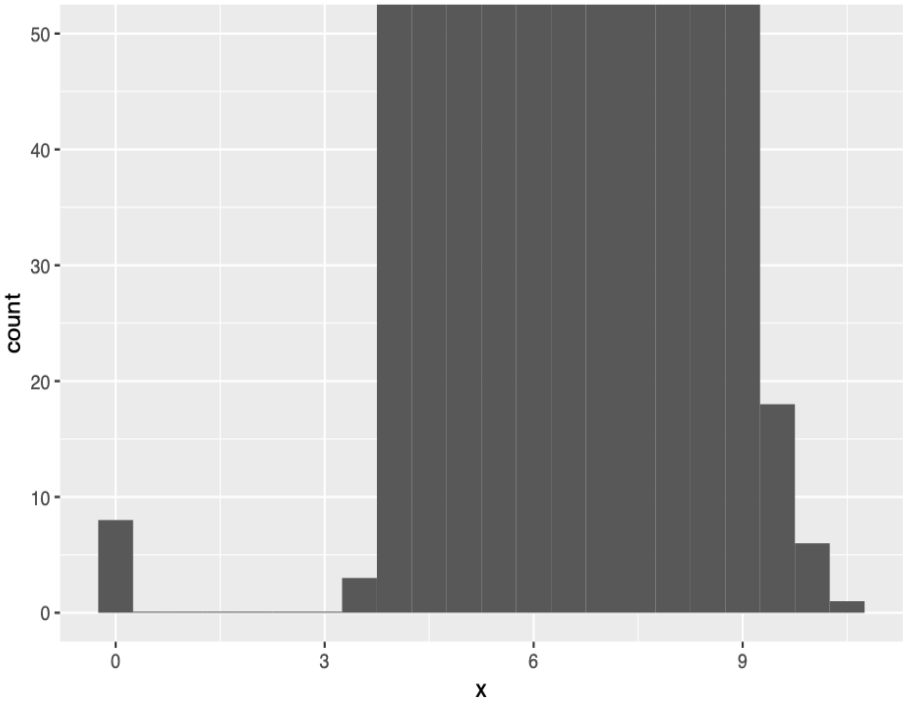
Now, we can take a look at the x,y,z parameter of a diamond and how they are related with each other. Still, first we can inspect the distribution of x variable.

```
ggplot(diamonds, aes(x = x)) +
  geom_histogram(binwidth = 0.5)
```



There are too many observations in the common bins but we notice there is something unusual on the position 0. We can zoom closer to see those smaller value on the y-axis around 0 on the x-axis with `coord_cartesian()`

```
ggplot(diamonds, aes(x = x)) +
  geom_histogram(binwidth = 0.5) + coord_cartesian(ylim= c(0,50))
```



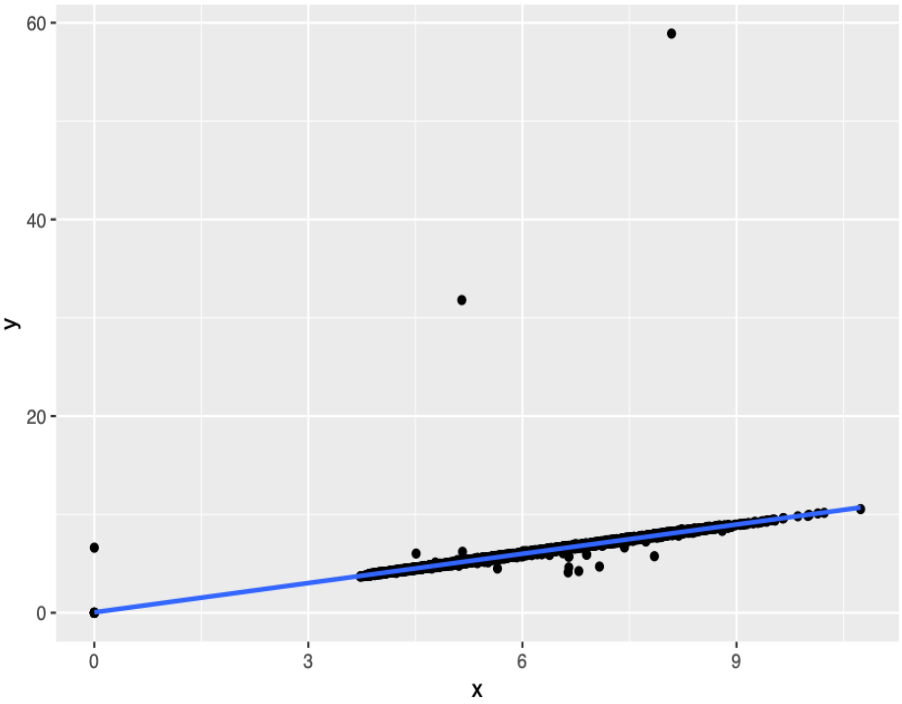
Let's take a look at those abnormal data with x=0.

```
diamonds %>%
  filter(x==0) %>%
  select(price,x,y,z)
```

```
## # A tibble: 8 x 4
##   price     x     y     z
##   <int> <dbl> <dbl> <dbl>
## 1  4954     0  6.62     0
## 2  5139     0  0        0
## 3  6381     0  0        0
## 4 12800     0  0        0
## 5 15686     0  0        0
## 6 18034     0  0        0
## 7  2130     0  0        0
## 8  2130     0  0        0
```

It is apparent diamonds cannot have 0 width. It is likely these data points have missing values of variable x,y,and z and some people substitube them with 0s. Let's take a look at the relationship between the x and y.

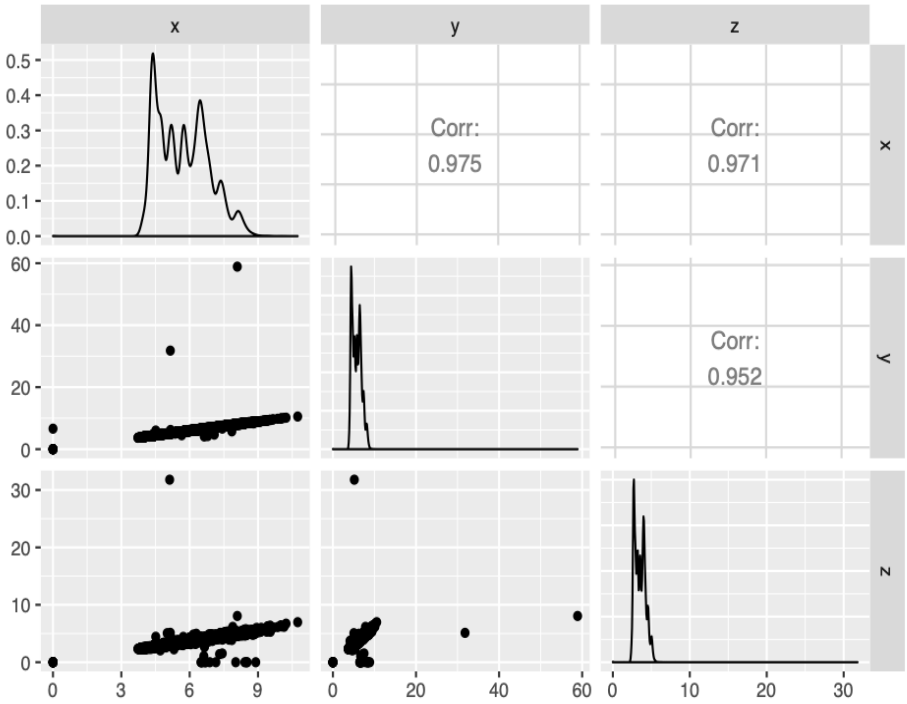
```
ggplot(diamonds,aes(x=x,y=y)) + geom_point() +  geom_smooth(method = "lm")
```



It turns out they have highly linear correlation while we can see the outliers with x=0 and y=0 and those with very high y value. If we want to do further analysis, we would want to get rid of those outliers first.

Finally let's take a look at the relationship between x,y, and z.

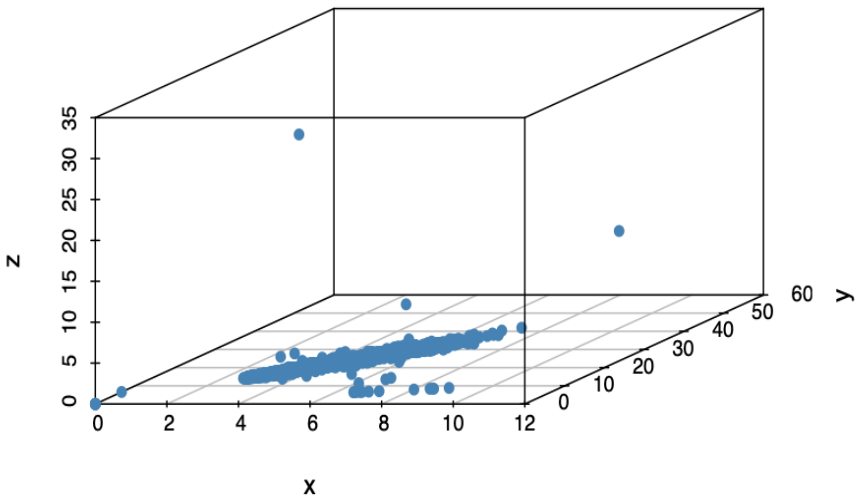
```
ggpairs(diamonds %>% select(x,y,z))
```



From the pair graph, we can see x,y,z are very correlated with each other.

```
library(scatterplot3d)
scatterplot3d(diamonds %>% select('x','y','z'), pch = 16, color="steelblue")
```

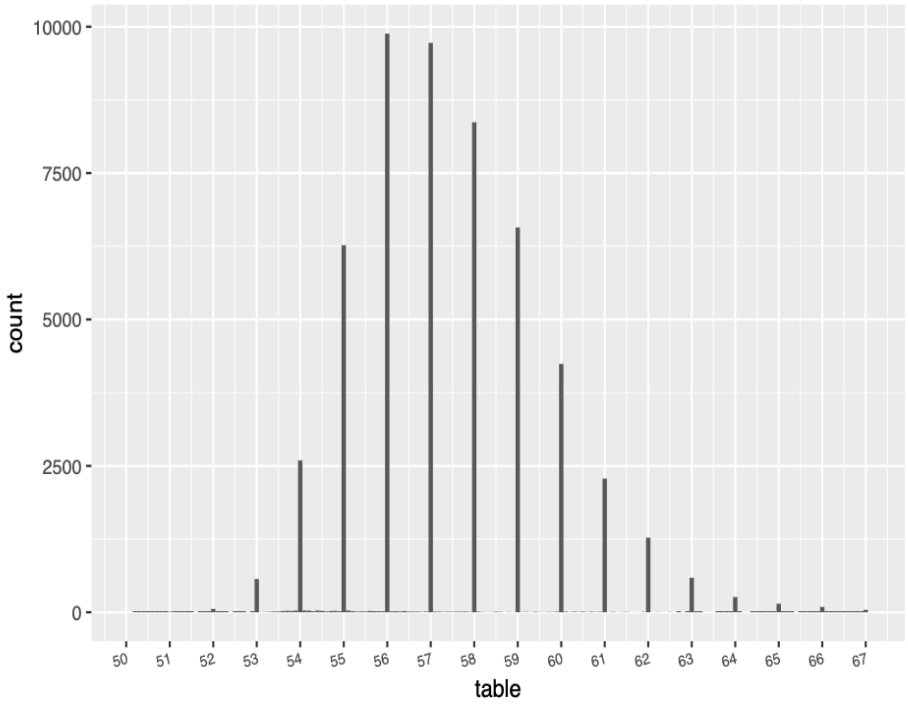
```
## Warning: Unknown or uninitialised column: 'color'.
```



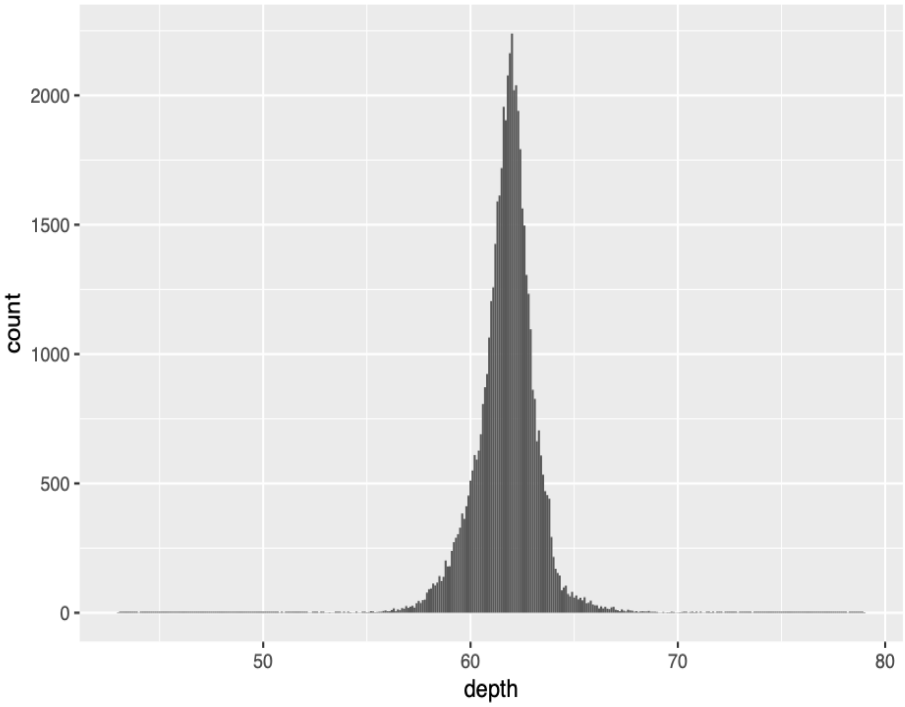
In the 3D space, besides the outliers, it is a perfect straight line. This makes sense because the cut of a diamond has to be subjected to certain rules of proportion.

Finally, let's inspect the table and depth variable.

```
ggplot(diamonds %>% filter (table<68 & table>50) ,aes(x=table)) + geom_histogram(binwidth = 0.1) +  sca
```




```
ggplot(diamonds,aes(x=depth)) + geom_histogram(binwidth = 0.1)
```



The table variable is apparently a categorical value rather than a continuous one. We should pay attention to this in our further analysis.