Day 7 code

Anil Kumar Yadav

10/6/2019

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.

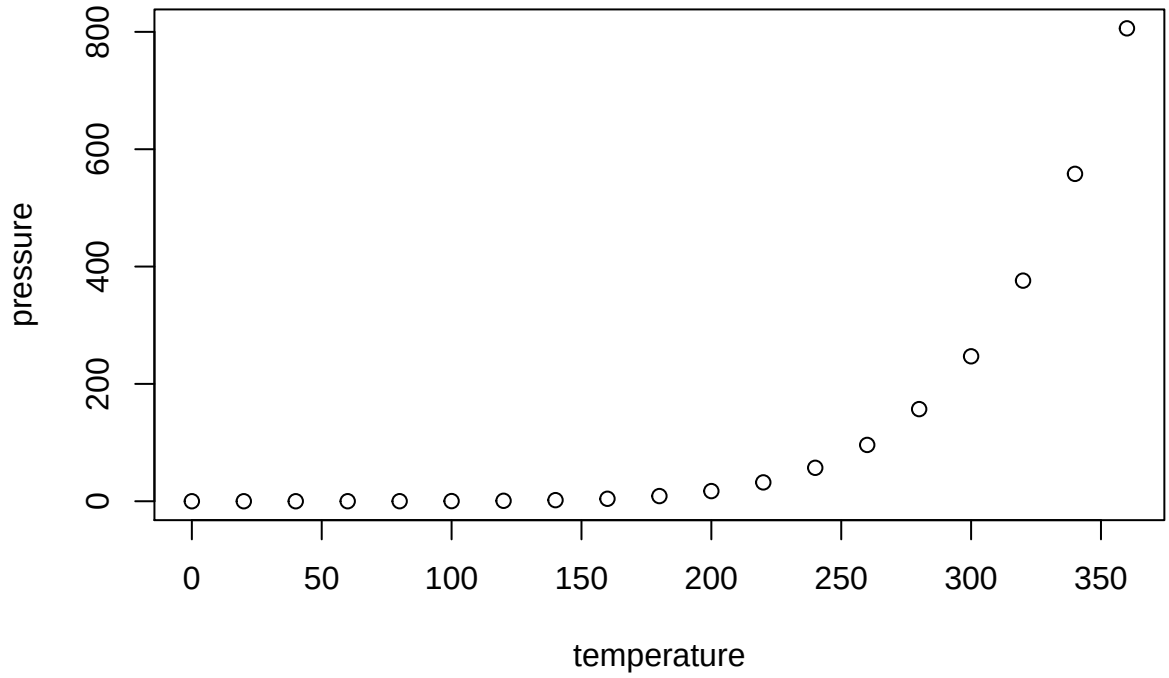
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Loading required libraries

```
library(tidyverse)
```

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

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

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

```
library(ggplot2)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(hexbin)
```

Understading the data (data validation)

```
data("diamonds")
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1 0.23  Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2 0.21  Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
## 3 0.23  Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4 0.290 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5 0.31  Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6 0.24  Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

```
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 ...
```

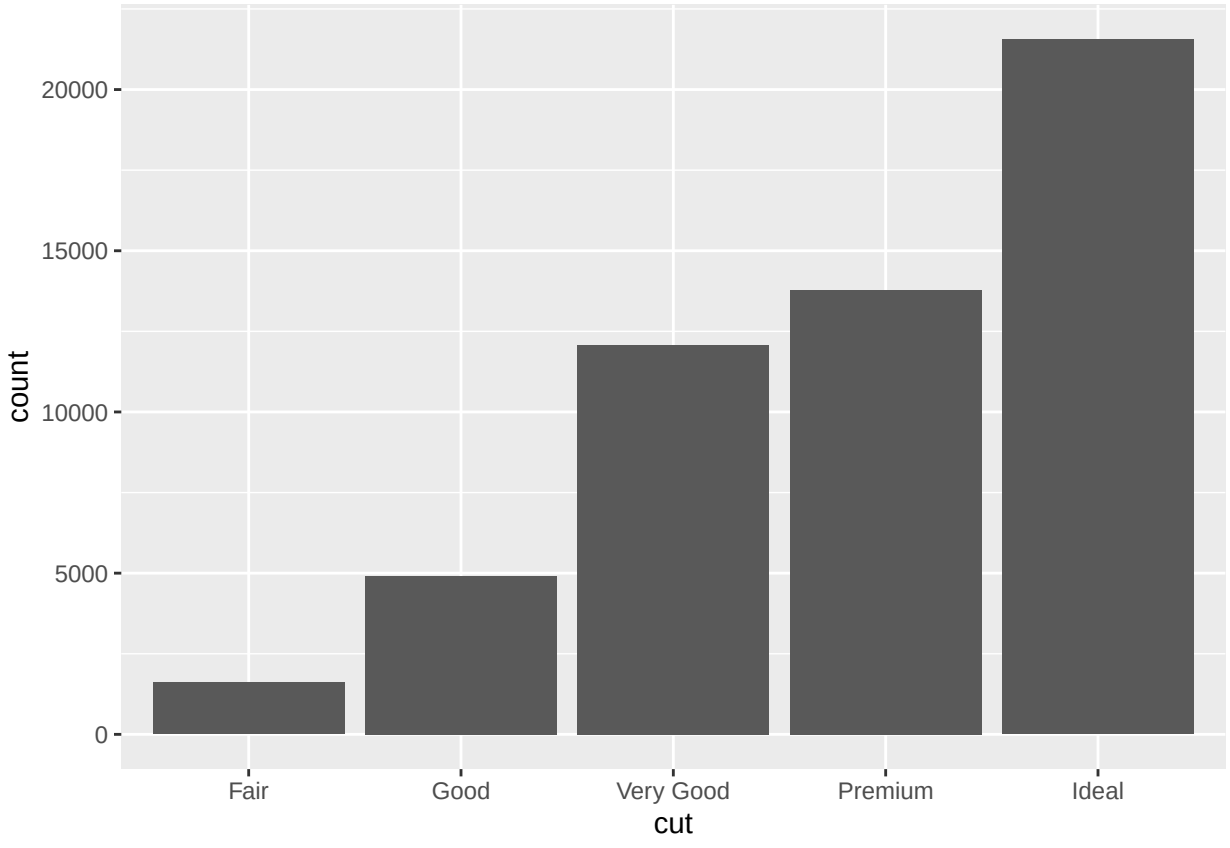
```
summary(diamonds)
```

```
##      carat               cut        color        clarity
##  Min.   :0.2000   Fair      : 1610   D: 6775   SI1    :13065
##  1st Qu.:0.4000   Good      : 4906   E: 9797   VS2    :12258
##  Median :0.7000   Very Good:12082   F: 9542   SI2    : 9194
##  Mean   :0.7979   Premium   :13791   G:11292   VS1    : 8171
##  3rd Qu.:1.0400   Ideal     :21551   H: 8304   VVS2   : 5066
##  Max.   :5.0100                     I: 5422   VVS1   : 3655
##                                     J: 2808   (Other): 2531
##      depth           table           price             x
##  Min.   :43.00   Min.   :43.00   Min.   :  326   Min.   : 0.000
```

```
##  1st Qu.:61.00   1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710
##  Median :61.80   Median :57.00   Median : 2401   Median : 5.700
##  Mean   :61.75   Mean   :57.46   Mean   : 3933   Mean   : 5.731
##  3rd Qu.:62.50   3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540
##  Max.   :79.00   Max.   :95.00   Max.   :18823   Max.   :10.740
##
##        y                z
##  Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 4.720   1st Qu.: 2.910
##  Median : 5.710   Median : 3.530
##  Mean   : 5.735   Mean   : 3.539
##  3rd Qu.: 6.540   3rd Qu.: 4.040
##  Max.   :58.900   Max.   :31.800
##
```

Plotting

```
ggplot(data = diamonds)+geom_bar(mapping= aes(x=cut))
```



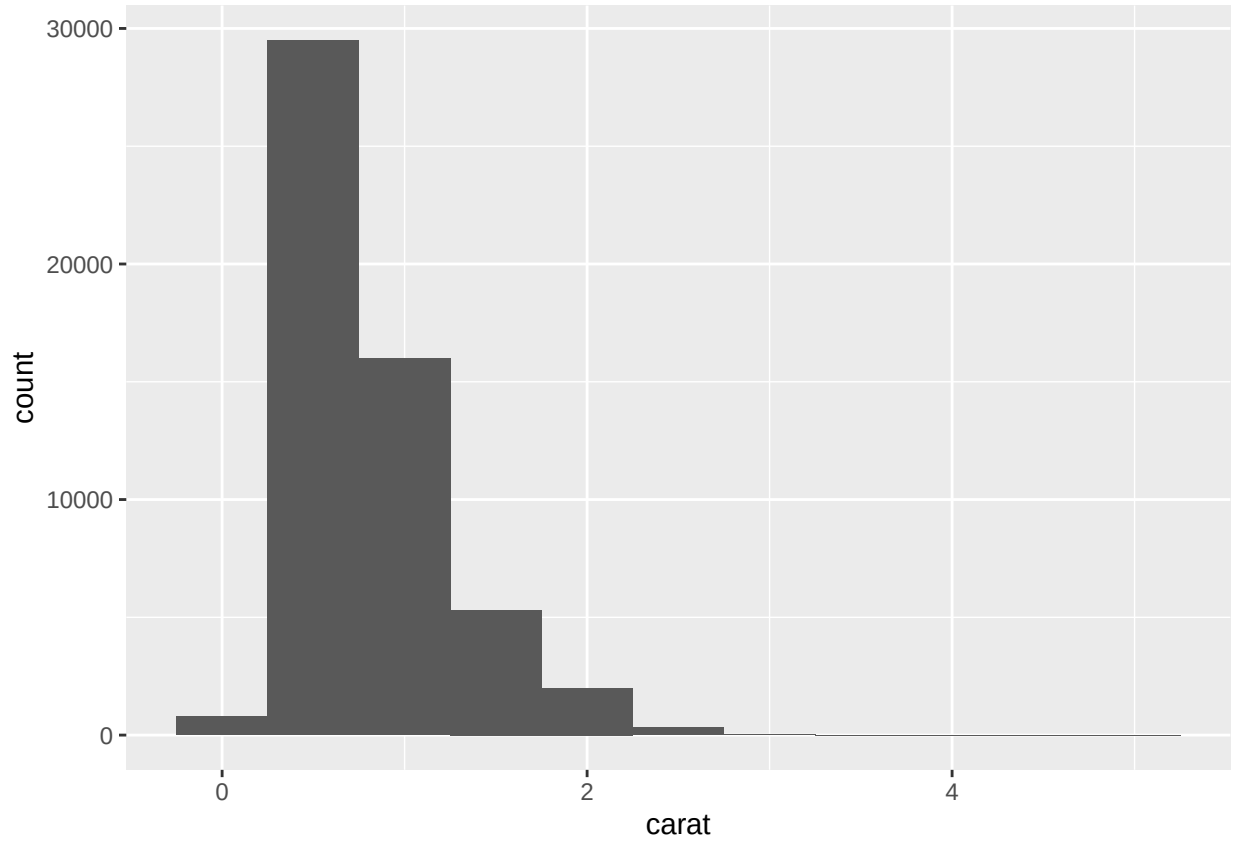

```
diamonds %>% count(cut)
```

```
## # A tibble: 5 x 2
##   cut           n
##   <ord>     <int>
## 1 Fair       1610
```

```
## 2 Good       4906
## 3 Very Good 12082
## 4 Premium   13791
## 5 Ideal     21551
```

Histogram along with bin width

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



Arranging with bin width as 0.5

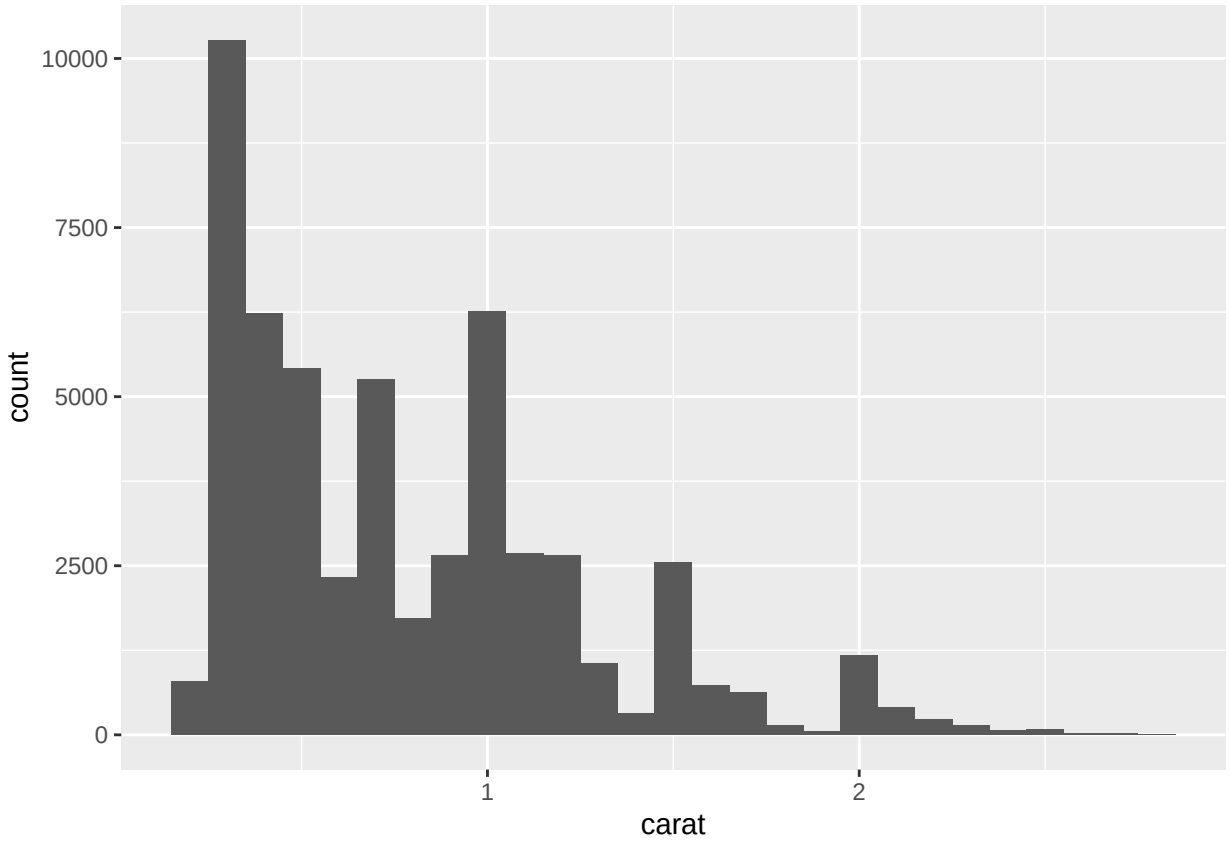
```
diamonds %>% count(cut_width(carat, 0.5))
```

```
## # A tibble: 11 x 2
##    `cut_width(carat, 0.5)`     n
##    <fct>                   <int>
##  1 [-0.25,0.25]              785
##  2 (0.25,0.75]             29498
##  3 (0.75,1.25]             15977
##  4 (1.25,1.75]              5313
##  5 (1.75,2.25]              2002
##  6 (2.25,2.75]               322
##  7 (2.75,3.25]                32
##  8 (3.25,3.75]                 5
##  9 (3.75,4.25]                 4
```

```
## 10 (4.25,4.75]                 1
## 11 (4.75,5.25]                 1
```

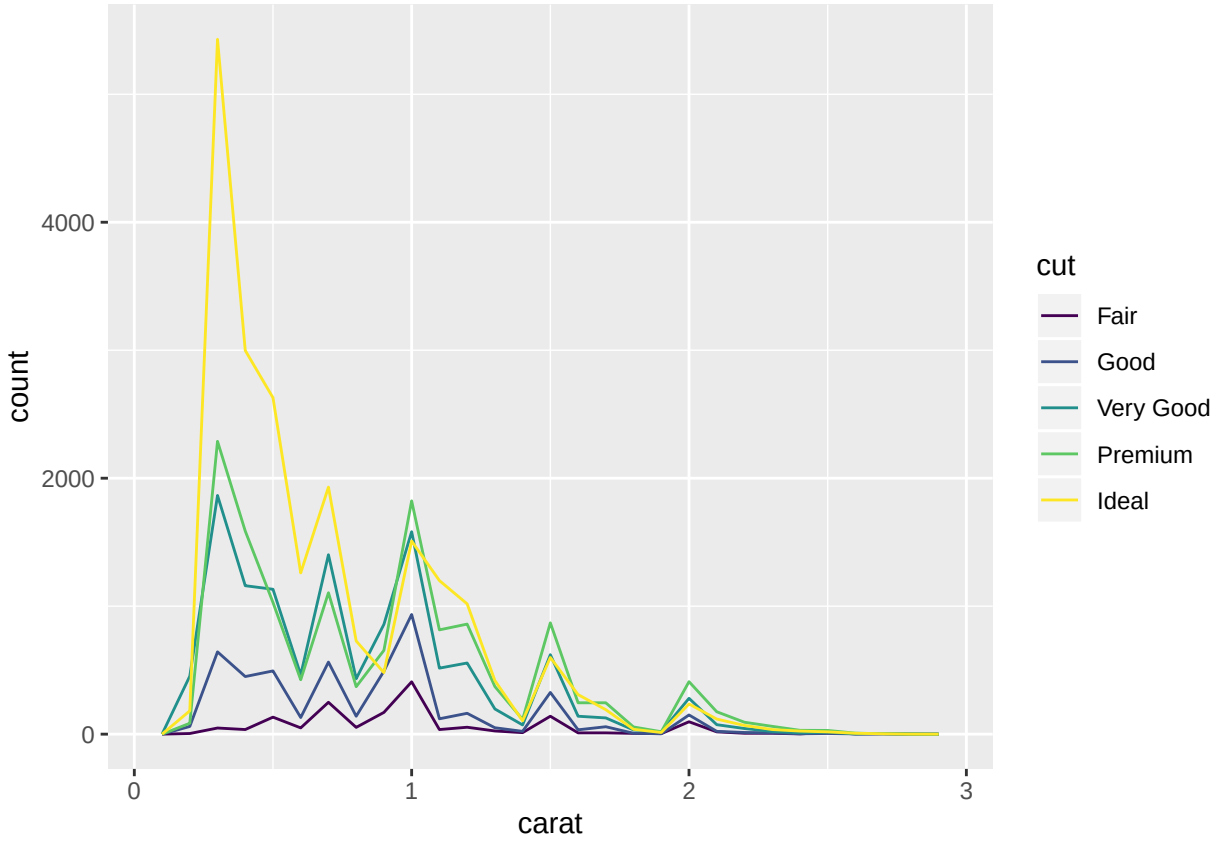
Specifying particular coulmn of carat size less than 3

```
smaller <- diamonds %>%
  filter(carat <3)
ggplot(data= smaller, mapping = aes(x=carat)) + geom_histogram(binwidth=0.1)
```



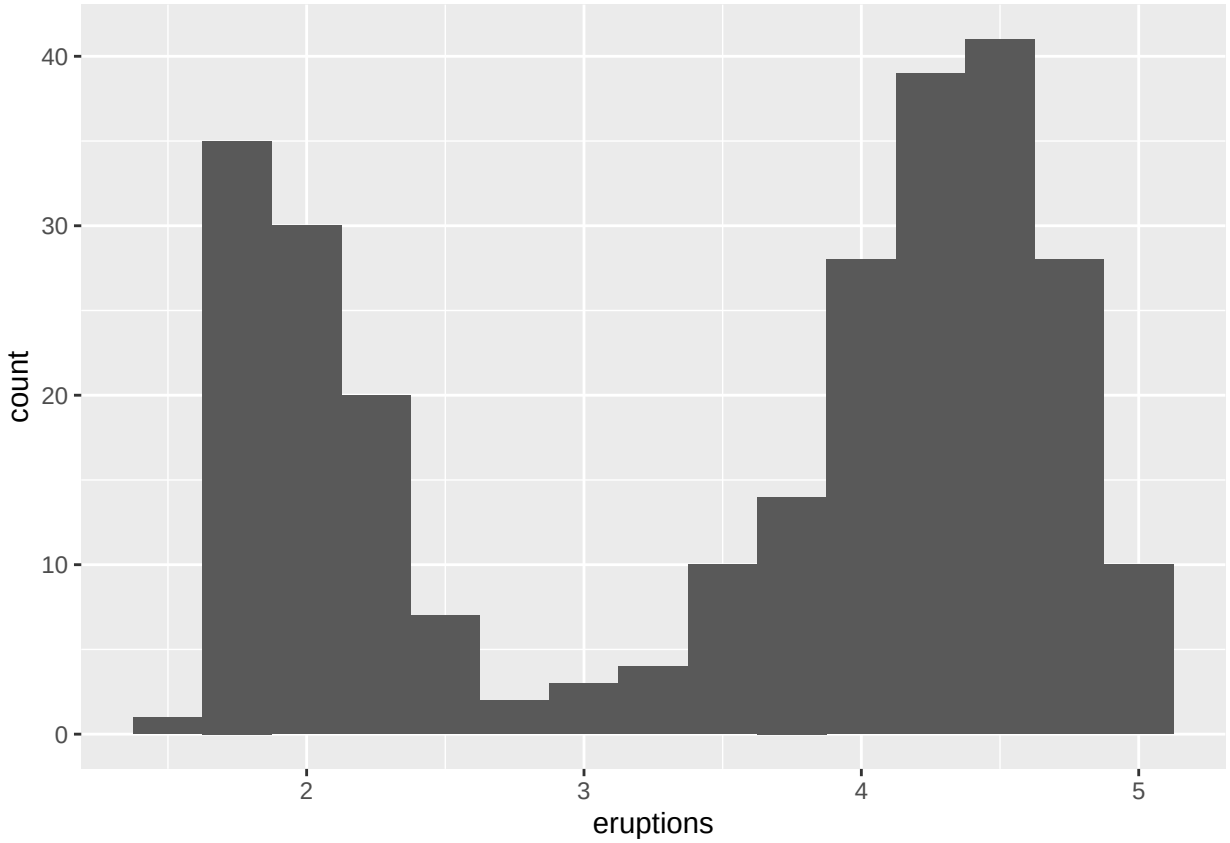
Use geom_greqploy() function for overlaying multiple histograms

```
ggplot(data= smaller, mapping = aes(x= carat, colour= cut))+ geom_freqpoly(binwidth = 0.1)
```



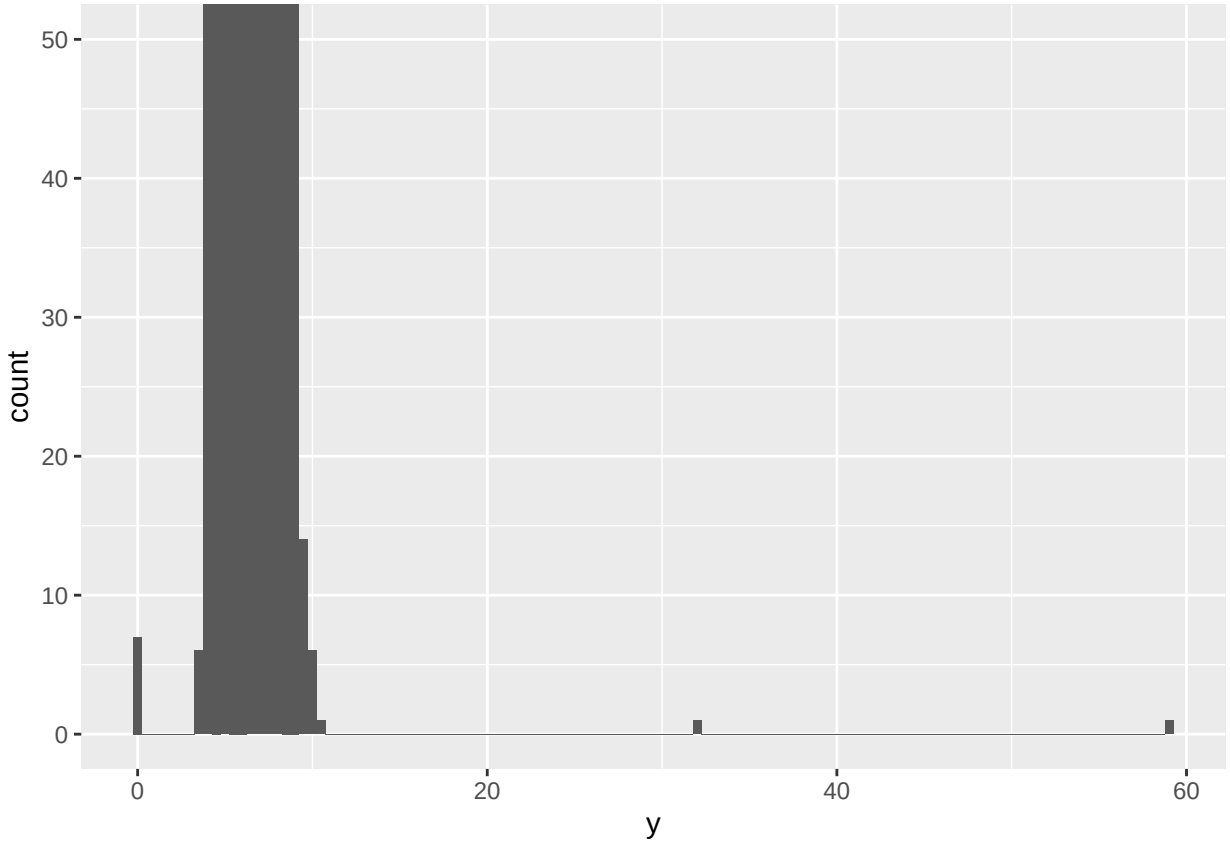
hiistogram of erupption times

```
ggplot(data=faithful, mapping = aes(x=eruptions))+ geom_histogram(binwidth= 0.25)
```



To find the outliers, zoom the samllest values of y-axis with coord_cartesian()

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

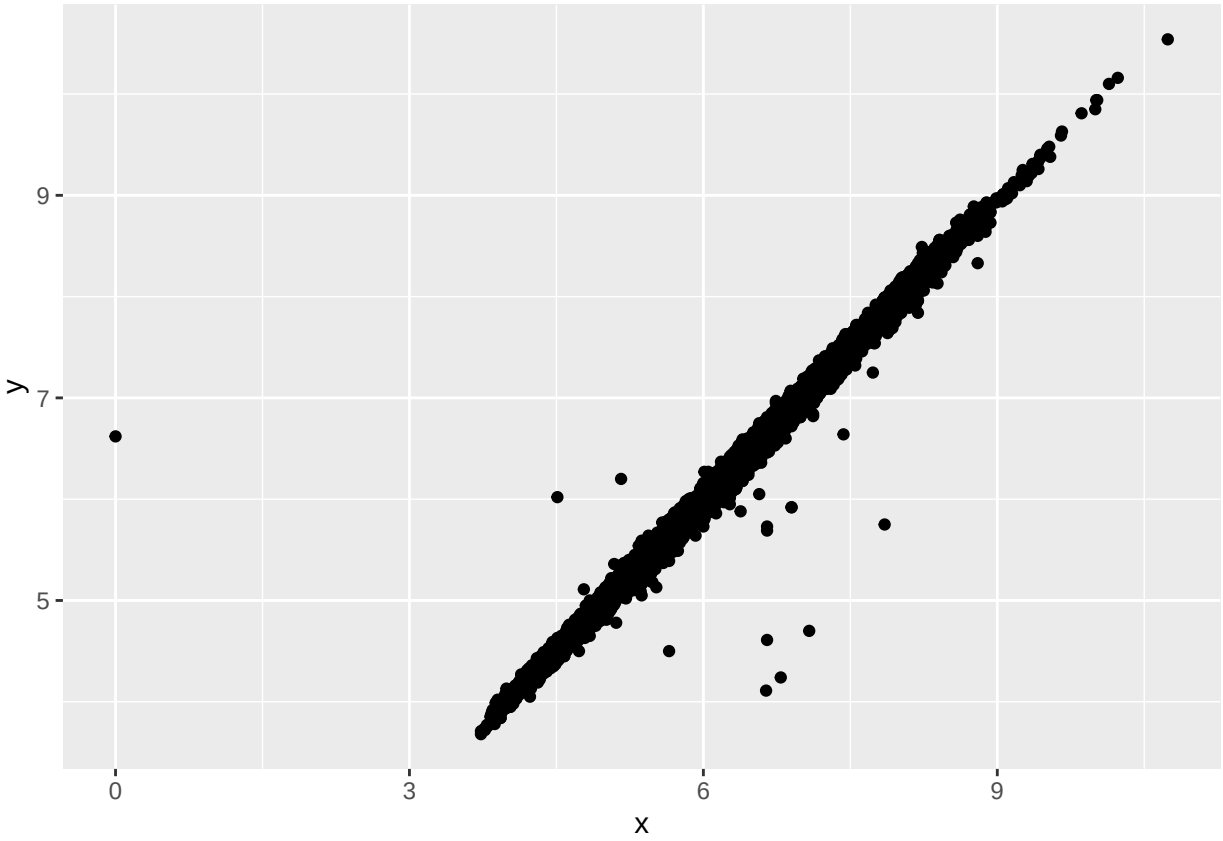


Missing Values Replacing the unsual(outliers) values with missing values

```
diamonds2 <- diamonds %>% mutate(y=ifelse(y<3 | y>20, NA, y))
```

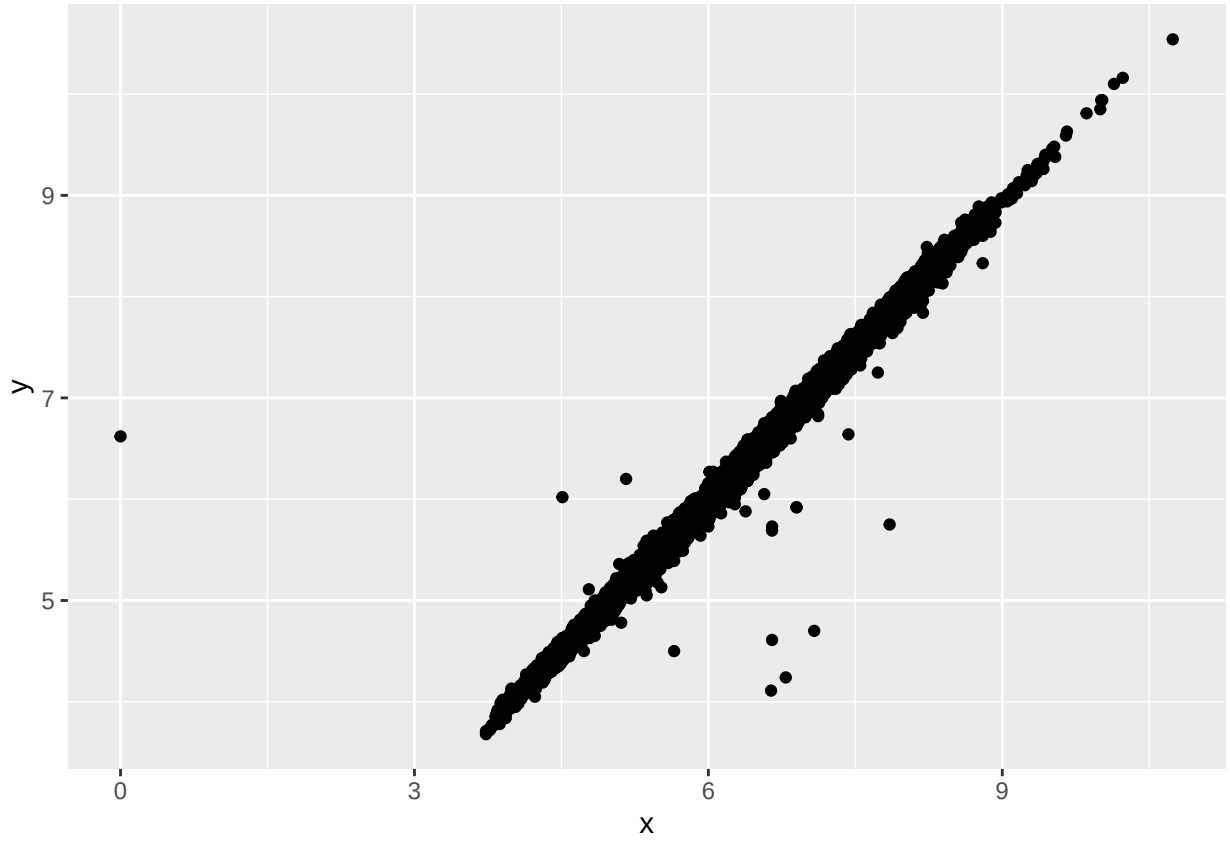
```
ggplot(data=diamonds2, mapping = aes(x=x, y=y))+ geom_point()
```

```
## Warning: Removed 9 rows containing missing values (geom_point).
```



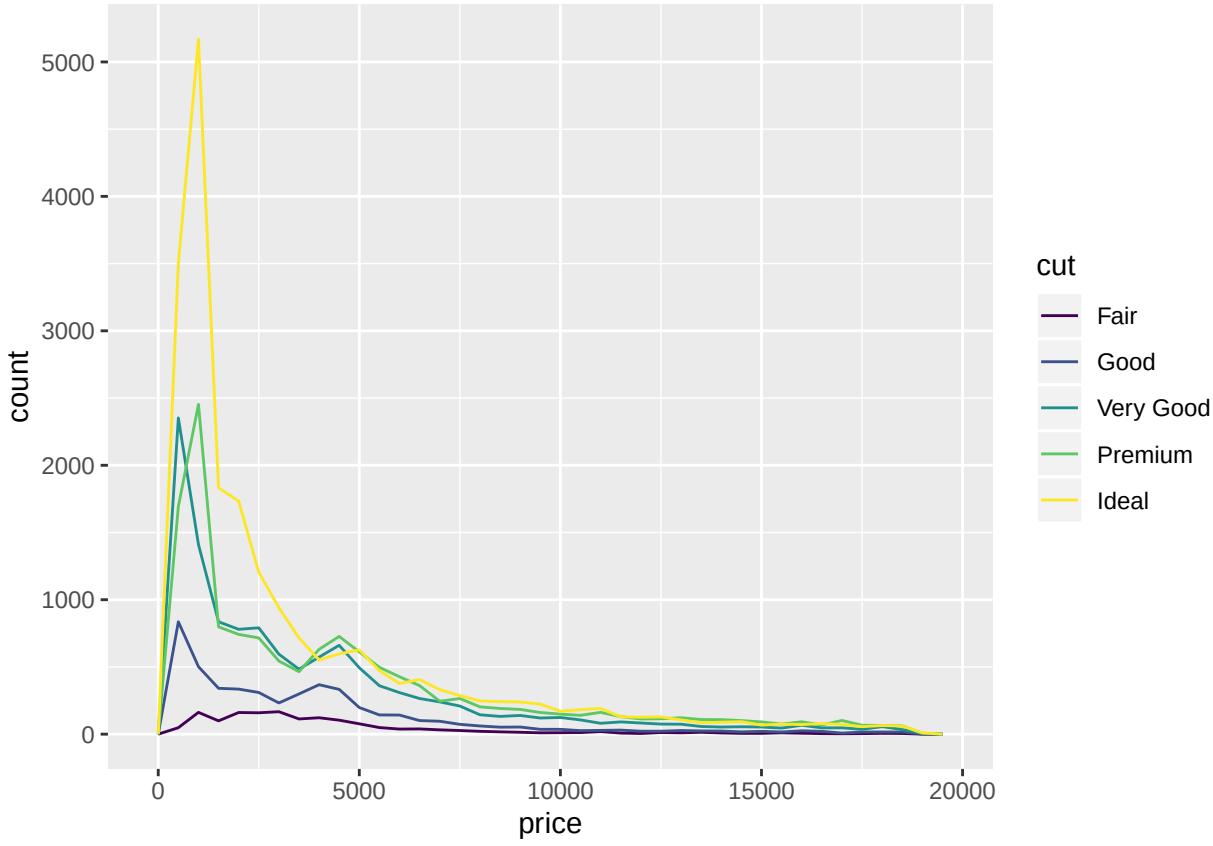
To remove the produced error, use the below line of R code

```
ggplot(data=diamonds2, mapping = aes(x=x, y=y))+ geom_point(na.rm=TRUE)
```



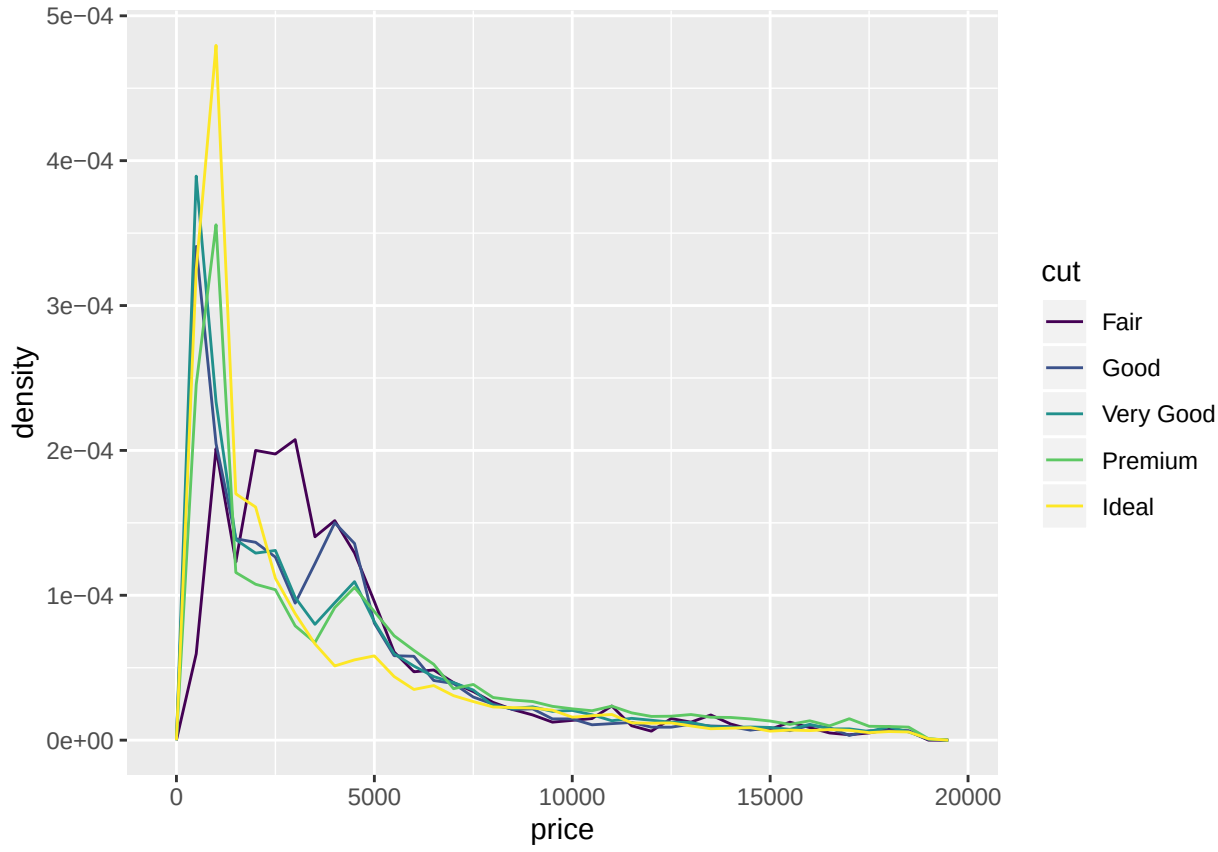
price VS qulaity of diamonds

```
ggplot(data=diamonds, mapping = aes(x=price))+ geom_freqpoly(mapping=aes(colour=cut), binwidth=500)
```



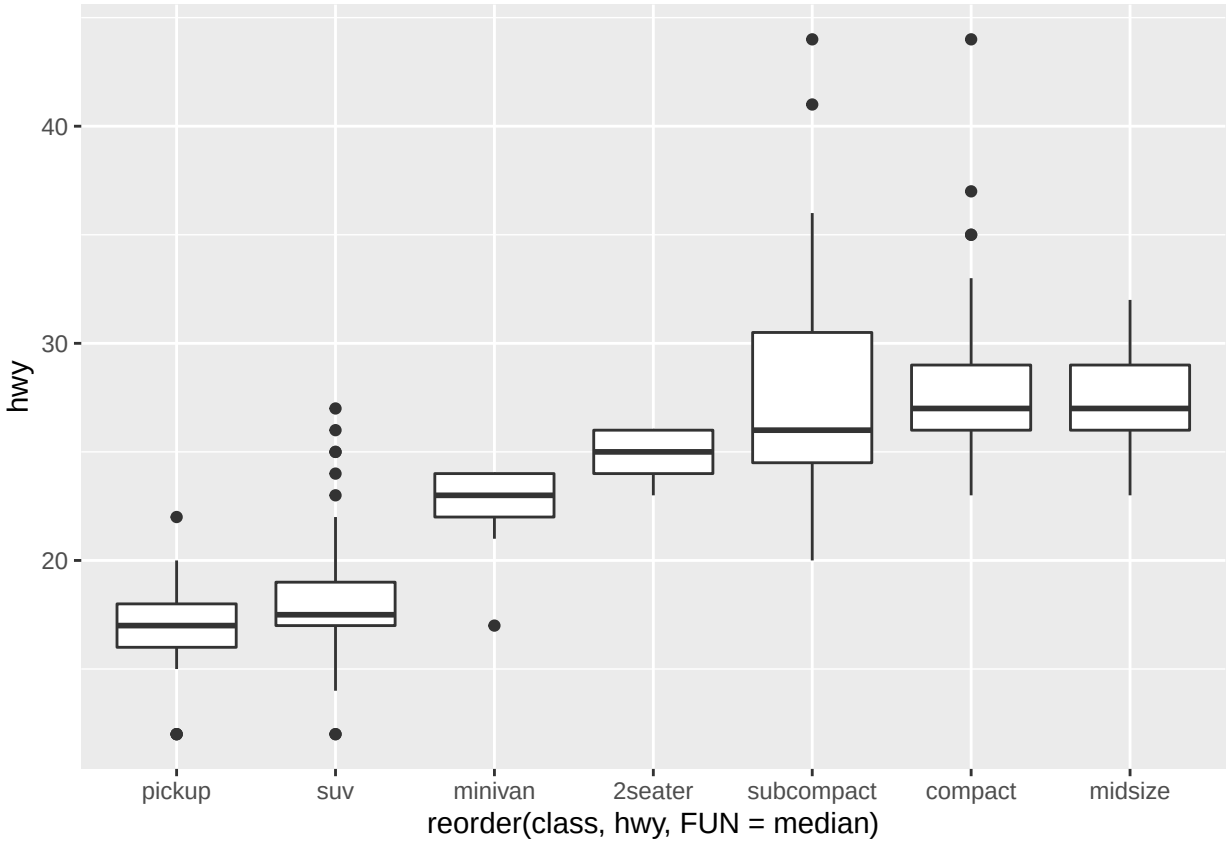
Plotting with density as variable

```
ggplot(data= diamonds, mapping = aes(x=price, y=..density..)) + geom_freqpoly(mapping = aes(colour=cut)
```



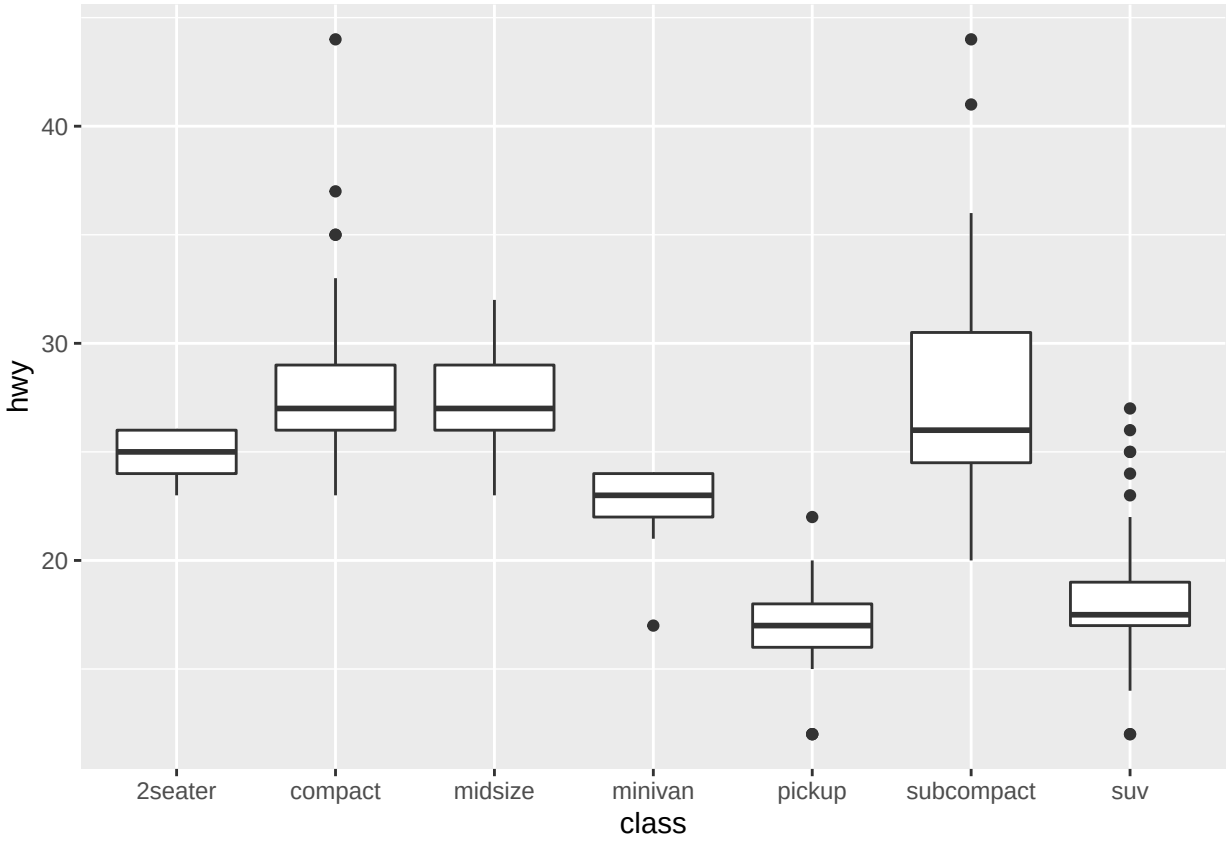
in boxplot

```
ggplot(data=mpg) +  geom_boxplot(mapping = aes(x=reorder(class, hwy, FUN=median ), y=hwy))
```



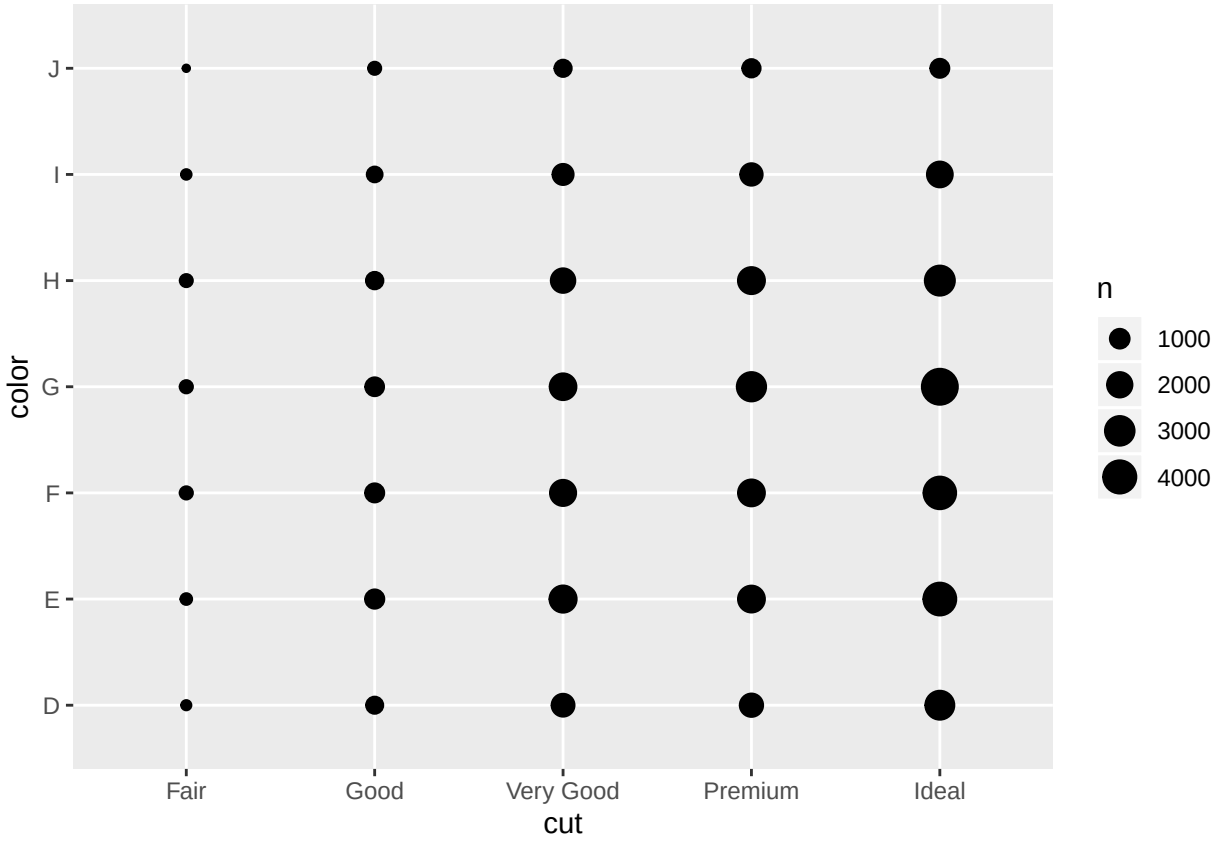
in order to diaplay the values in intrinsc way, use the function reorder()

```
ggplot(data=mpg, mapping = aes(x=class, y=hwy))+ geom_boxplot()
```



The covariation between two variables

```
ggplot(data=diamonds)+ geom_count(mapping = aes(x=cut, y=color))
```

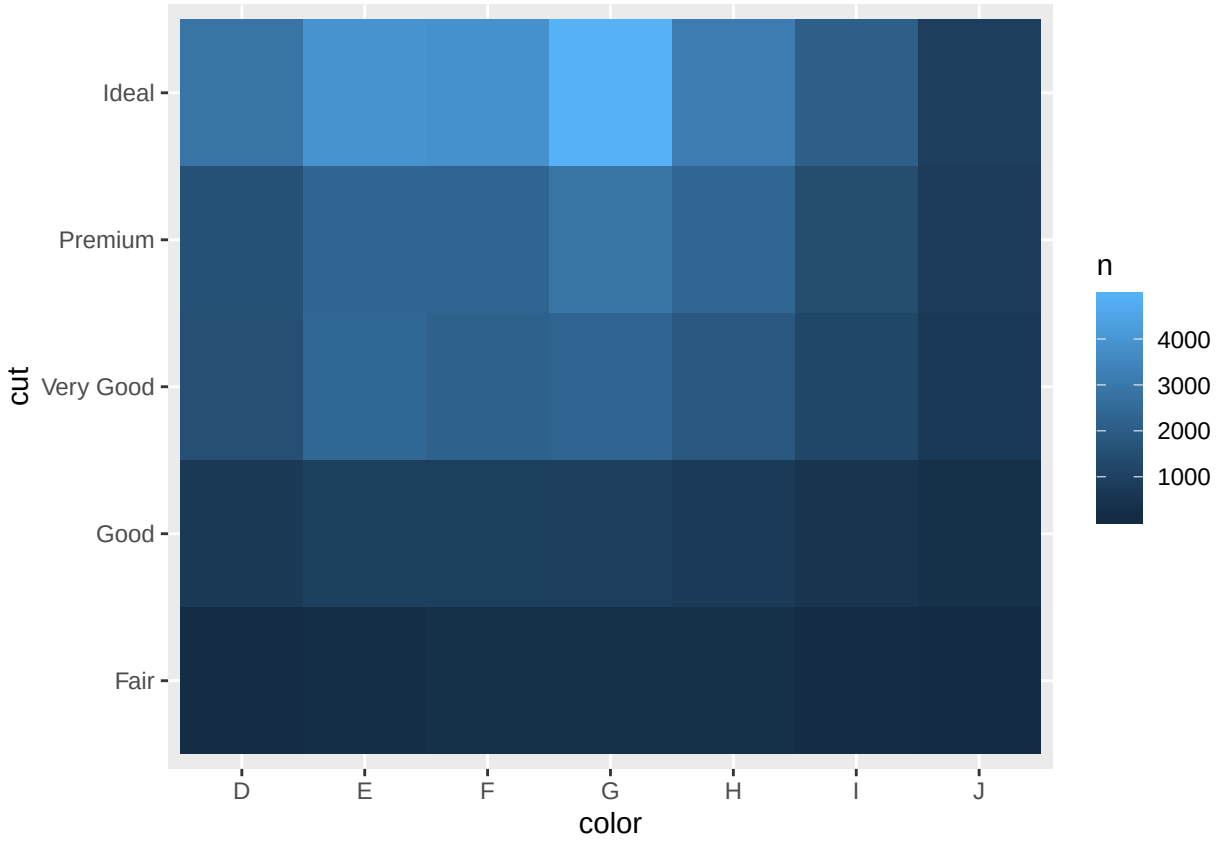


with dplyr

```
diamonds %>% count(color, cut)
```

```
## # A tibble: 35 x 3
##    color cut           n
##    <ord> <ord>     <int>
##  1 D     Fair        163
##  2 D     Good        662
##  3 D     Very Good  1513
##  4 D     Premium    1603
##  5 D     Ideal      2834
##  6 E     Fair        224
##  7 E     Good        933
##  8 E     Very Good  2400
##  9 E     Premium    2337
## 10 E     Ideal      3903
## # ... with 25 more rows
```

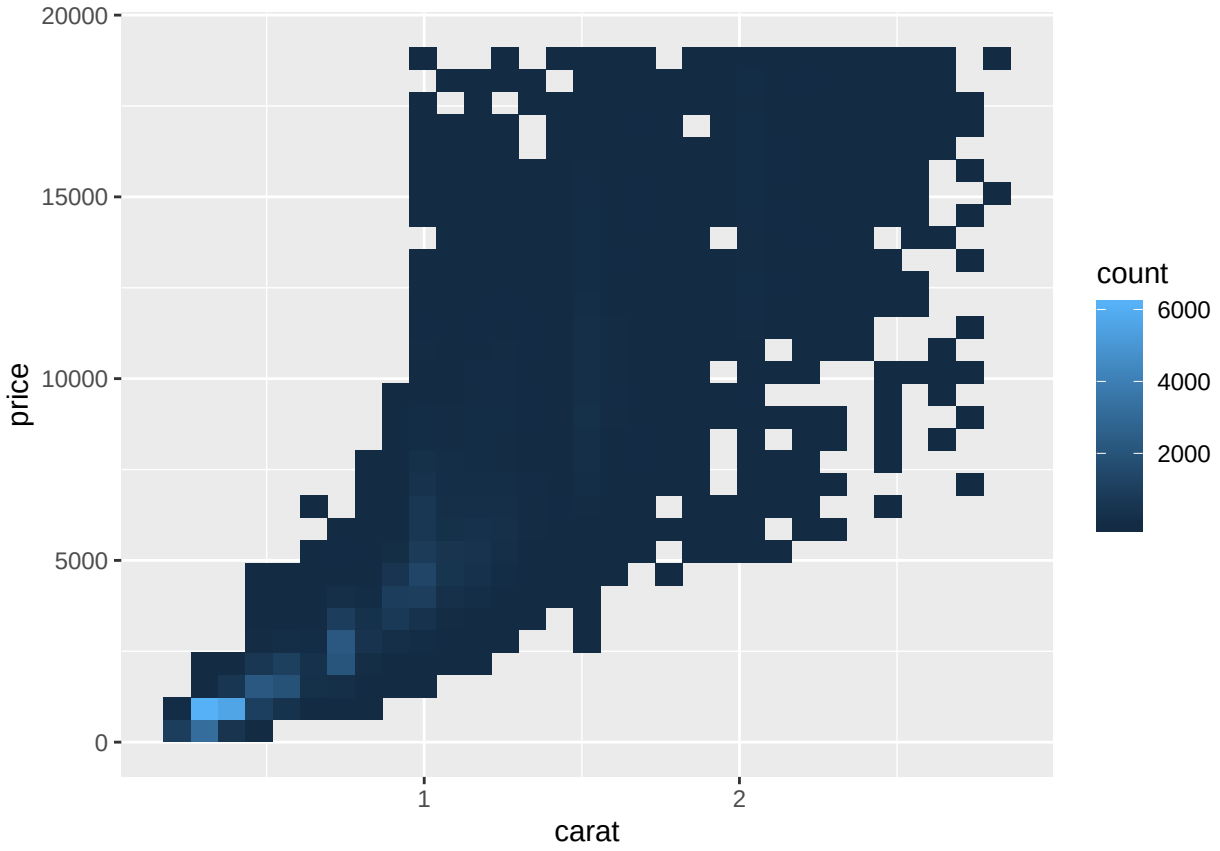
```
diamonds %>%
count(color, cut) %>%
ggplot(mapping=aes(x=color, y=cut ))+ geom_tile(mapping= aes(fill =n))
```



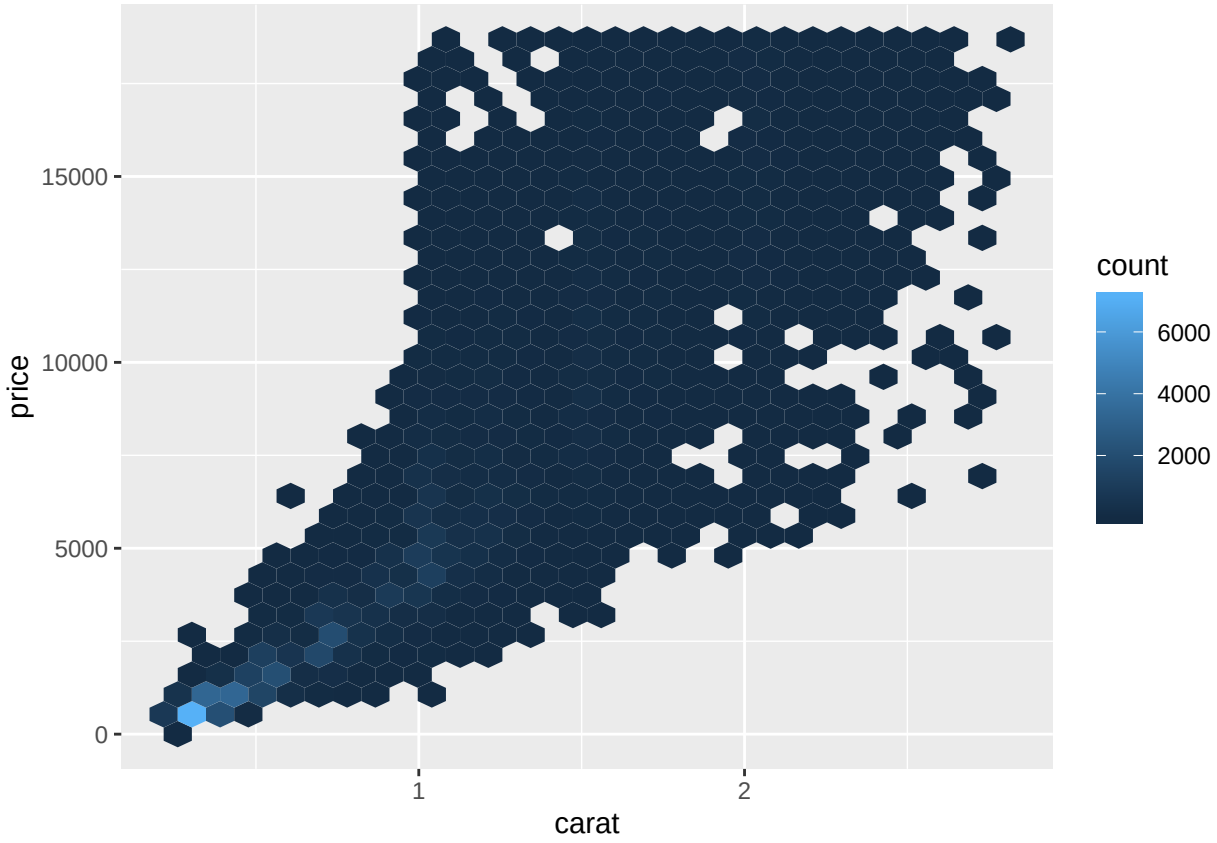
colour filling with the aesthetic

Use the geom_hex(), to create hexagonal bins

```
ggplot(data=smaller)+ geom_bin2d(mapping = aes(x=carat, y=price))
```

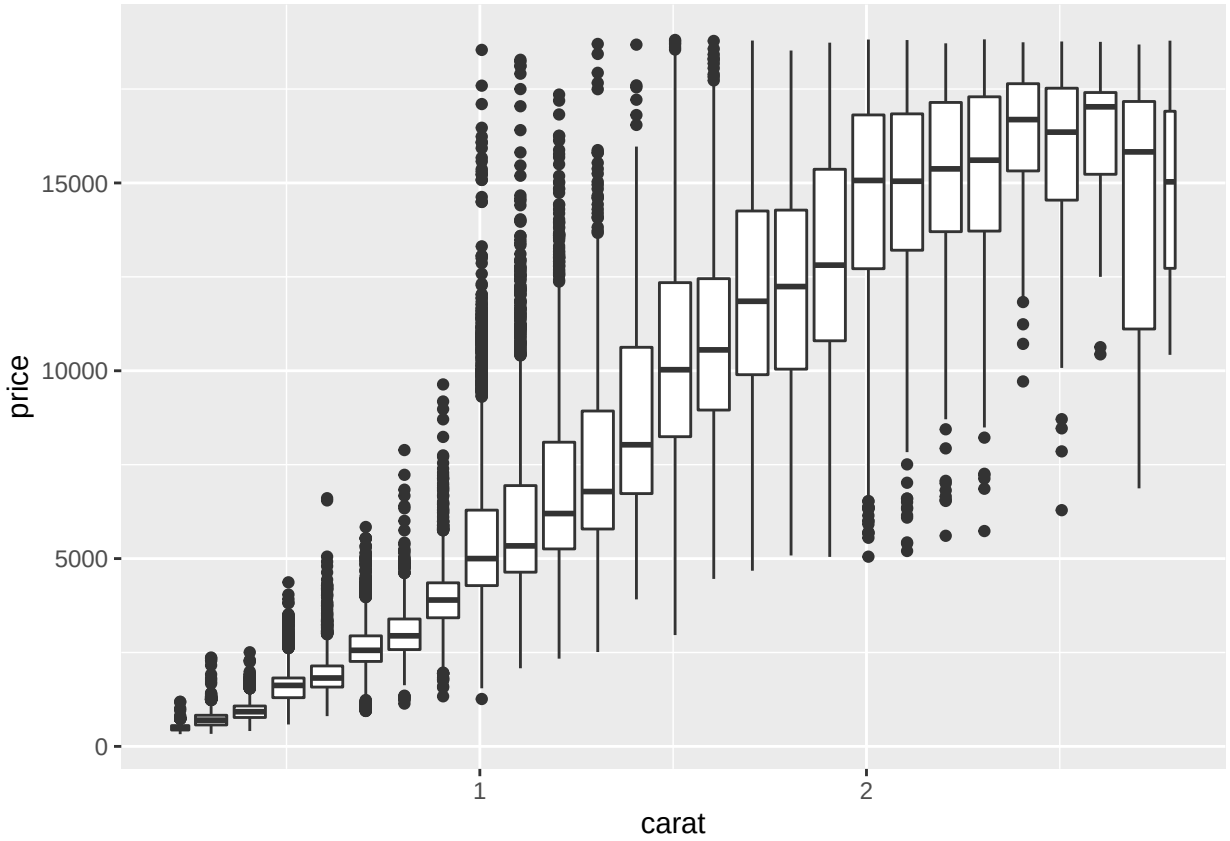



```
ggplot(data = smaller)+ geom_hex(mapping = aes(x= carat, y=price))
```



Use boxplot function,

```
ggplot(data=smaller, mapping = aes(x=carat, y=price)) + geom_boxplot(mapping = aes(group=cut_width(cara
```


price
15000
10000
5000
0
1
2
carat