Exploratory Data Analysis Brush Up

Orumwese Esosa Cyriaque

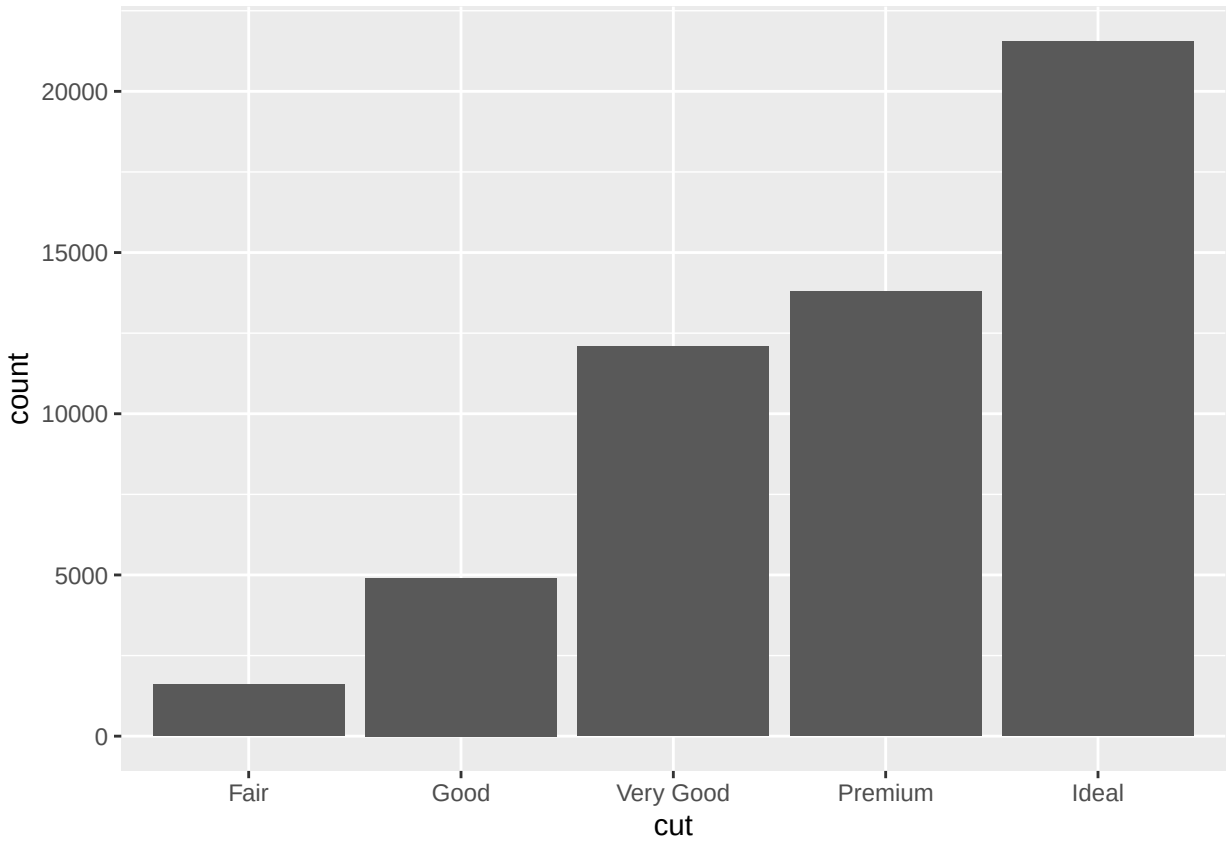
8/2/2021

```
library(tidyverse)
```

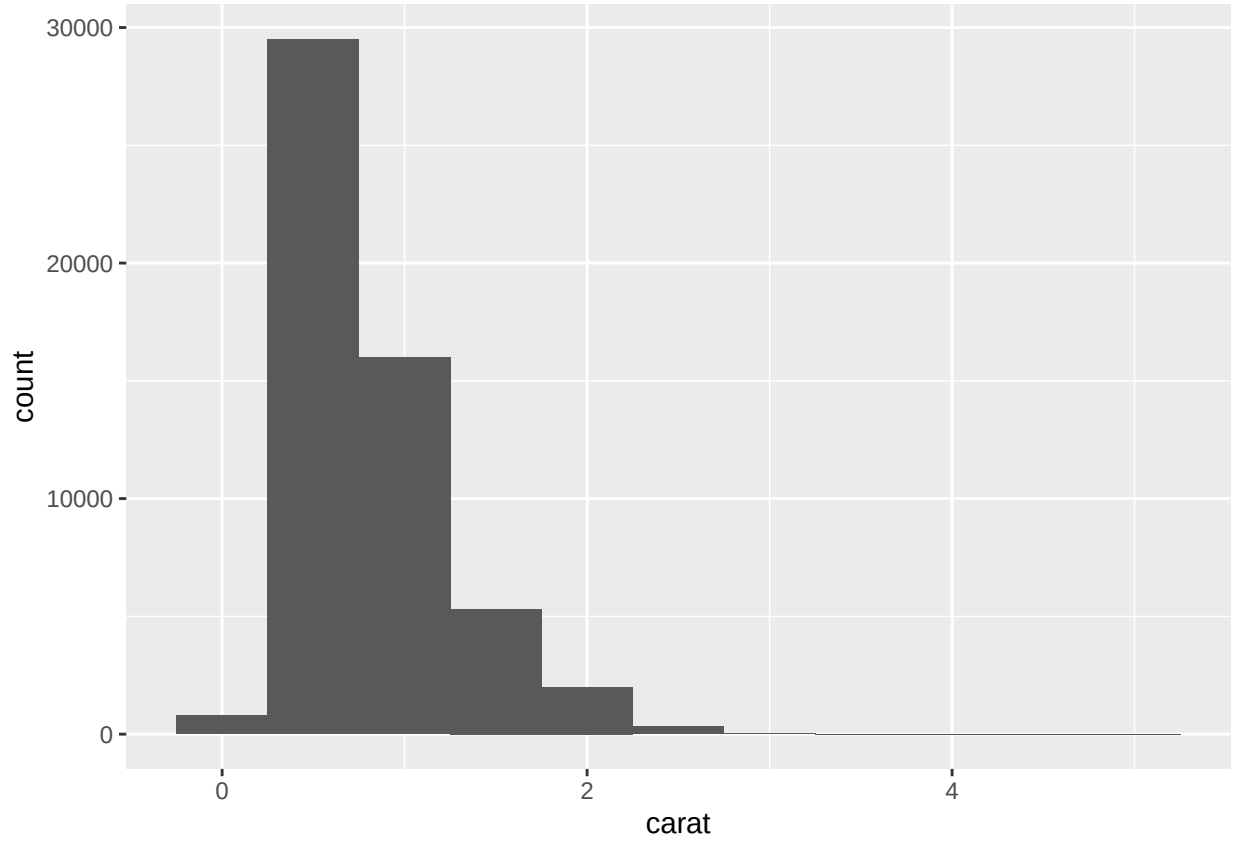
```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

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

V A R I A T I O N

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



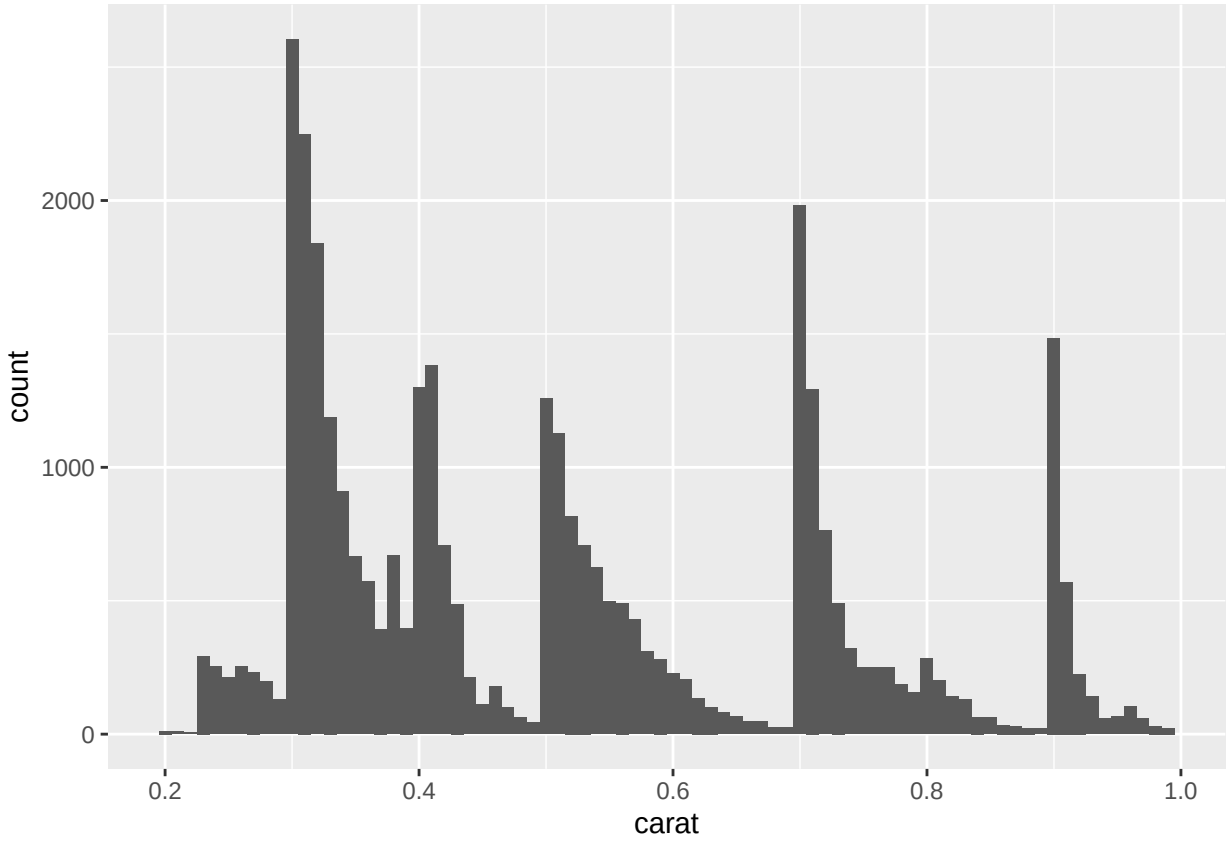

```
ggplot(diamonds) +
        geom_histogram(mapping = aes(x=carat), binwidth = 0.5)
```



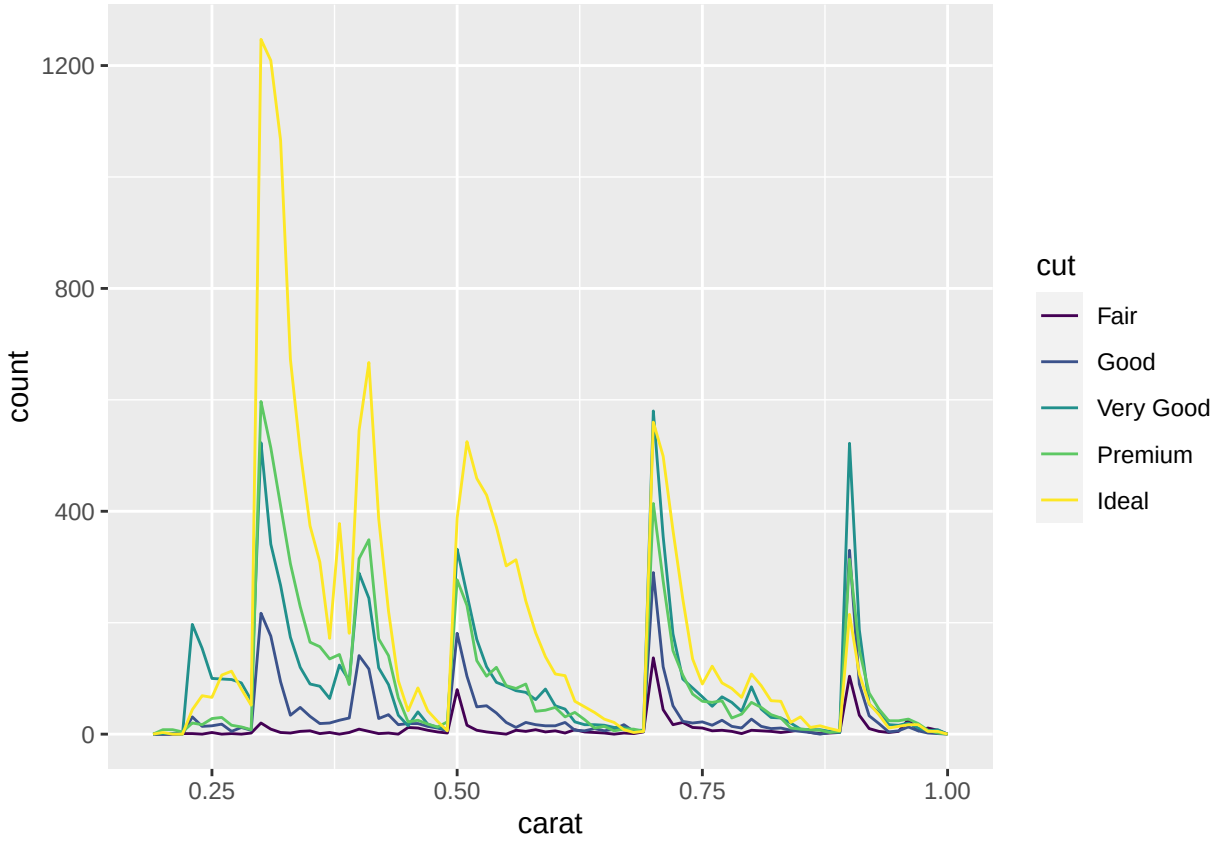

```
count(diamonds, cut_width(carat,0.5))
```

```
## # 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
```

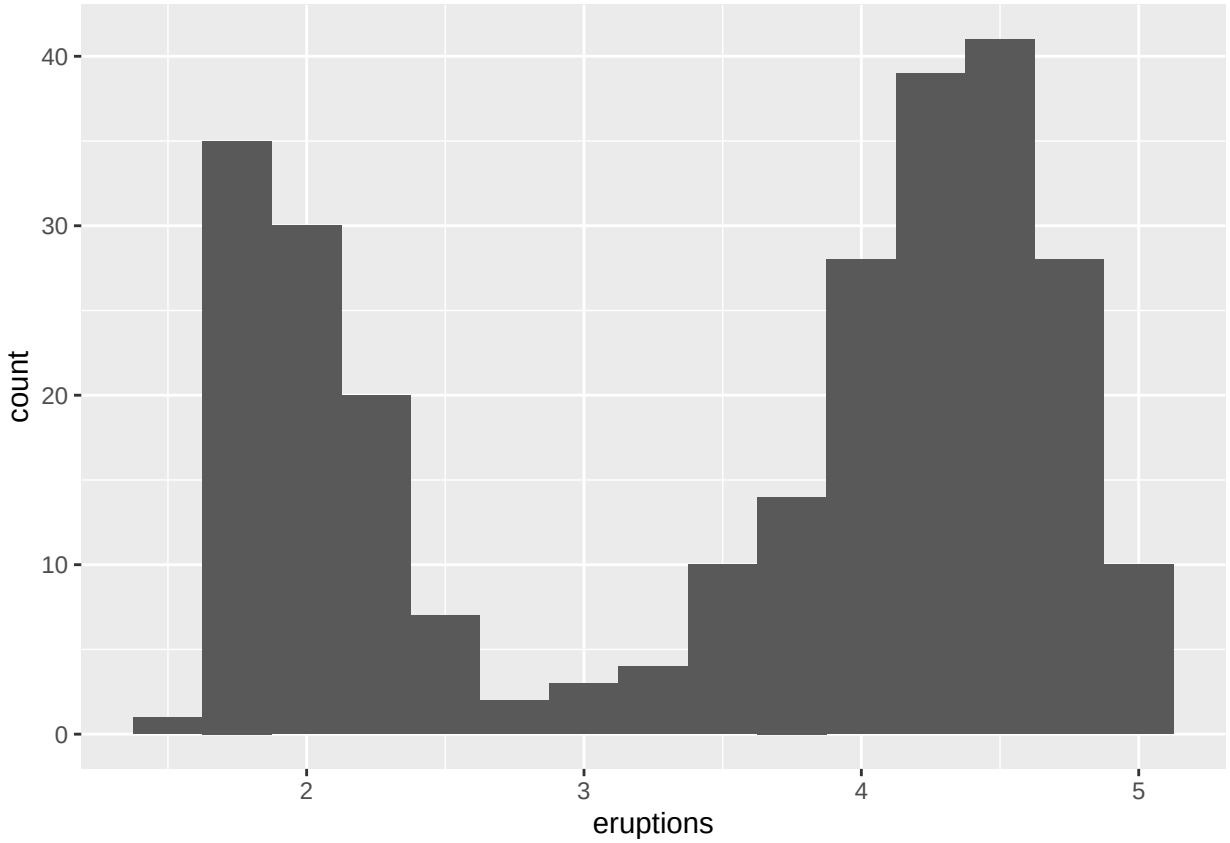
```
smaller <- diamonds %>%
        filter(carat<1)

ggplot(data = smaller, mapping = aes(x = carat)) +
        geom_histogram(binwidth = 0.01)
```




```
ggplot(data = smaller, mapping = aes(x = carat, color = cut)) +
       geom_freqpoly(binwidth = 0.01)
```

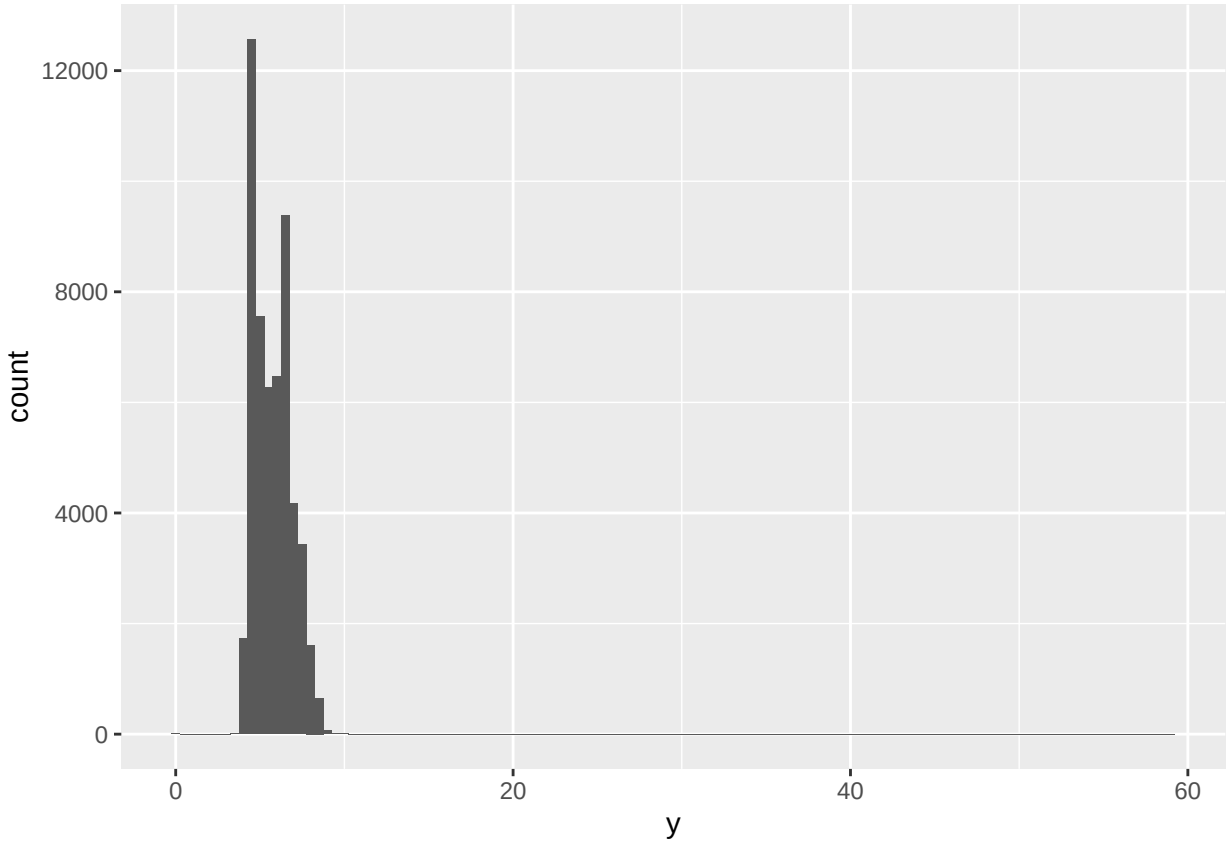



```
ggplot(data=faithful, mapping=aes(x=eruptions)) +
        geom_histogram(binwidth=0.25)
```

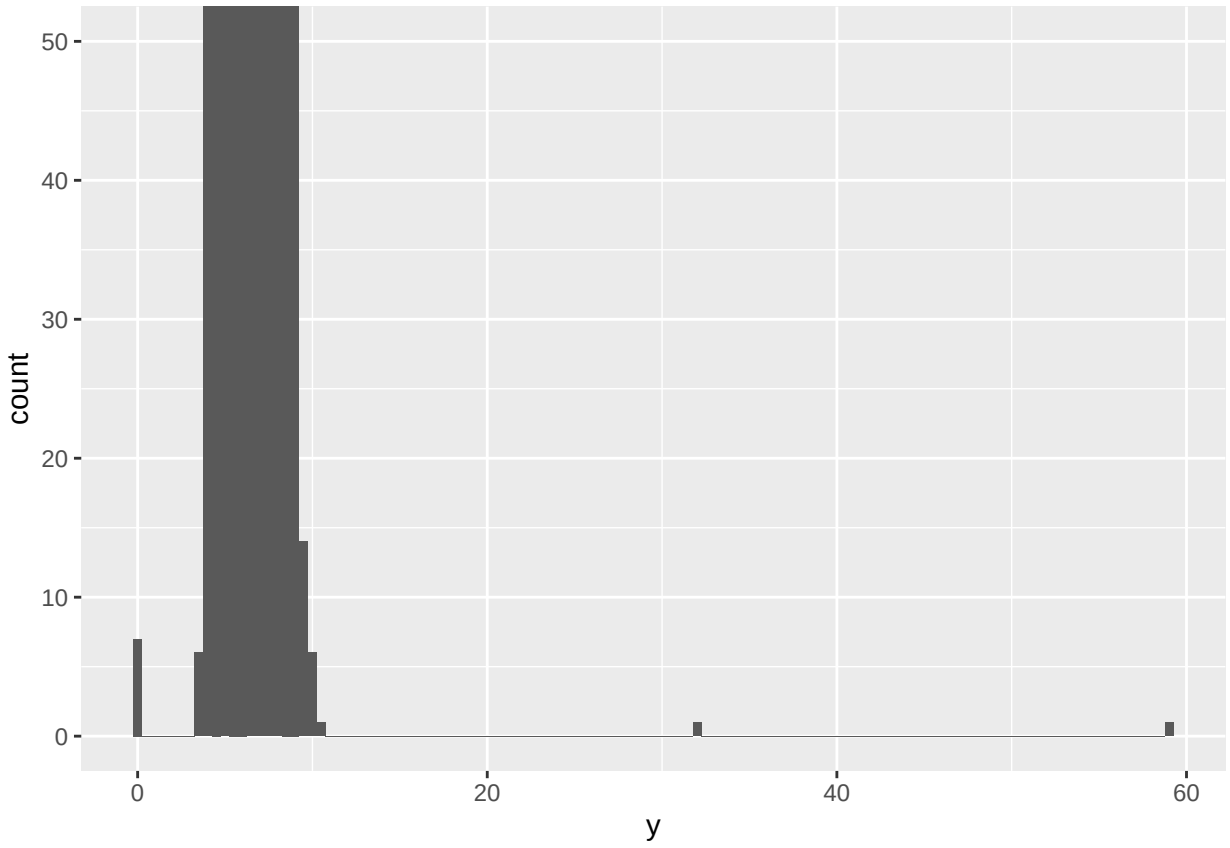



```
# clustered into two groups. Short eruptions(~2mins) and long eruptions(>5mins)
```

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




```
# Zooming in for a closer look
ggplot(diamonds) +
        geom_histogram(mapping = aes(x=y), binwidth = 0.5) +
        coord_cartesian(ylim = c(0,50))
```



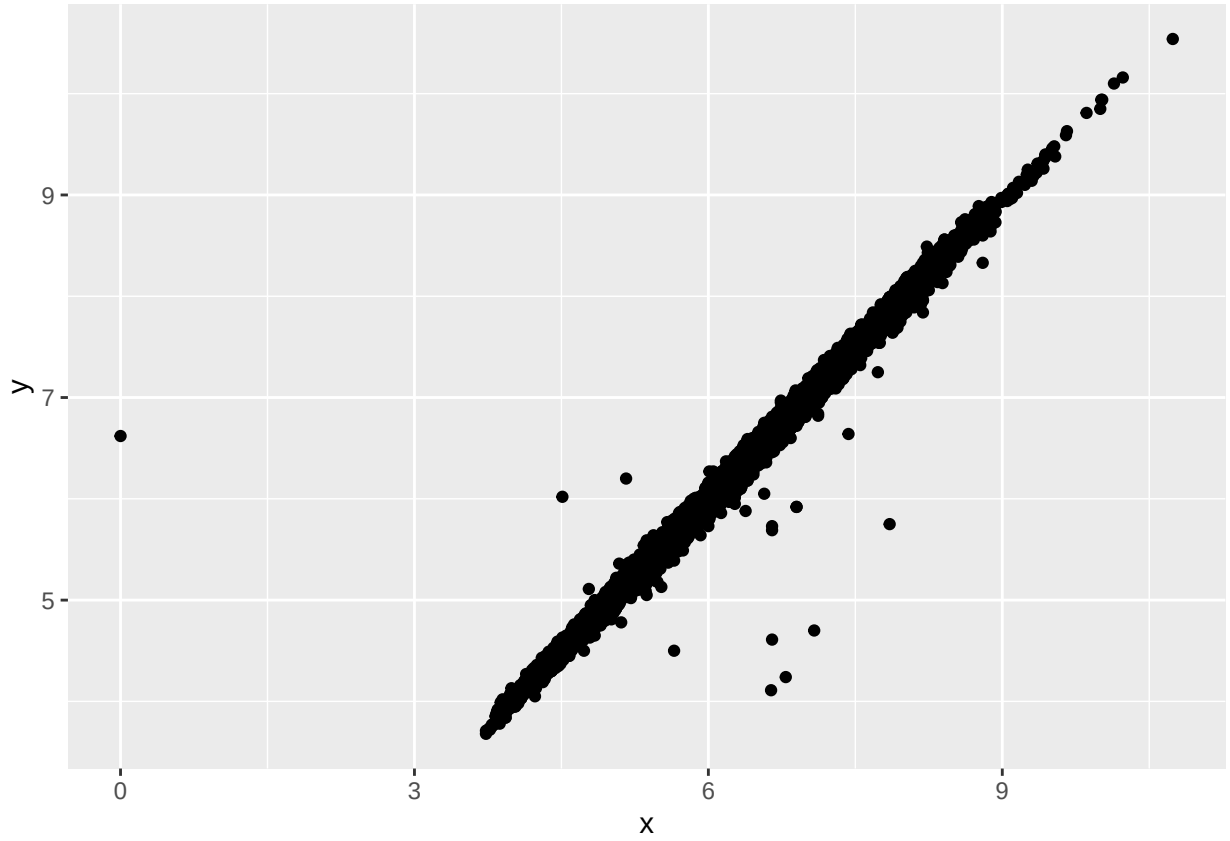

```
# It is noticed that there are 3 outliers: 0, ~30, and ~60.
```

```
unusual <- diamonds %>%
        filter(y<3|y>20) %>%
        arrange(y)
unusual
```

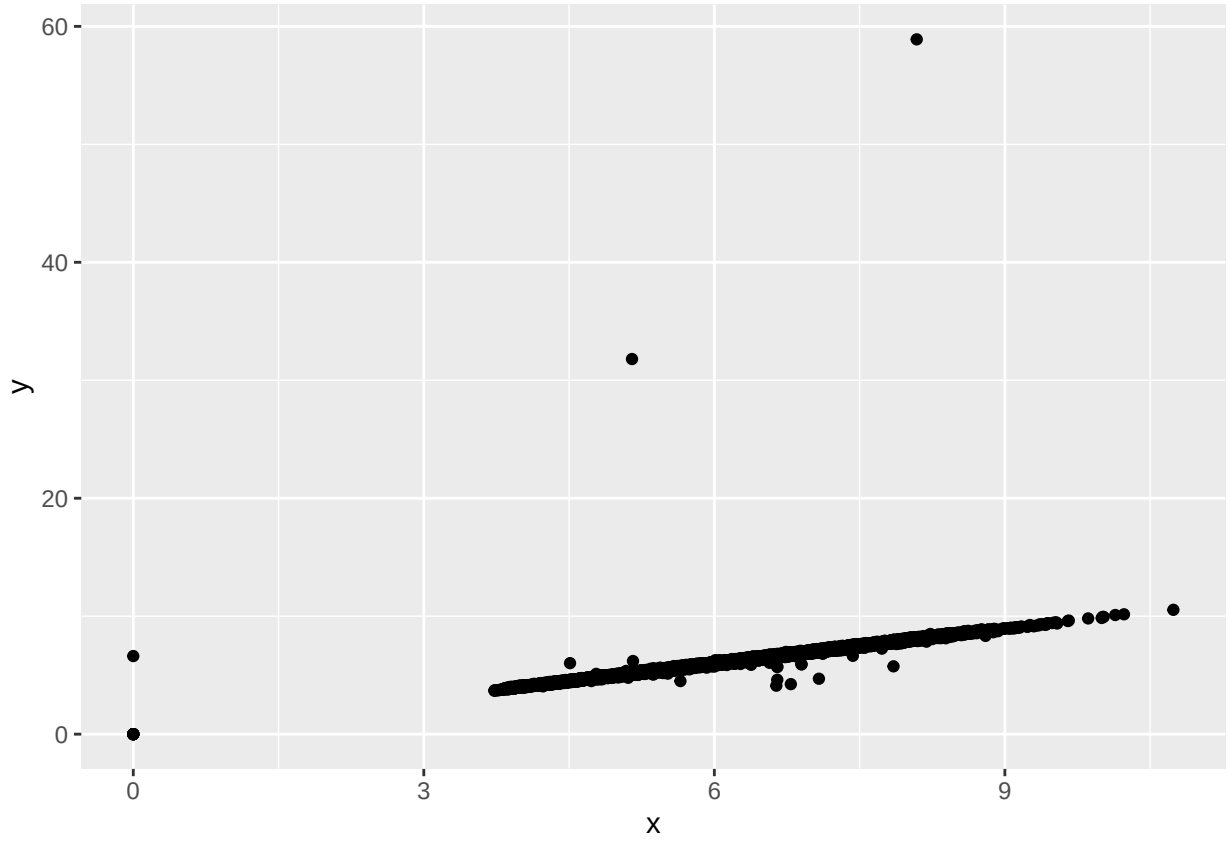
```
## # A tibble: 9 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  1    Very Good H     VS2      63.3    53  5139  0      0    0
## 2  1.14 Fair      G     VS1      57.5    67  6381  0      0    0
## 3  1.56 Ideal     G     VS2      62.2    54 12800  0      0    0
## 4  1.2  Premium   D     VVS1     62.1    59 15686  0      0    0
## 5  2.25 Premium   H     SI2      62.8    59 18034  0      0    0
## 6  0.71 Good      F     SI2      64.1    60  2130  0      0    0
## 7  0.71 Good      F     SI2      64.1    60  2130  0      0    0
## 8  0.51 Ideal     E     VS1      61.8    55  2075  5.15  31.8  5.12
## 9  2    Premium   H     SI2      58.9    57 12210  8.09  58.9  8.06
```

```
# Removes outliers by making them NA 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).
```

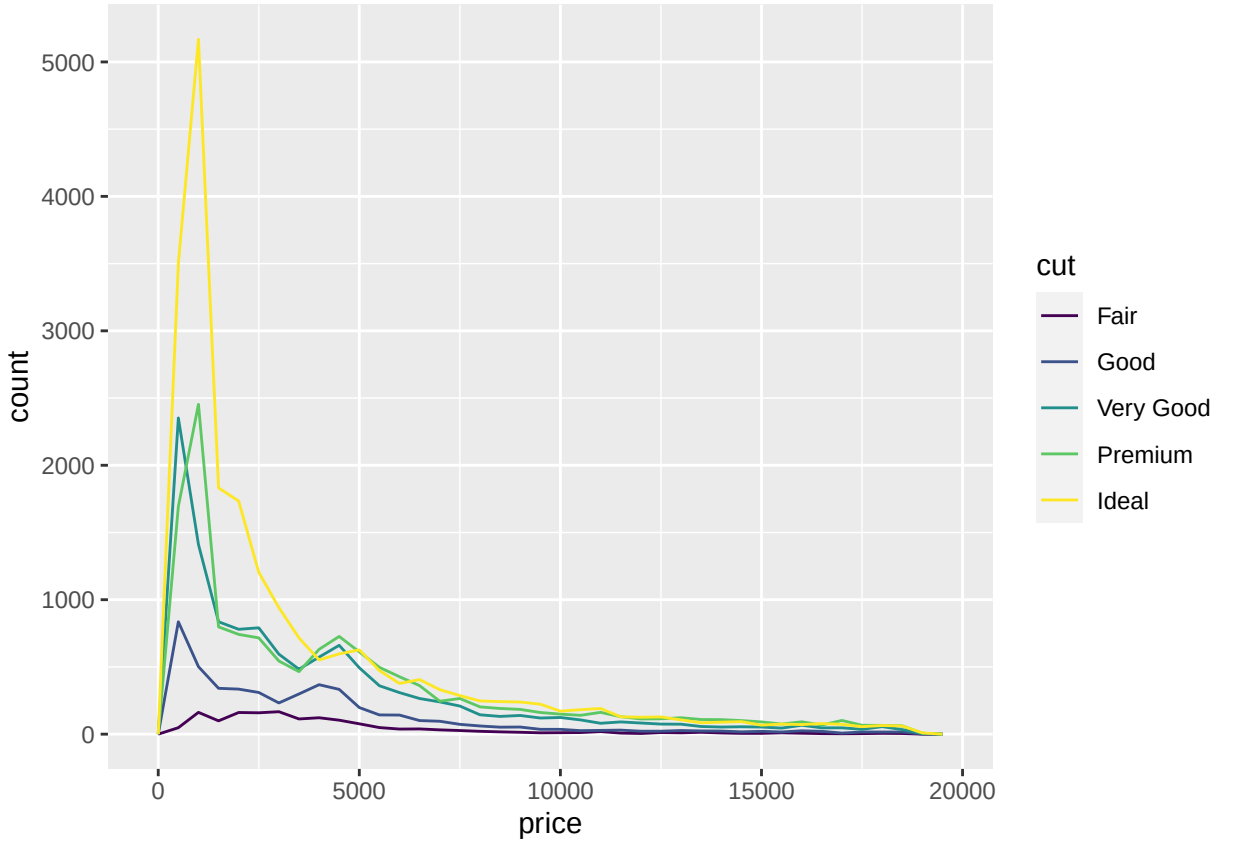



```
ggplot(data=diamonds, mapping=aes(x=x,y=y)) +
        geom_point()
```

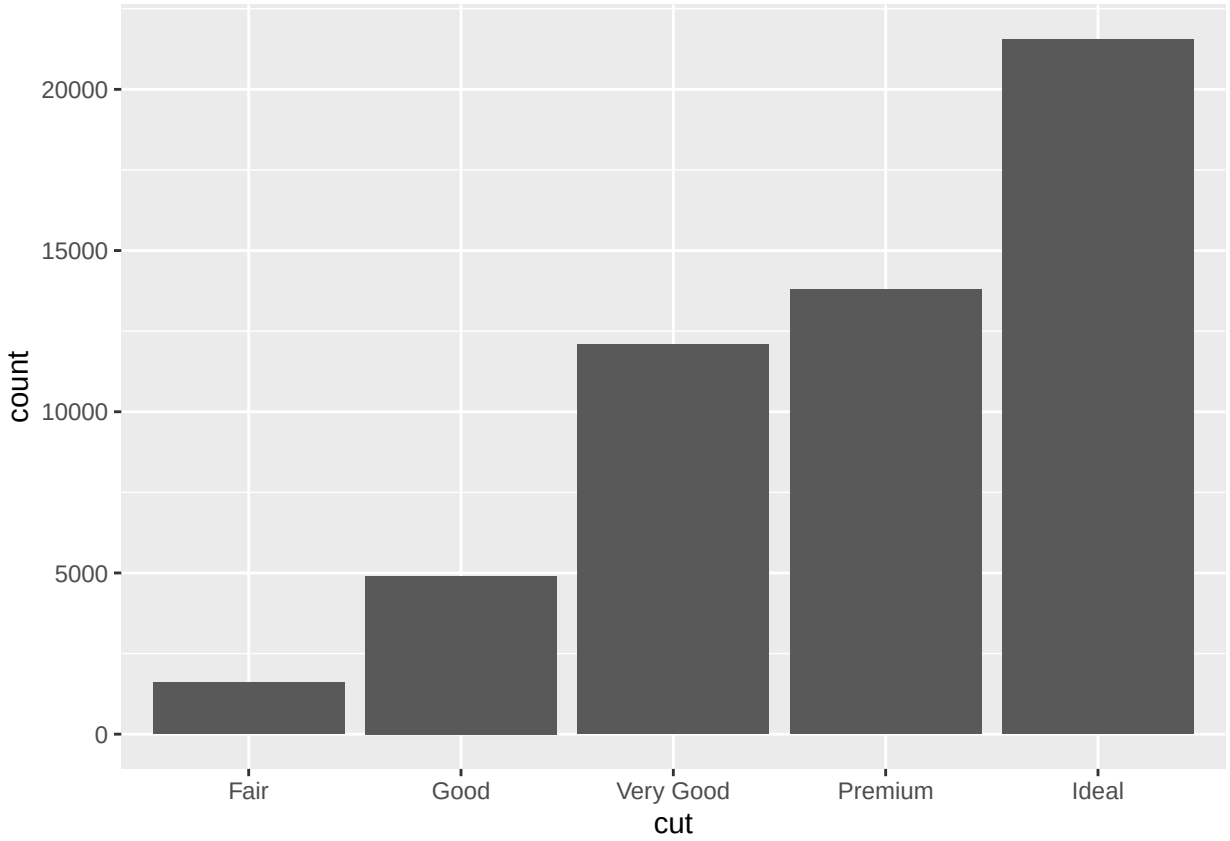


COVARIATION

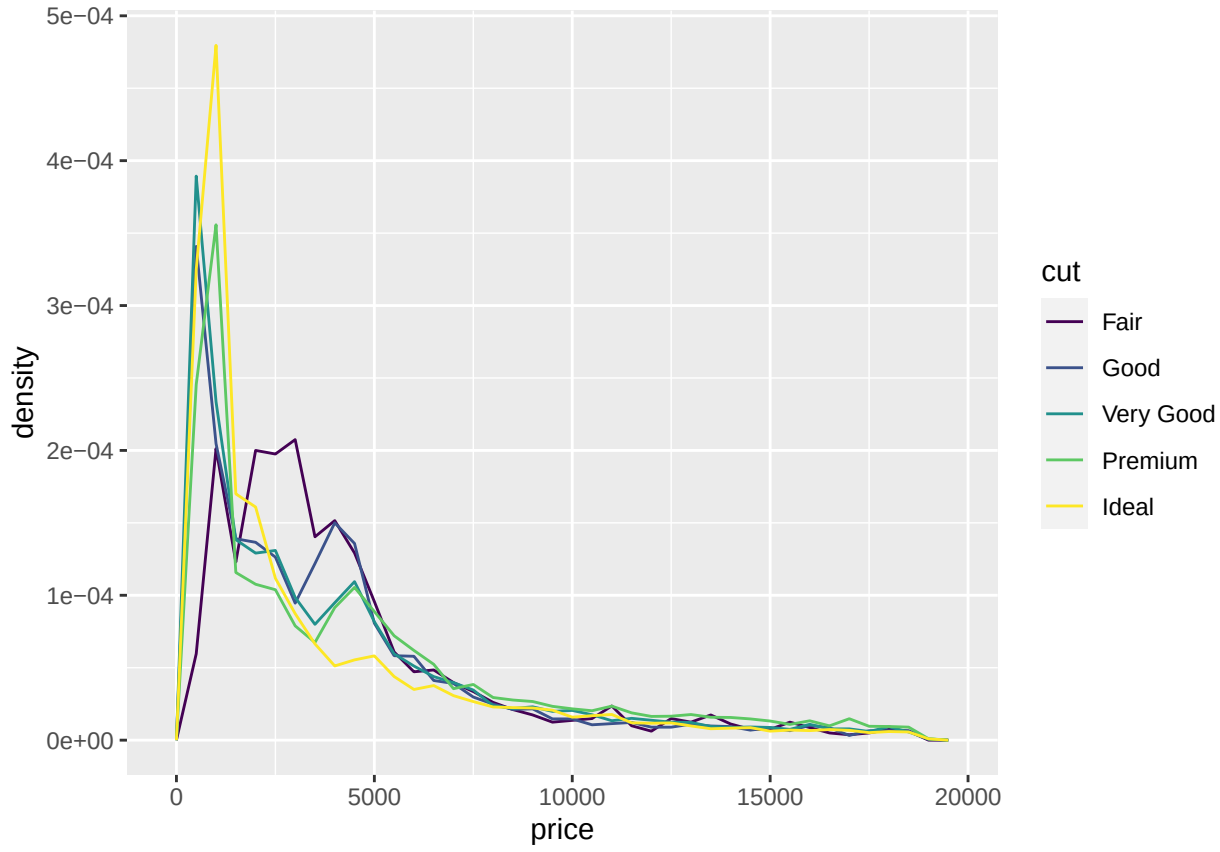
```
# Displaying the uselessness of the def appearance of geom_freqpoly()
# It makes it hard to see the differences in shape to the data compared being different

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




```
# It's hard to see the difference in distribution because the overall counts differ so much
ggplot(diamonds) + geom_bar(mapping=aes(x=cut))
```

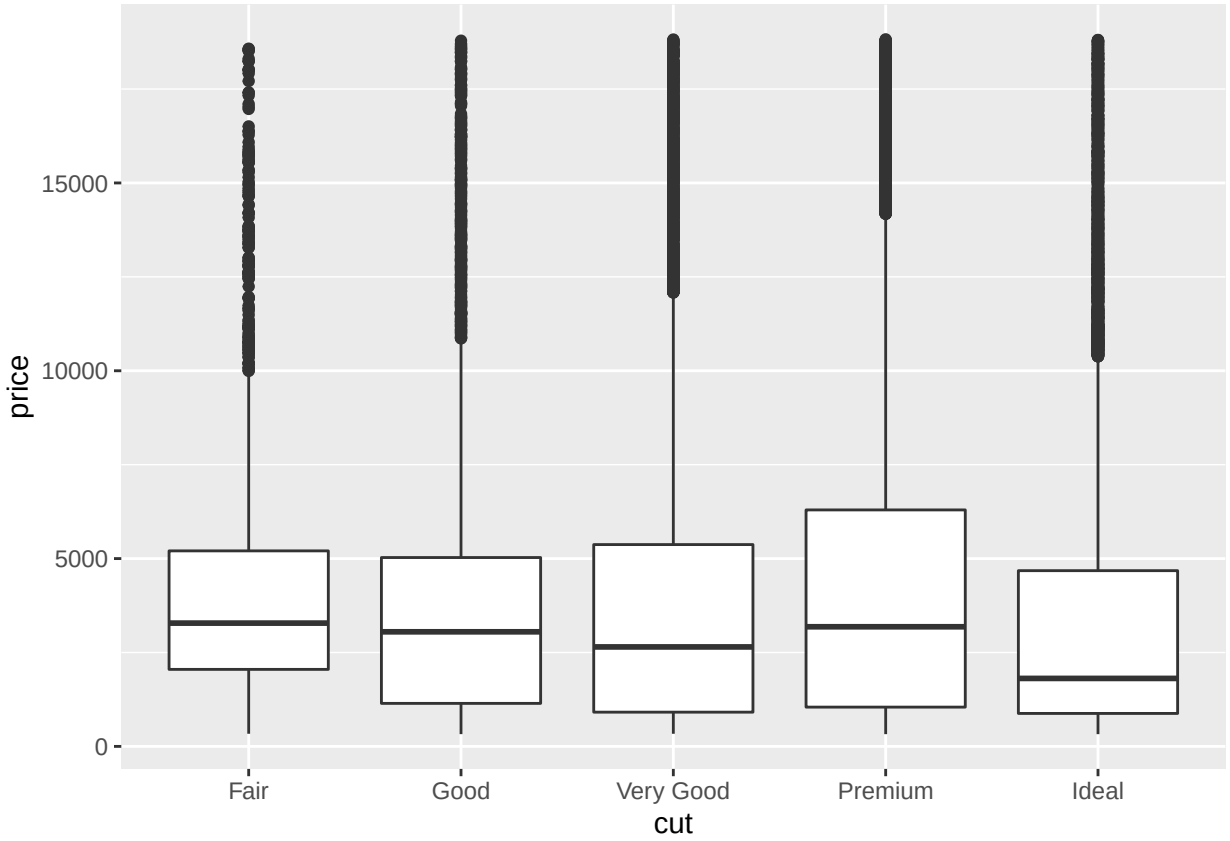



```
# This issue can be solved via standardization and this is achieved by replacing count (on y axis) with
# Density is count standardized so that the area under each frequency polygon is 1.

ggplot(diamonds, aes(x=price, y=..density..)) +
       geom_freqpoly(aes(color=cut), binwidth=500)
```

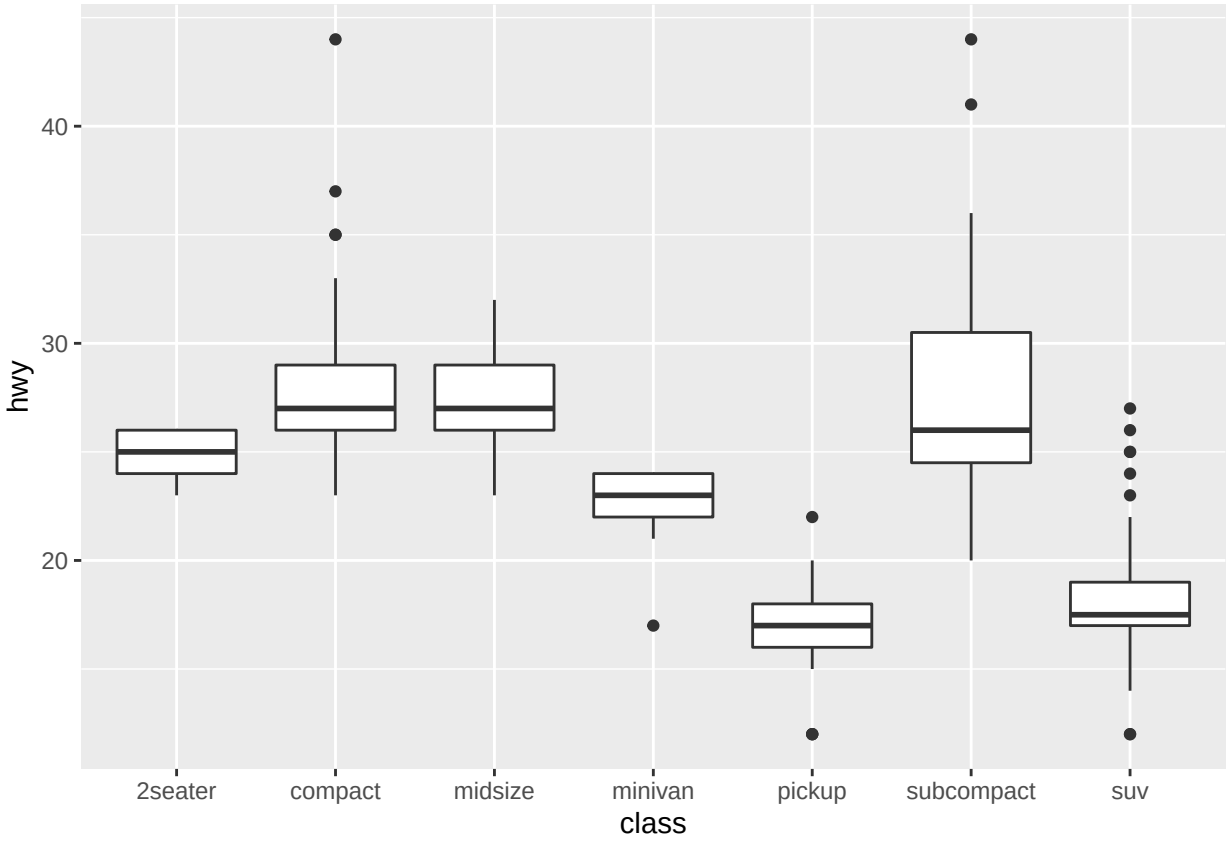



```
# Frequency Plots are a little hard to interprete
```

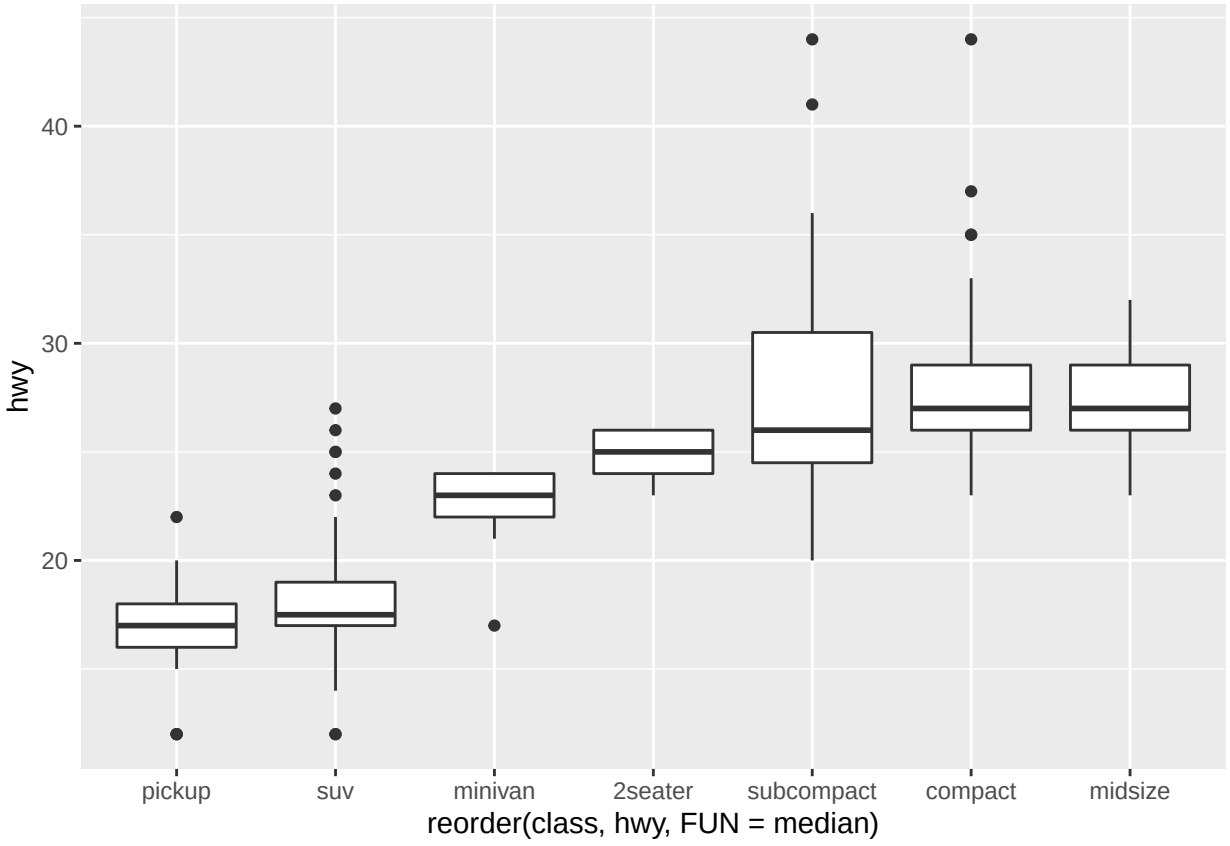
```
ggplot(diamonds, aes(x=cut, y=price)) +
        geom_boxplot()
```



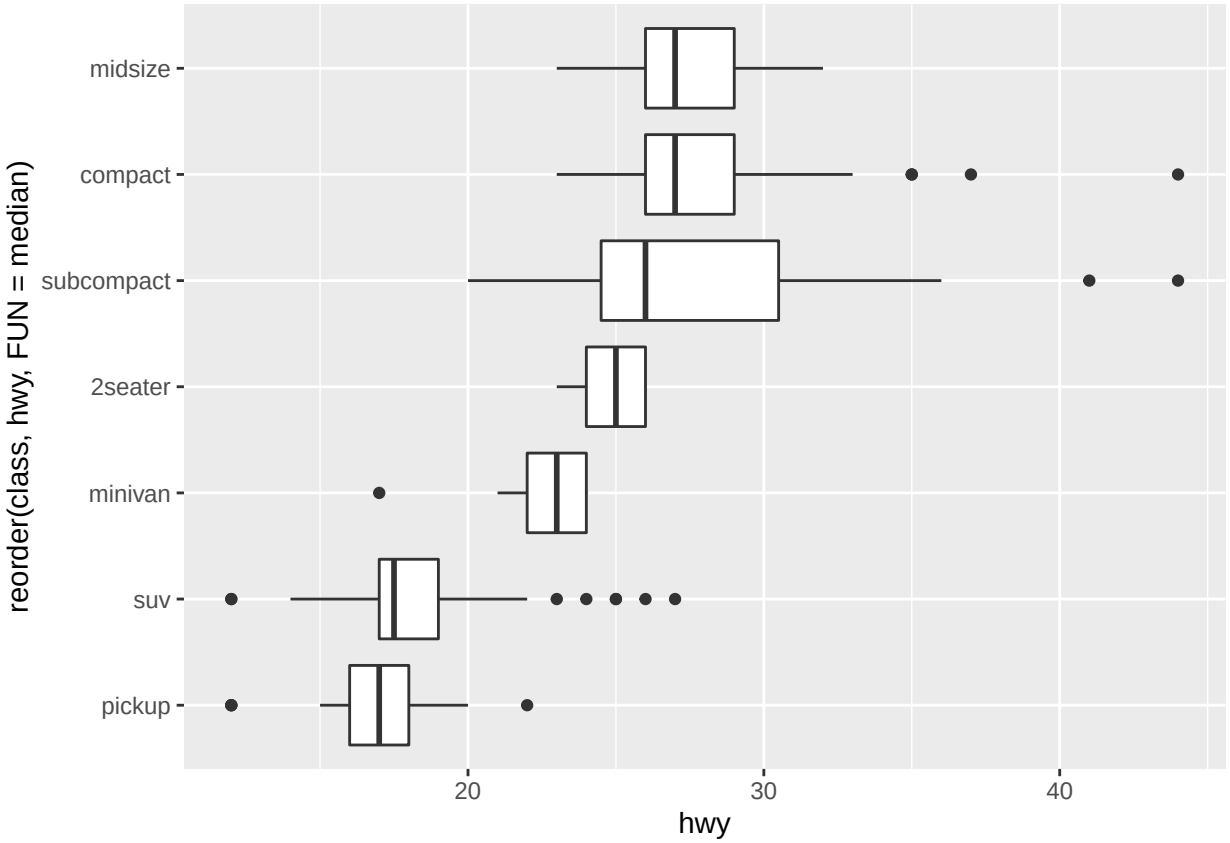

```
ggplot(mpg, aes(x=class, y=hwy)) +
       geom_boxplot()
```




```
# Reordering class based on median value of hwy

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

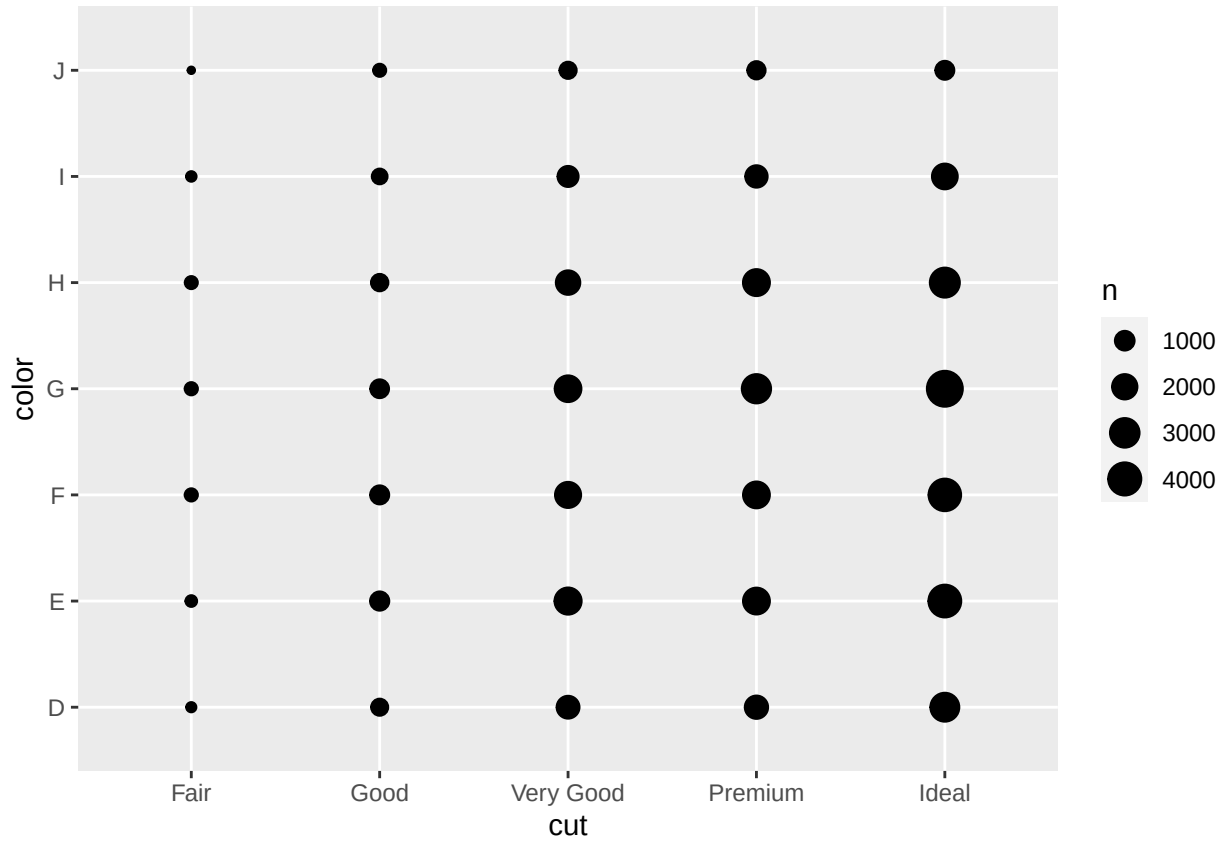



```
# flipping plot
ggplot(mpg) +
        geom_boxplot(aes(x=reorder(class, hwy, FUN=median), y=hwy)) +
        coord_flip()
```



Two Categorical Variables

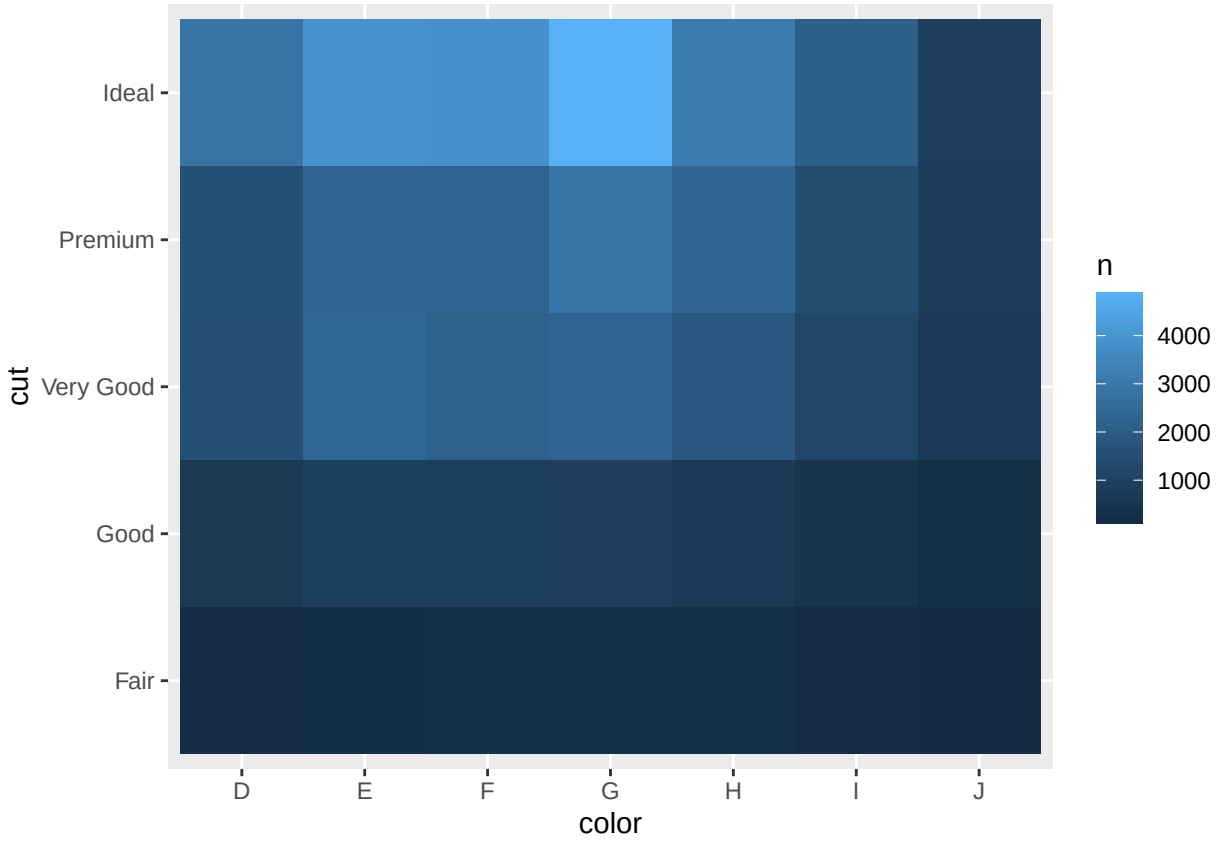
```
# Visualizing the covariation between two categorical variables
# Use geom_count()

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




```
# Another approach is to compute the count with dplyr and then visualize with geom_tile()

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

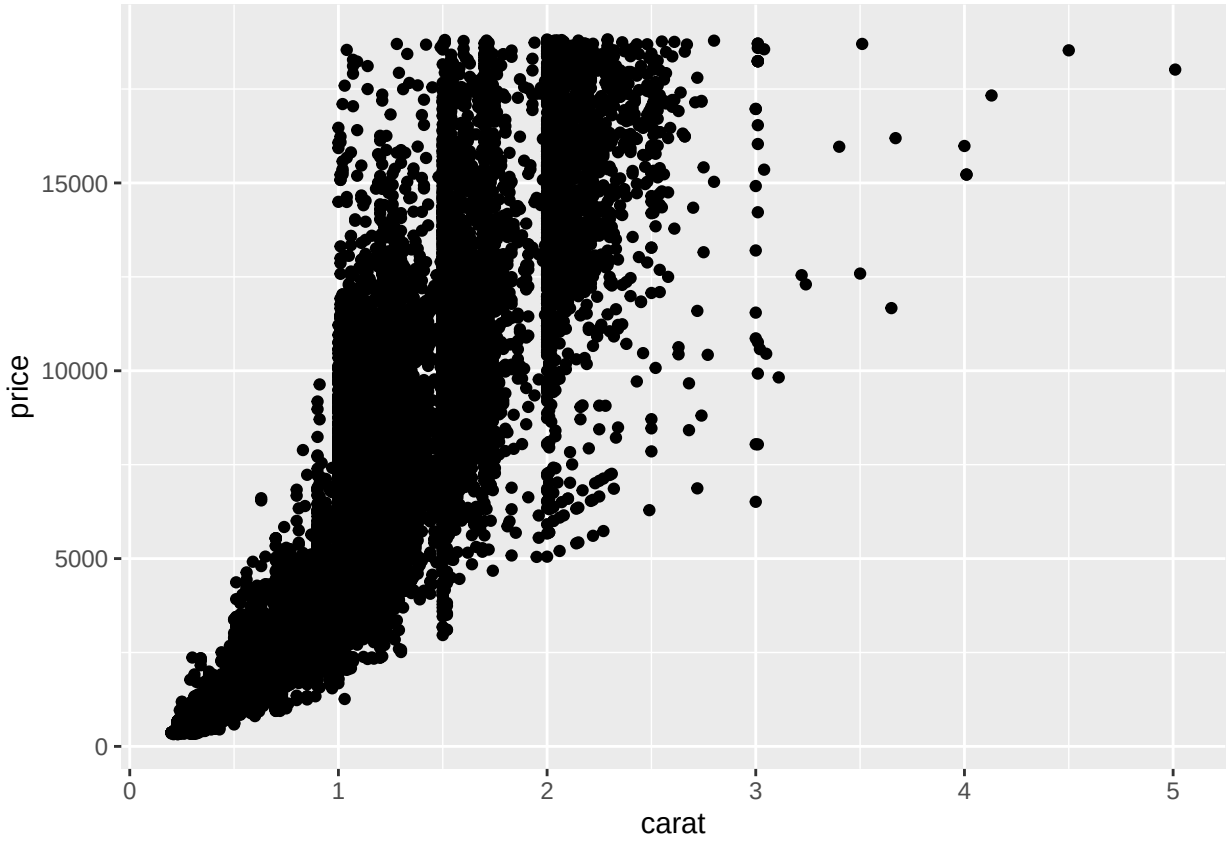
```
## # A tibble: 35 x 3
##    color cut           n
##    <ord> <ord>     <int>
##  1 D     Fair        163
##  2 D     Good        662
##  3 D     Very Good  1513
##  4 D     Premium    1603
##  5 D     Ideal      2834
##  6 E     Fair        224
##  7 E     Good        933
##  8 E     Very Good 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

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

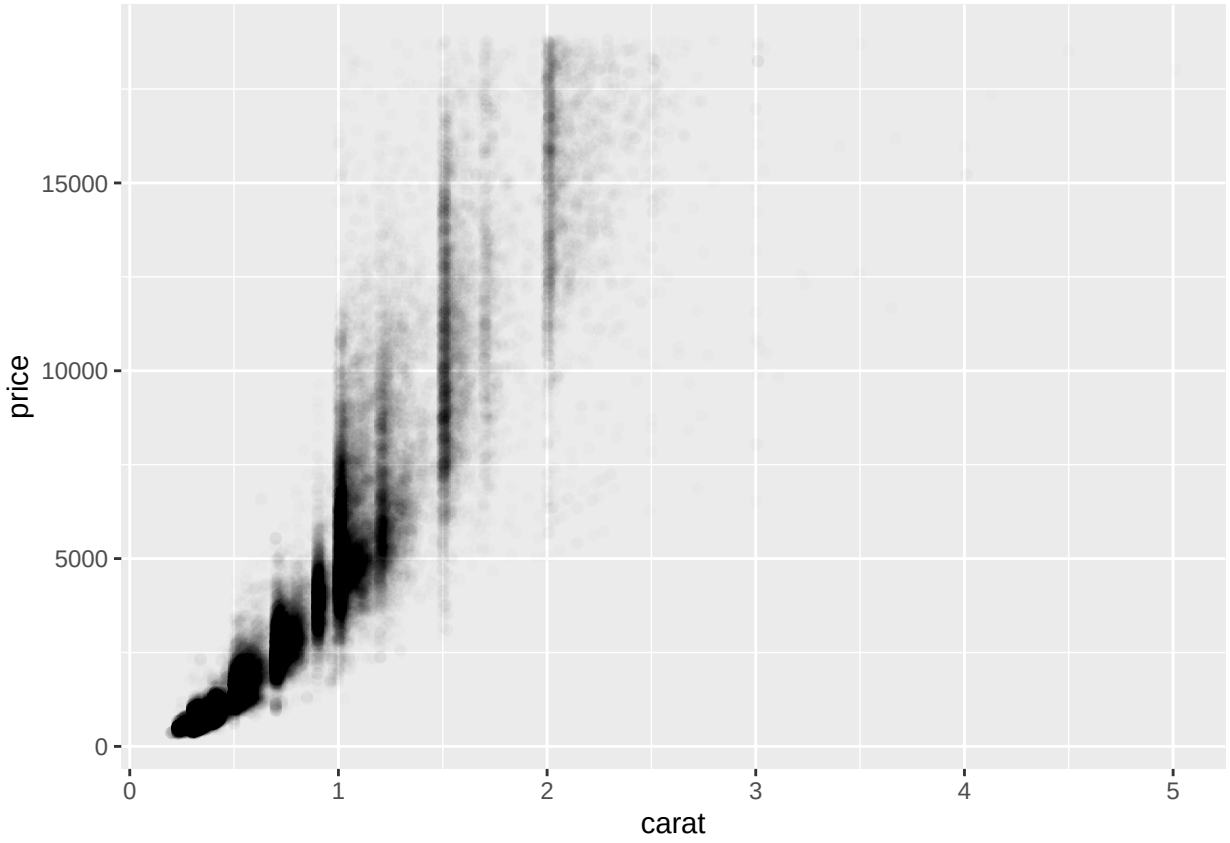


Two Continuous Variables

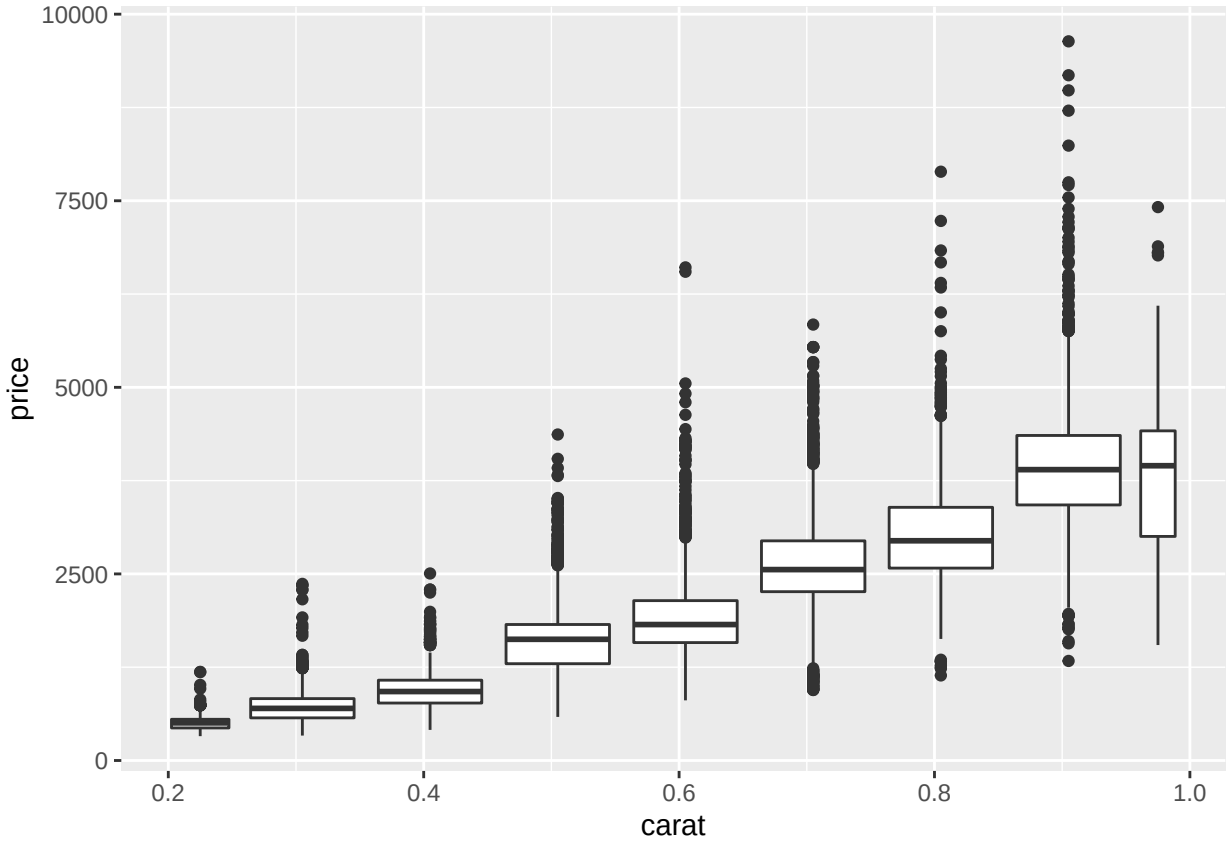
```
# Visualization using a scatterplot.

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




```
# Enhancing readability by adding transparency
ggplot(diamonds) +
       geom_point(aes(y=price, x=carat), alpha = 0.01)
```




```
# Don't get this sha
ggplot(smaller, aes(x=carat, y=price)) +
       geom_boxplot(aes(group=cut_width(carat, 0.1)))
```


10000
7500
5000
2500
0
price
0.2
0.4
0.6
0.8
1.0
carat