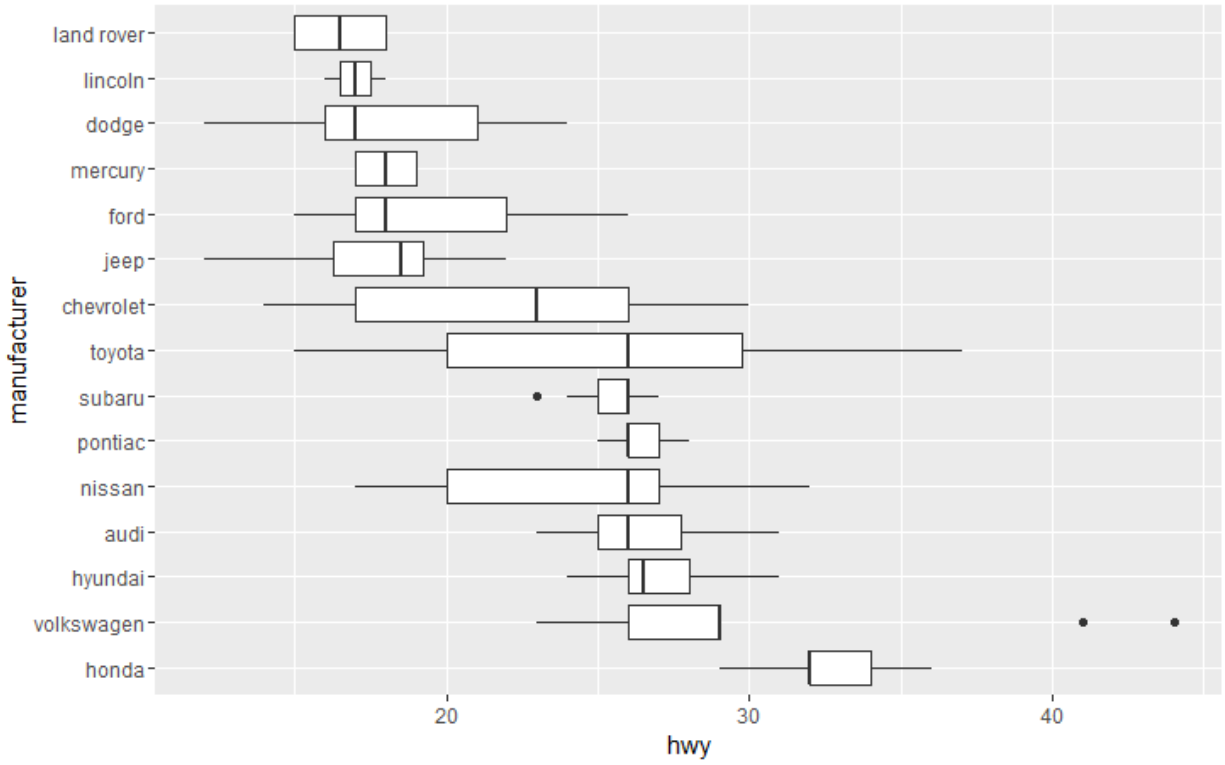
HW2

Author: Daniel Rozen, drozen3

```
library(ggplot2)

# 1. Using the mpg data, describe the relationship between highway mpg and
# car manufacturer. Describe which companies produce the most and least
# fuel efficient cars, and display a graph supporting your conclusion.

data(mpg)

# plot multiple box plots sorted by median hwy mpg vs. manufacturer.
ggplot(mpg, aes(reorder(manufacturer, -hwy, median), hwy)) +
  geom_boxplot() + # create boxplot
  coord_flip() + # make boxplots sideways
  scale_x_discrete("manufacturer")
```

With the code, we first plot the following side-by-side boxplots.



From the plot, we see that Honda clearly has the highest highway mp3. Therefore it produces the most fuel efficient cars.

I would say based on median hwy mpg, that the following cars also produce decently fuel efficient cars, with medians close to around 26-27 mpg: Toyota, Subaru, Pontiac, Nissan, Audi, Hyundai, and Volkswagen.

Based on median hwy mpg, the following cars produce the least fuel efficient cars:

Land over, Lincoln, Dodge, Mercury, Ford, and Jeep. Honda has disjoint values from the above low fuel efficiency cars.

 # 2. Using the mpg data, explore the three-way relationship between highway

 # mpg, city mpg, and model class. What are your observations? Display a

 # graph supporting these observations.

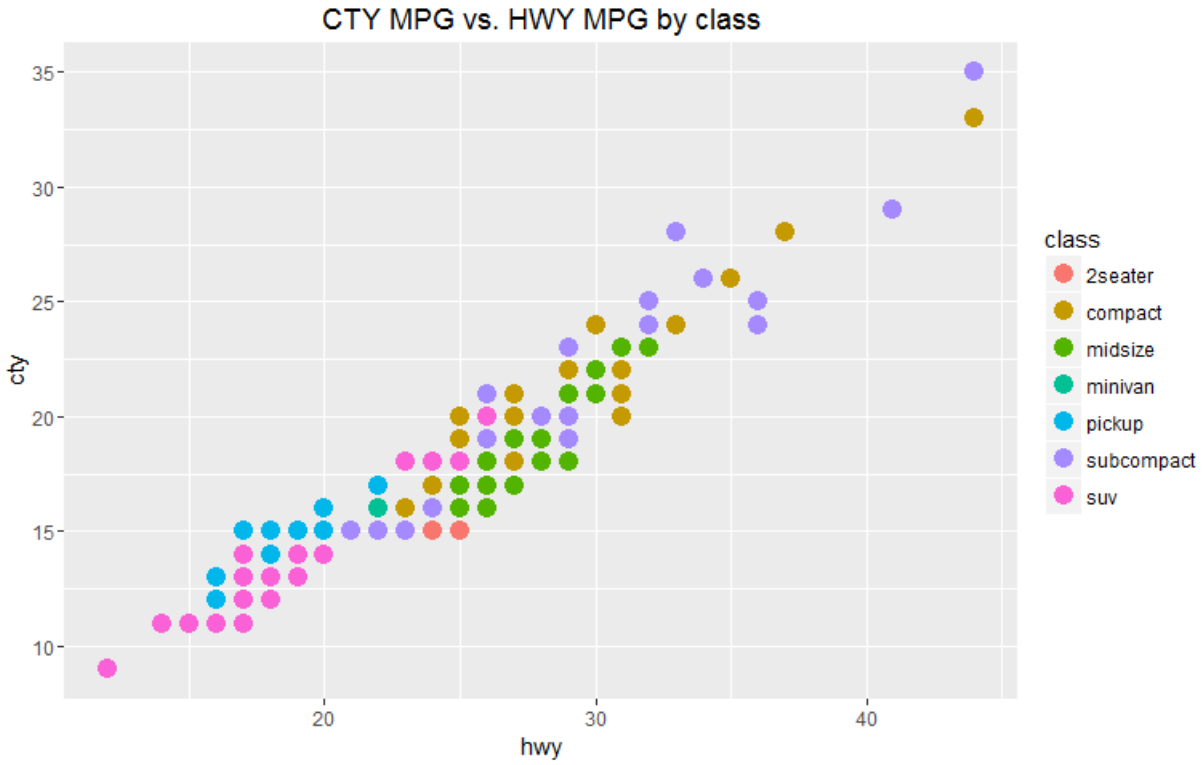
```
# Plot a scatter plot with markers of different colors for class

qplot(x = hwy,

    y = cty,
```

data = mpg,

color = class,

size=I(4),

main = "CTY MPG vs. HWY MPG by class")



#multiple panel plot with smoothed line

qplot(x = hwy,

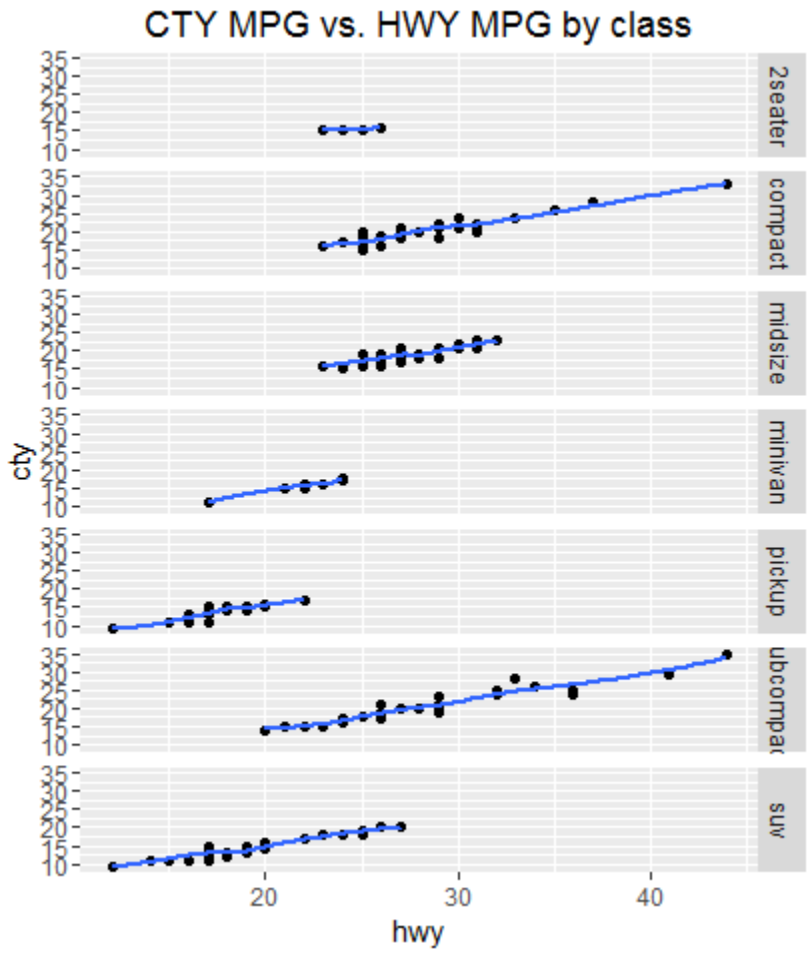
y = cty,

facets = class~.,

data = mpg,

main = "CTY MPG vs. HWY MPG by class") +

stat_smooth(se = FALSE)



From the above 2plots , we can see that cty mpg and hwy mpg are all positively linearly correlated since the scatter plots demonstrates a line with a positive slope.

The classes with the highest cty and hwy mpgs are 2seater, compact, and subcompact. Midsize is in the middle. Pickups and Suvs have the lowest. Minivan is also low.

Subcompact and compact have a large spread, whereas midsize and suv, and pickup is more concentrated.

 # 3. What are the pros and cons of using a histogram vs a box plot? Which one

 # will you prefer for what purpose?

Histograms:

Pros:

A correct bin width can help you easily draw conclusions from data by eliminating unnecessary noise while maintaining useful information from useful data aggregation. Eg. The 2 different eruption times on pg 7.

Cons:

Need to determine the correct bin width. With, too narrow of a bin width, the information will be present but it will be hard to draw conclusions from the histogram, since the histogram becomes equivalent to a sorted list of data values. If the bins are too wide, you will lose information due to overly aggressive smoothing.

Preferred Use:

Histograms are good for graphing one dimensional numeric data and looking for typical trends. Eg. The 2 different eruption times on pg 7.

Histograms are useful for summarizing numeric data since they show the rough distribution of values.

Box Plot

Pros:

Easy to read the median and IQR.

Eliminates outliers since they are separated from the box and whiskers.

Convenient to plot several box plots side by side in order to compare data corresponding to different values of a factor variable.

Cons:

does not convey the multimodal nature of a distribution that a histogram would show.

Preferred Use:

Box Plots are often used as an alternative or in conjunction with box-plots. Therefore Box Plots are also useful for are good for graphing one dimensional numeric data and looking for typical trends. And are also useful for summarizing numeric data since they show the rough distribution of values.

 # 4. Generate two sets of N random points using the function runif and display

 # a corresponding scatter plot.

N = 100

x = runif(N)

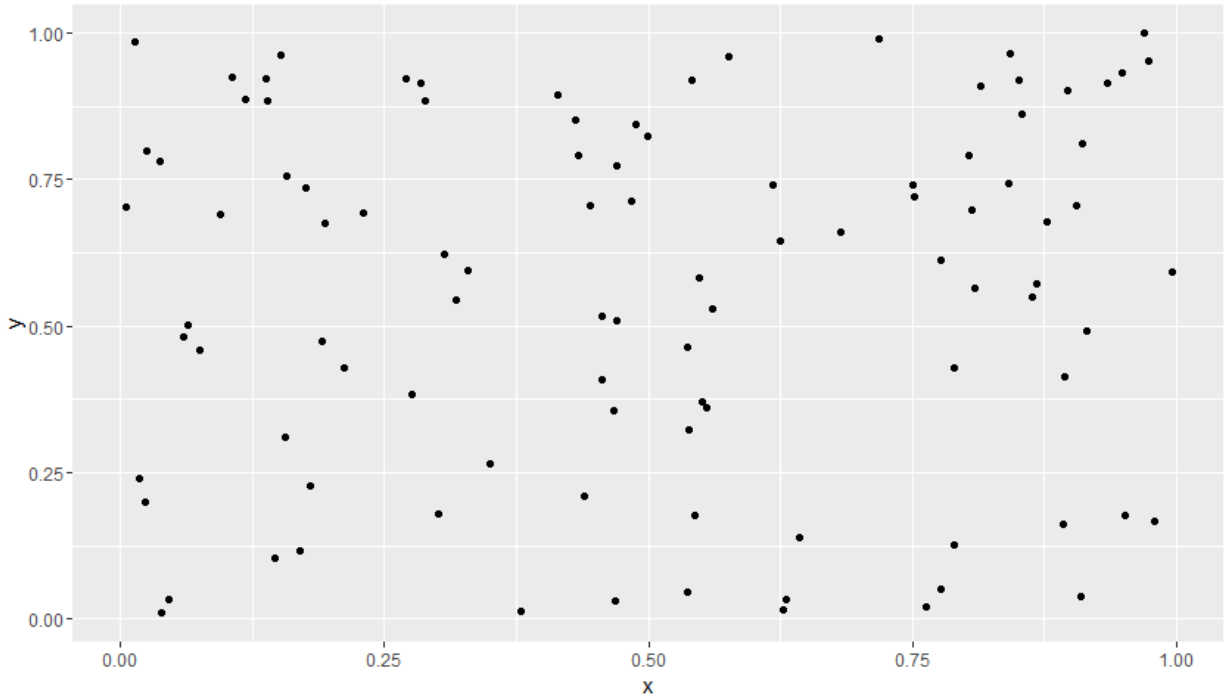
y = runif(N)

DF = data.frame(x = x, y = y) # create dataframe

#qplot(x=x, y=y, data=DF)

ggplot(DF, aes(x = x, y = y)) + geom_point() # generate scatter plot

Here is the resulting plot:



How do these values scale with increasing N?

library(ggplot2)

psSizes = c()

pdfSizes = c()

jpegSizes = c()

pngSizes = c()

```
# loop for different N sizes
sizes= c(1,10,50,100,500,1000,2500,5000,7500,10000, 15000, 25000, 35000,50000)

for (N in sizes) {
  x = runif(N)
  y = runif(N)
  DF = data.frame(x = x, y = y) # create dataframe
  myplot = ggplot(DF, aes(x = x, y = y)) + geom_point() # generate scatter plot
 # save files
  ggsave(file="myPlot.ps")
  ggsave(file="myPlot.pdf")
  ggsave(file="myPlot.jpeg")
  ggsave(file="myPlot.png")

  psSizes = c(psSizes, file.size("myPlot.ps")) # record file sizes
  pdfSizes = c(pdfSizes, file.size("myPlot.pdf")) # save file size
  jpegSizes = c(jpegSizes, file.size("myPlot.jpeg")) # save file size
  pngSizes = c(pngSizes, file.size("myPlot.png")) # save file size

}


#create chart

#print dataframe

df=data.frame(N=sizes, psSizes=psSizes, pdfSizes=pdfSizes, jpegSizes=jpegSizes, pngSizes=pngSizes)
df
```

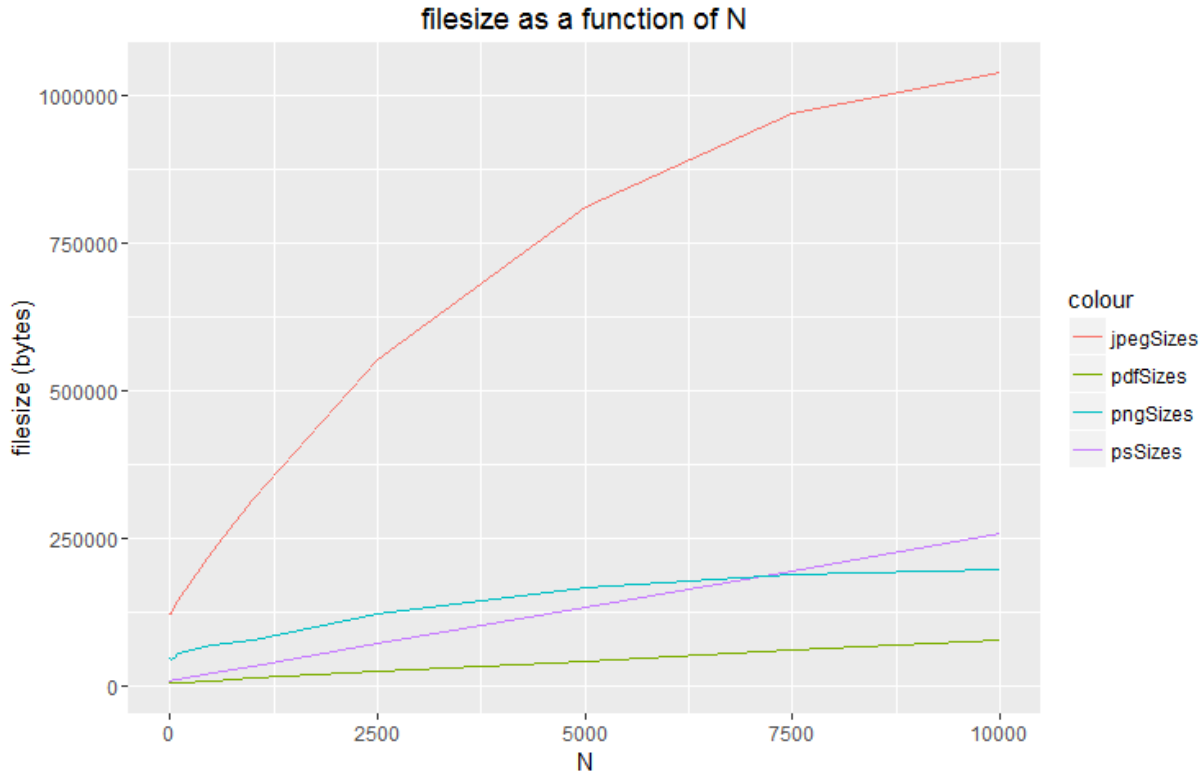
plot filesize as a function of N

```
ggplot(df, aes(N)) +
  geom_line(aes(y = psSizes, color = "psSizes")) +
  geom_line(aes(y = pdfSizes, color = "pdfSizes")) +
  geom_line(aes(y = jpegSizes, color = "jpegSizes")) +
  geom_line(aes(y = pngSizes, color = "pngSizes")) +
  xlab("N") +
  ylab("filesize (bytes)") +
  ggtitle("filesize as a function of N")
```

If I save the file to disk, the resulting file sizes for the various file formats are:

```
       N psSizes pdfSizes jpegSizes pngSizes
1      1    8428     4477    121428    45856
2     10    8637     4560    121782    45319
3     50    9830     4971    133132    46053
4    100   11067     5414    145338    54124
5    500   20995     8550    227011    67809
6   1000   33399    12403    316972    78446
7   2500   70604    23412    550988   121637
8   5000  132573    41825    809095   165167
9   7500  194564    60172    968537   188321
10 10000  256656    78551   1037467   196616
11 25000  628084   188642    766456   155657
12 50000 1247917   371524    313128    72075
13 15000  380312   115354    982248   186547
14 35000  876235   261852    535660   117334
```

Resulting Plot:



From the above plot with N from 0 to 10000:

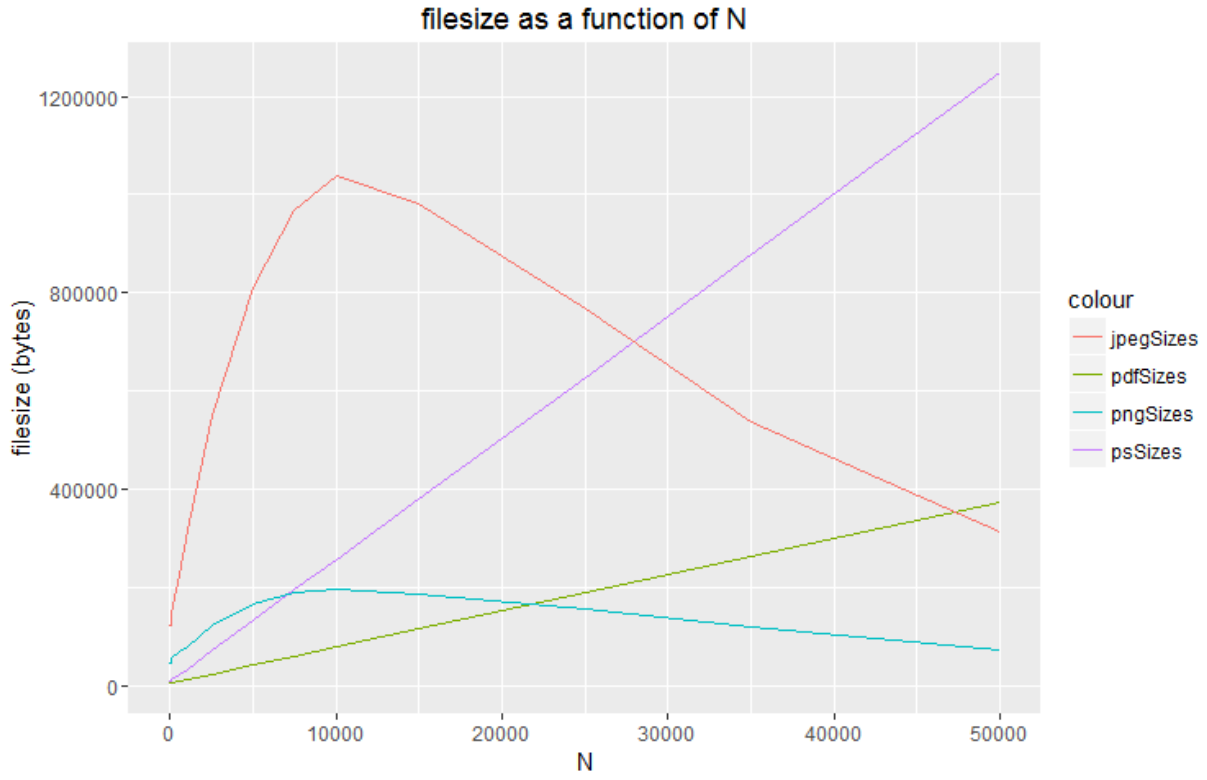
We see that pdf has very low filesizes and increases linearly with increasing N

Jpeg has the highest filesizes but scales as a decaying exponential curve with increasing N

ps starts low but increases linearly with a greater slope with increasing N than pdf

png starts higher than pdf and ps, but increases as a decaying exponential curve with increasing N to the point where ps surpasses png at around N=7500.

Increasing N to 50000, we find:



pdf has very low filesizes and increases linearly with consistently increasing N. Therefore pdf is 2nd highest at N=50000.

ps starts low but increases linearly with a greater slope with consistently increasing N than pdf. Therefore ps is the highest at N=50000.

Jpeg begins to decrease at around N=10000 and continues to decrease at quite a fast rate to the point that it becomes less than pdf at around N=47000 .

png starts to decrease at around 10000 and becomes less than pdf at around N =21000 and continues to decrease!

 # 5. The diamonds dataset within ggplot2 contains 10 columns (price, carat,

 # cut, color, etc.) for 53940 different diamonds. Type help(diamonds) for

 # more information. Plot histograms for color, carat, and price, and comment

 # on their shapes. Investigate the three-way relationship between price, carat,

 # and cut. What are your conclusions? Provide graphs that support your

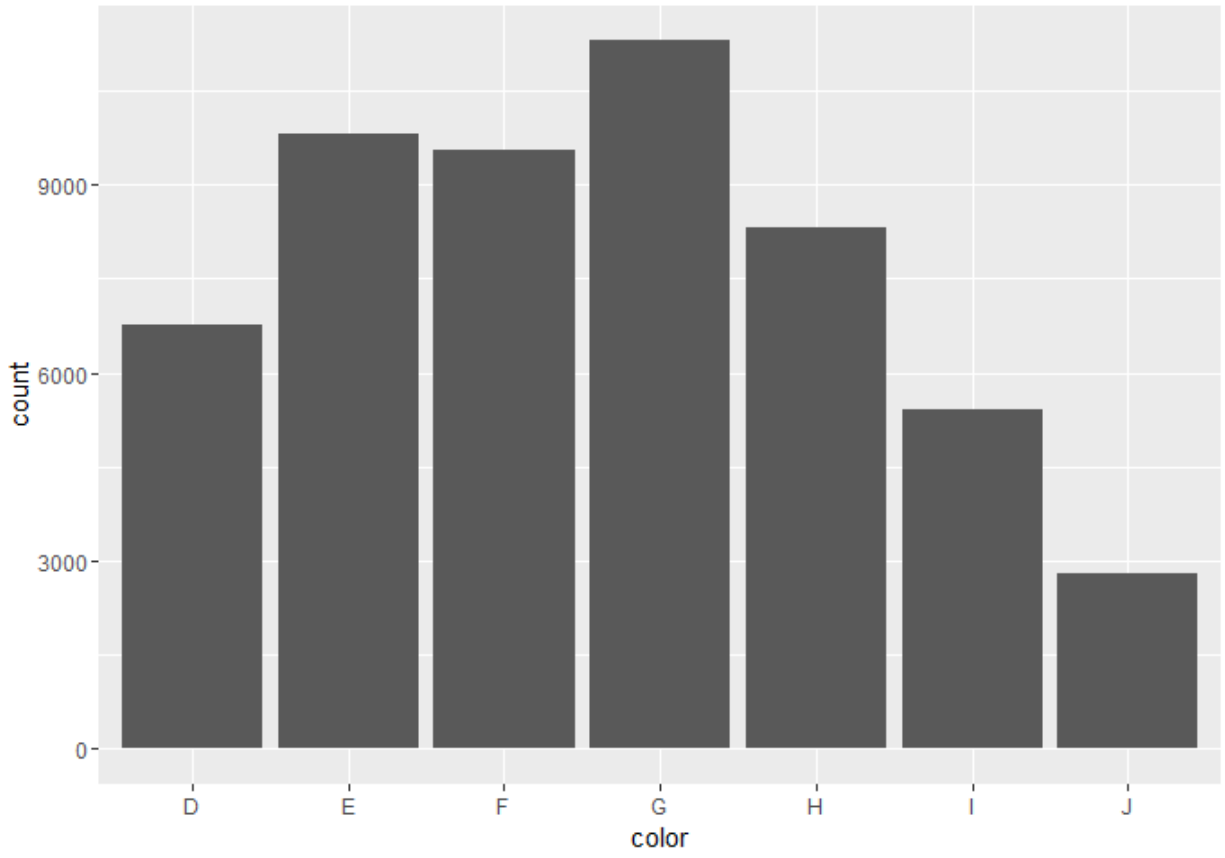
 # conclusions. If you encounter computational difficulties, consider using a

 # smaller dataframe whose rows are sampled from the original diamonds

 # dataframe. Use the function sample to create a subset of indices that

 # may be used to create the smaller dataframe.

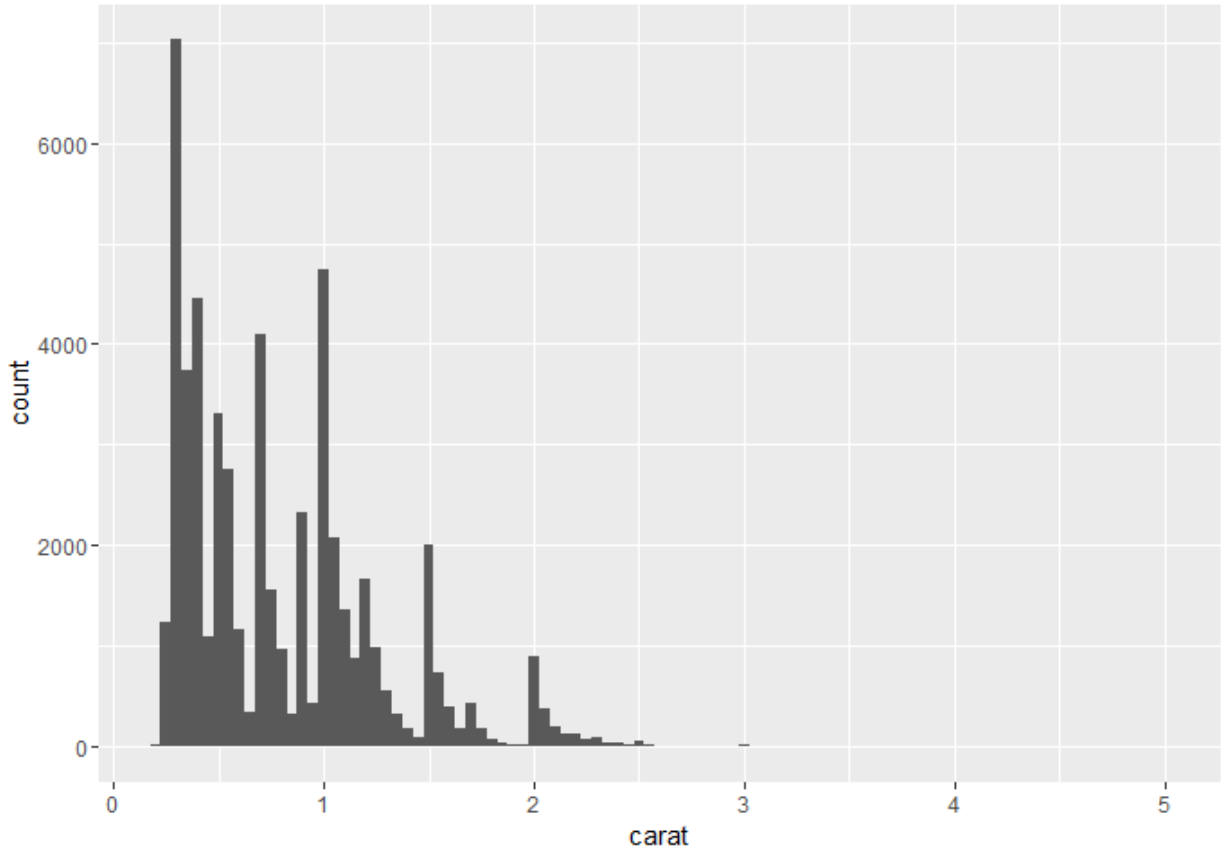
```
library(ggplot2)

data(diamonds)

# Plot histograms for color, carat, and price
#use bar graph for color since it's discrete

qplot(diamonds$color)
```



With color, we find a sort of Gaussian distribution which has higher values on the left. The highest values are near the centre but slightly to the left: E, F, G, H. Whereas D, I, J on the outside are the lowest.

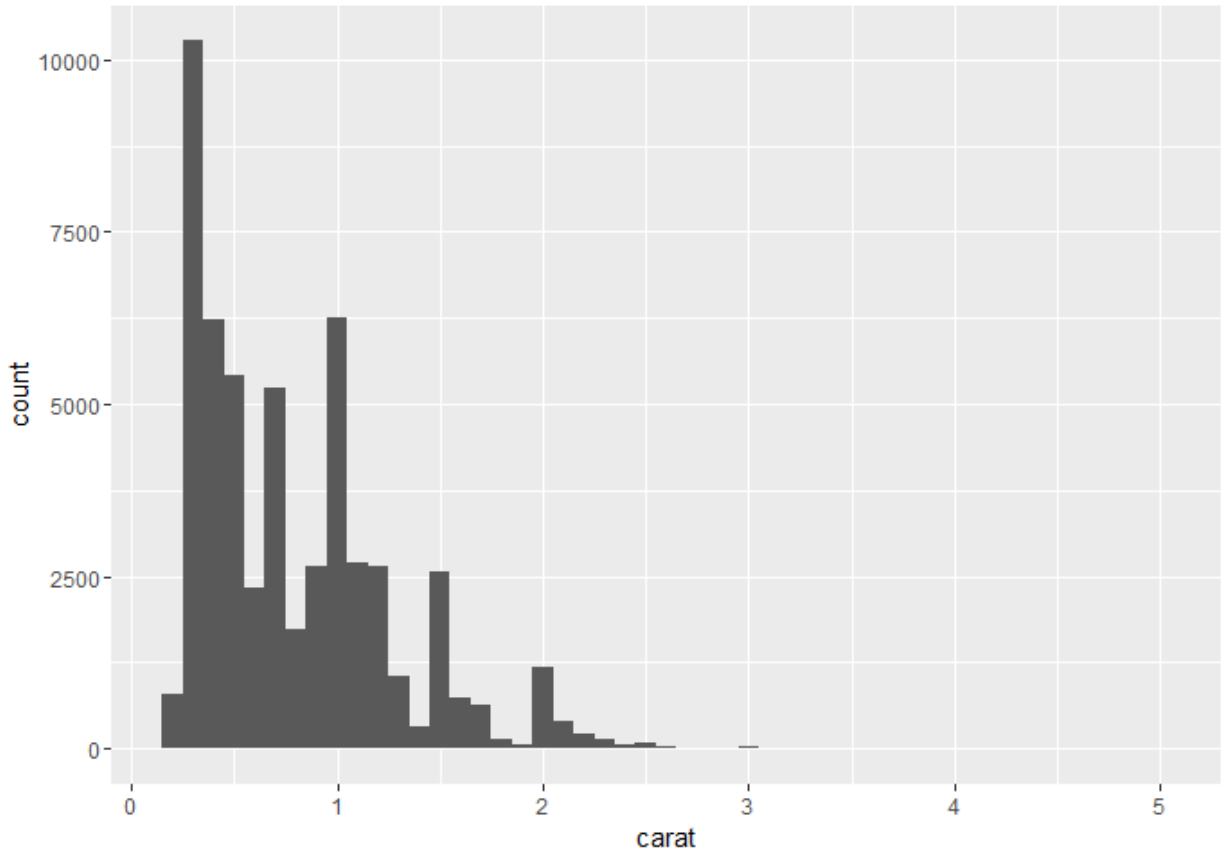
#histograms

```
ggplot(diamonds ,aes(x = carat)) +
  geom_histogram(binwidth = .05)
```

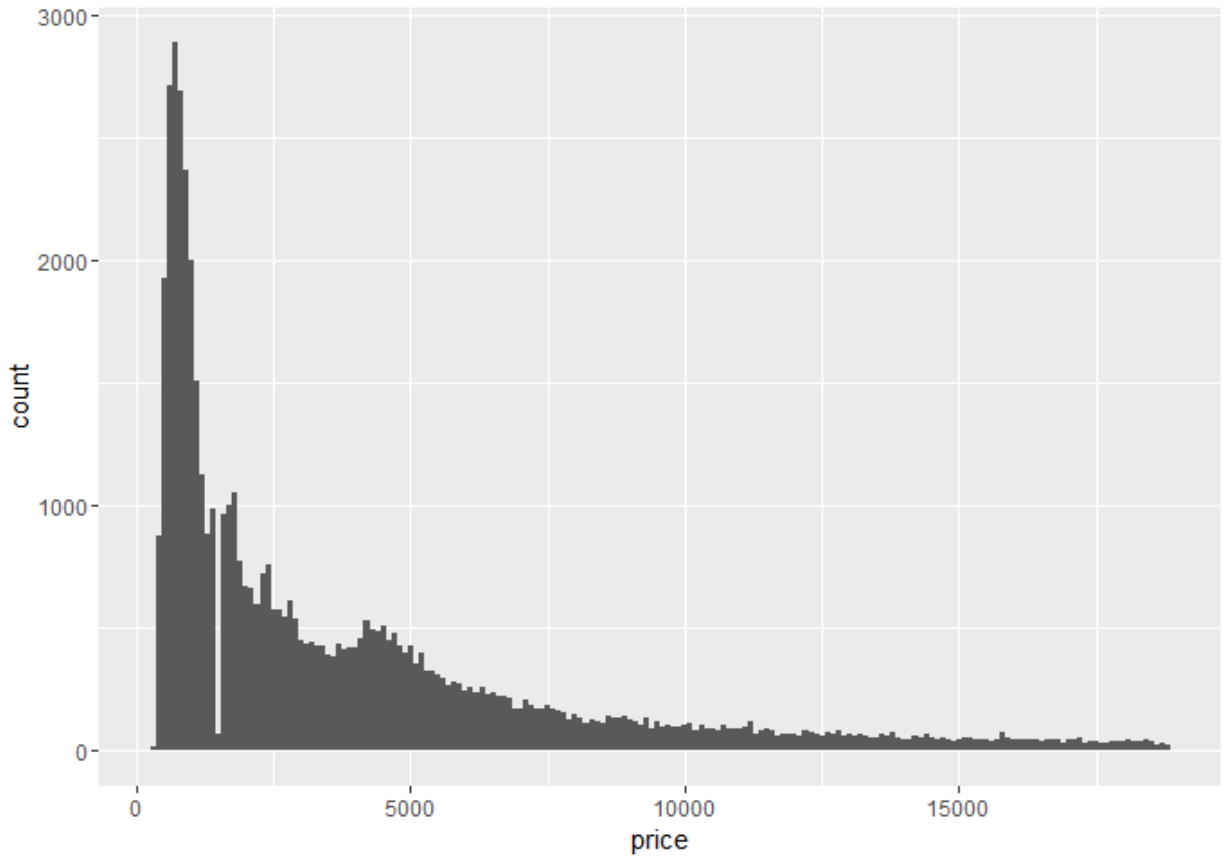


At a binwidth of .05, Carat seems to have 5 discrete values with decaying curve coming after them. Eg. Around .3, .7, 1, 1.5, 2

Similarly at a bandwidth of .1 to hide some noise




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



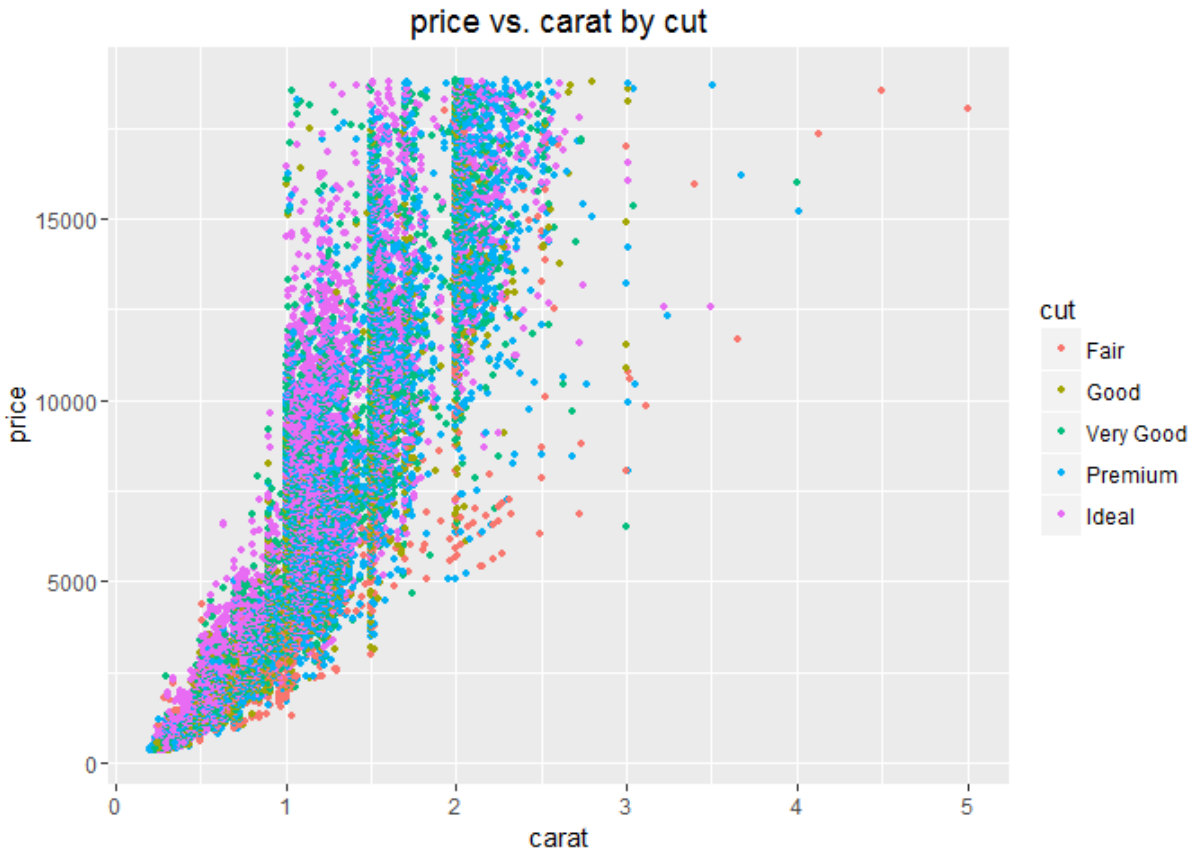
Price looks like a decaying exponential curve. There's a very sharp peak at around $1000.

#Investigate the three-way relationship between price, carat,

and cut.

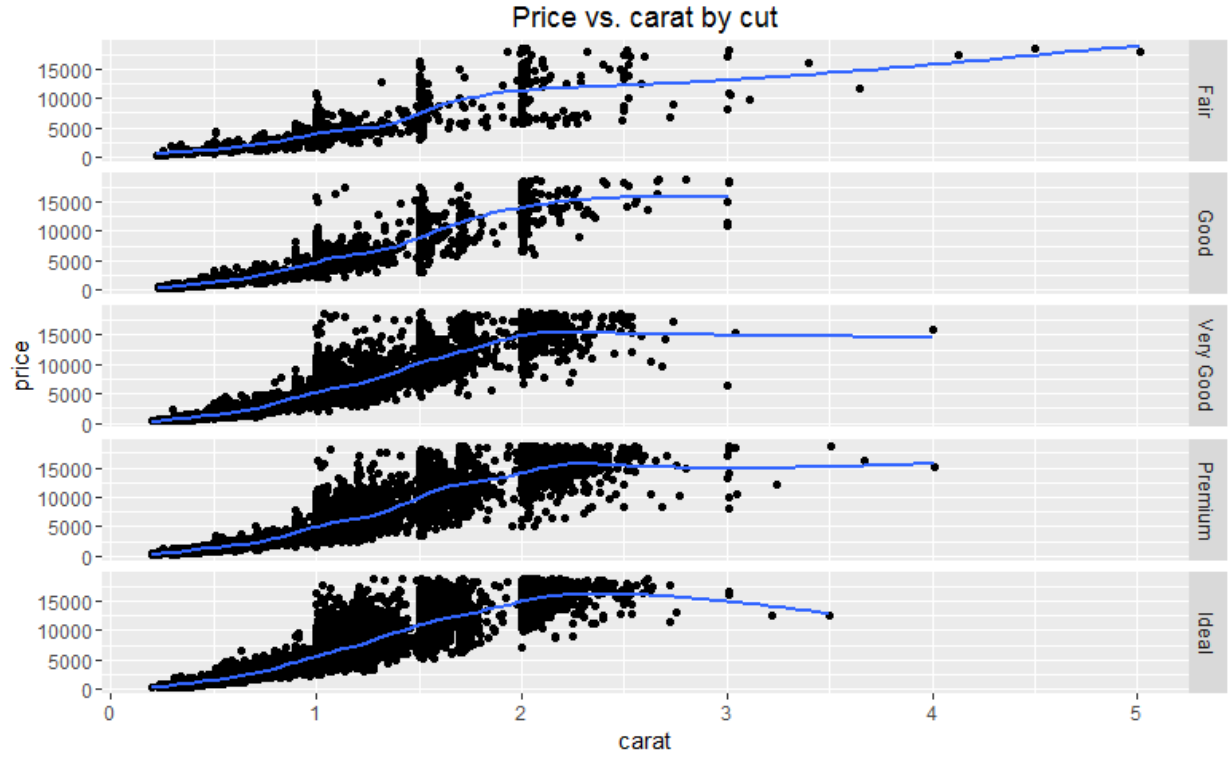
Plot a scatter plot with markers of different colors for cut

```
qplot(x = carat,
      y = price,
      data = diamonds,
      color = cut,
      size=I(1),
      main = "price vs. carat by cut")
```




```
#multiple panel plot
qplot(x = carat,
      y = price,
      facets = cut~.,
      data = diamonds,
      main = "Price vs. carat by cut") +
```

stat_smooth(se = FALSE)



Investigating the 3 way relationship between price, carat, and cut:

In general price increases with carat.

In general the smoothed line curve shows somewhat of an “S” shape, where the slope starts off small, then increases and then decreases.

We see that with fair cut, price increase slowly with carat, but increases indefinitely while the others taper off and ideal even decreases in price after we pass the 2.5 carat mark.

It appears as if the higher quality cuts increase in price faster and reach their maximum price faster.