GRAMMAR OF GRAPHICS

Dr. Ami Gates,
Data Visualization

RESOURCES AND REFERENCES FOR GGPLOTS

1) **Book: ggplot by Hadley Wickham**

2) http://vita.had.co.nz/papers/layered-grammar.pdf

3) **Book**: The Grammar of Graphics, Leland Wilkinson

https://www.amazon.com/gp/product/0387245448/ref=as_li_ss_tl?ie=UTF8&camp=1789&creative=390957&creativeASIN=0387245448&linkCode=as2&tag=civilstatis-20

4) Review of Wilkinson

https://drive.google.com/file/d/0B4RXVYeUUKitbmJuRk4zaW45X2M/view

5) Examples in ggplot/R

https://www3.nd.edu/~steve/computing_with_data/11_geom_examples/ggplot_examples.html

https://rstudio-pubs-static.s3.amazonaws.com/86115_e78c3a8e3ec9446892a3bc1838e170c4.html

DATASETS

http://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

WHAT IS THE GRAMMAR OF GRAPHICS

“A grammar of graphics is a tool that enables us to concisely describe the components of a graphic….beyond named graphics (such as a scatterplot….and to gain insight into the deep structure that underlies statistical graphics. “ (**Wickham, 2009**)

“In ggplot, we can produce many plots that don't make sense, yet are grammatically valid. This is no different than English, where we can create senseless but grammatical sentences like **the angry rock barked like a comma**.”, (https://github.com/hadley/ggplot2-book/blob/master/mastery.rmd)

GRAMMAR OF GRAPHICS (GG)
REF: WILKINSON,2005

Grammar of Graphics

❑ A set of independent components that can be "**composed**" in different ways – no pre-specified graphics.

❑ Offers options to create visualizations that are tailored to the application.

❑ Describes "deep" features that underlie all statistical graphics.

ggplot2 in R

❑ The underlying "grammar" of ggplot2 is based on the grammar of graphics.

❑ We will use ggplot2 to both investigate the grammar or graphics and to produce visualizations.

ggplot2 is a plotting system for R that contains ggplot() and qplot() and can create **layered graphics.**

WHAT THE GRAMMAR SUGGESTS

- ❑ A graphic is a “mapping” from data to **aesthetic attributes** (color, shape, size, position) of **geometric objects** (point types, bars, lines, etc.)
- ❑ A graphic may contain **statistical transformations** of the data
- ❑ **Faceting** can be used to generate the same graphic for different subsets of the data.

BASIC TERMINOLOGY

- **Data**: what you want to visualize.
- **Aesthetic Mappings**: How variables in the data are mapped to aesthetic attributes.
- **Geoms:** Geometric objects that are visible on a graphic, such as points, lines, polygons, etc.
- **Stats**: Statistical measures and transformations that summarize or describe data, such as binning.
- **Scales**: Map data values to aesthetic space, via color, size, shape, etc. Scales draw a legend or axes.
- **Coord**: The coordinate system that describes how data are mapped to the plane of the graphic (if 2D). This can be Cartesian, polar, map-based, etc.
- **Facet**: A faceting describes how to break up and display the data as subsets – latticing/trellising.

THE DIAMONDS DATASET

The next several examples will use the diamonds dataset from R and will reference Chapter 2 in the Wickham book.

1) Check if you have the diamonds dataset:

Type the following into R

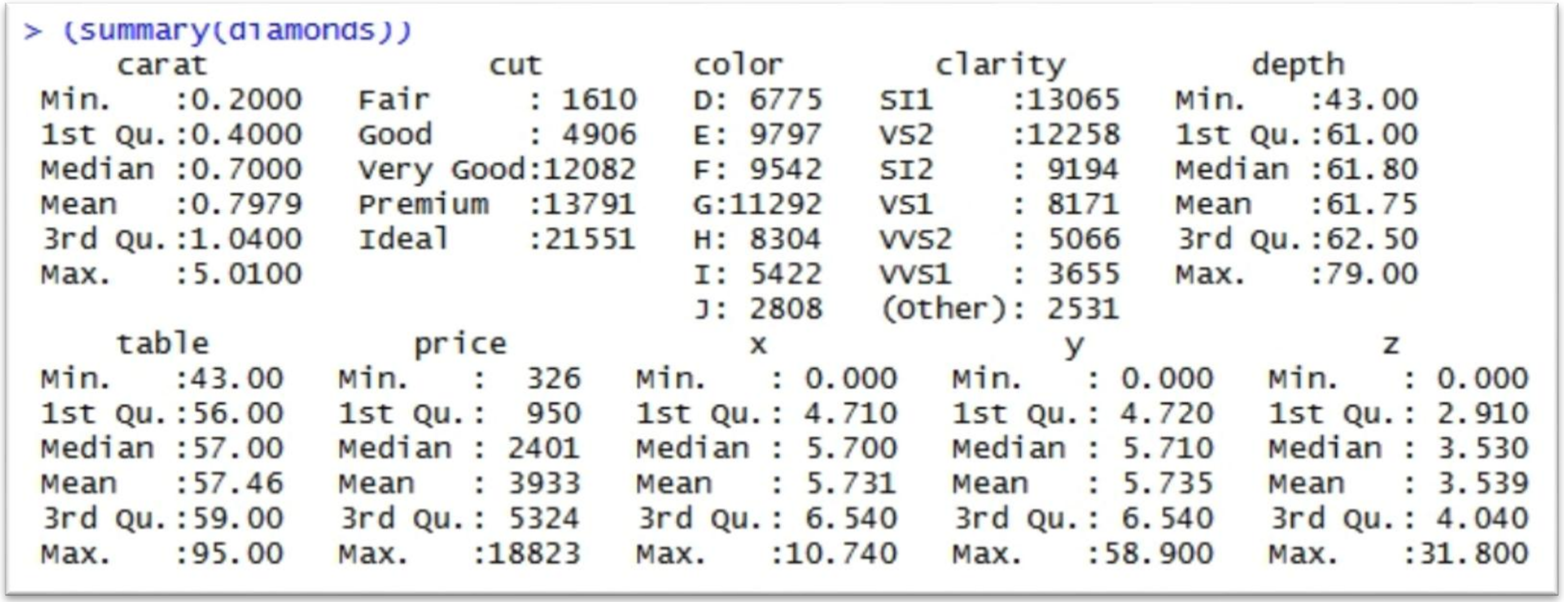
#DO THIS ONCE

#install.packages("ggplot2")

library(ggplot2)

(summary(diamonds))

Output:



```
> (summary(diamonds))
     carat                cut        color        clarity          depth      
 Min.   :0.2000   Fair     : 1610   D: 6775   SI1    :13065   Min.   :43.00  
 1st Qu.:0.4000   Good     : 4906   E: 9797   VS2    :12258   1st Qu.:61.00  
 Median :0.7000   Very Good:12082   F: 9542   SI2    : 9194   Median :61.80  
 Mean   :0.7979   Premium  :13791   G:11292   VS1    : 8171   Mean   :61.75  
 3rd Qu.:1.0400   Ideal    :21551   H: 8304   VVS2   : 5066   3rd Qu.:62.50  
 Max.   :5.0100                     I: 5422   VVS1   : 3655   Max.   :79.00  
                                    J: 2808   (Other): 2531                  
     table           price             x                y                z         
 Min.   :43.00   Min.   :  326   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000  
 1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910  
 Median :57.00   Median : 2401   Median : 5.700   Median : 5.710   Median : 3.530  
 Mean   :57.46   Mean   : 3933   Mean   : 5.731   Mean   : 5.735   Mean   : 3.539  
 3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040  
 Max.   :95.00   Max.   :18823   Max.   :10.740   Max.   :58.900   Max.   :31.800  
```

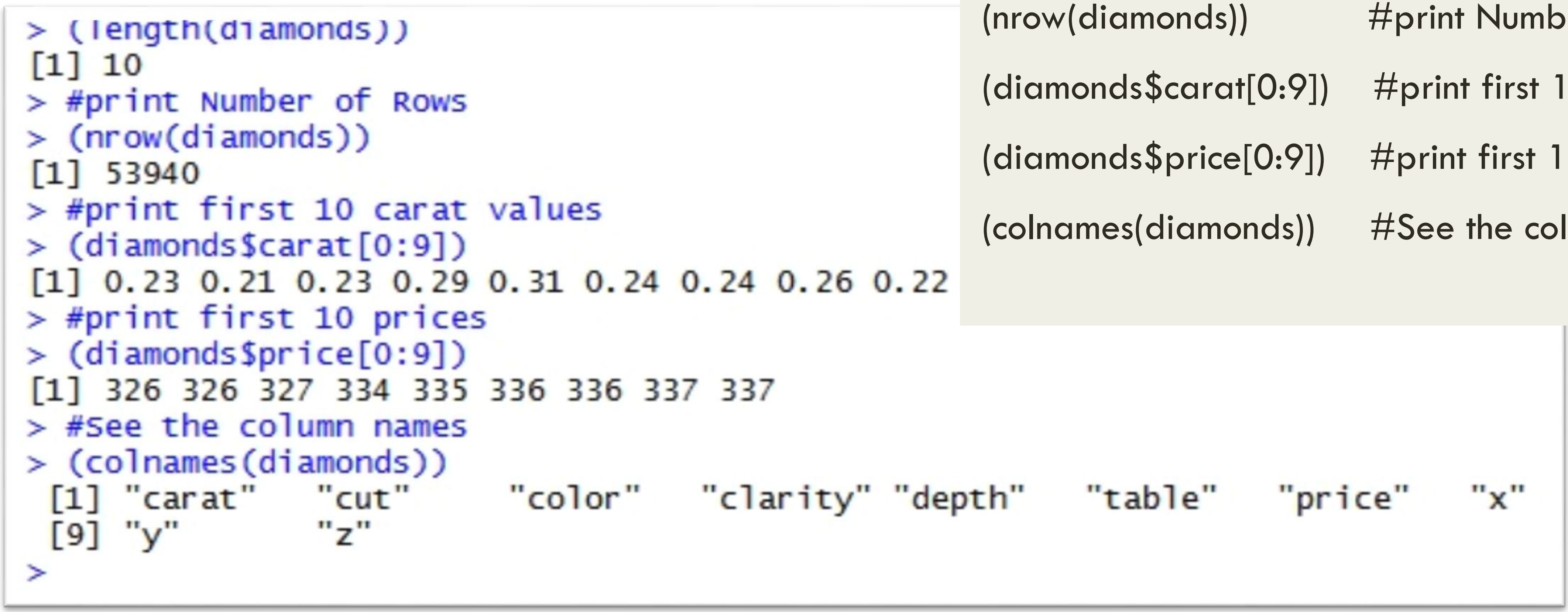
Note: the metric, “carat”

is 200mg (.2 g). The “cut”

is the symmetry, proportion, polish. The “color” ranges from D (colorless) onward (yellower). The “clarity” ranges from FL (flawless) to I3 (not as brilliant).

EXPLORE THE DIAMONDS DATASET

```
#Bring in the ggplot2 library which contains the
#diamonds dataset

library(ggplot2)

(summary(diamonds))        #Summarize the dataset

(length(diamonds))         #print Number Columns

(nrow(diamonds))           #print Number of Rows

(diamonds$carat[0:9])      #print first 10 carat values

(diamonds$price[0:9])      #print first 10 prices

(colnames(diamonds))       #See the column names
```



```
> (length(diamonds))
[1] 10
> #print Number of Rows
> (nrow(diamonds))
[1] 53940
> #print first 10 carat values
> (diamonds$carat[0:9])
[1] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22
> #print first 10 prices
> (diamonds$price[0:9])
[1] 326 326 327 334 335 336 336 337 337
> #See the column names
> (colnames(diamonds))
 [1] "carat"   "cut"     "color"   "clarity" "depth"   "table"   "price"   "x"
 [9] "y"       "z"
>
```

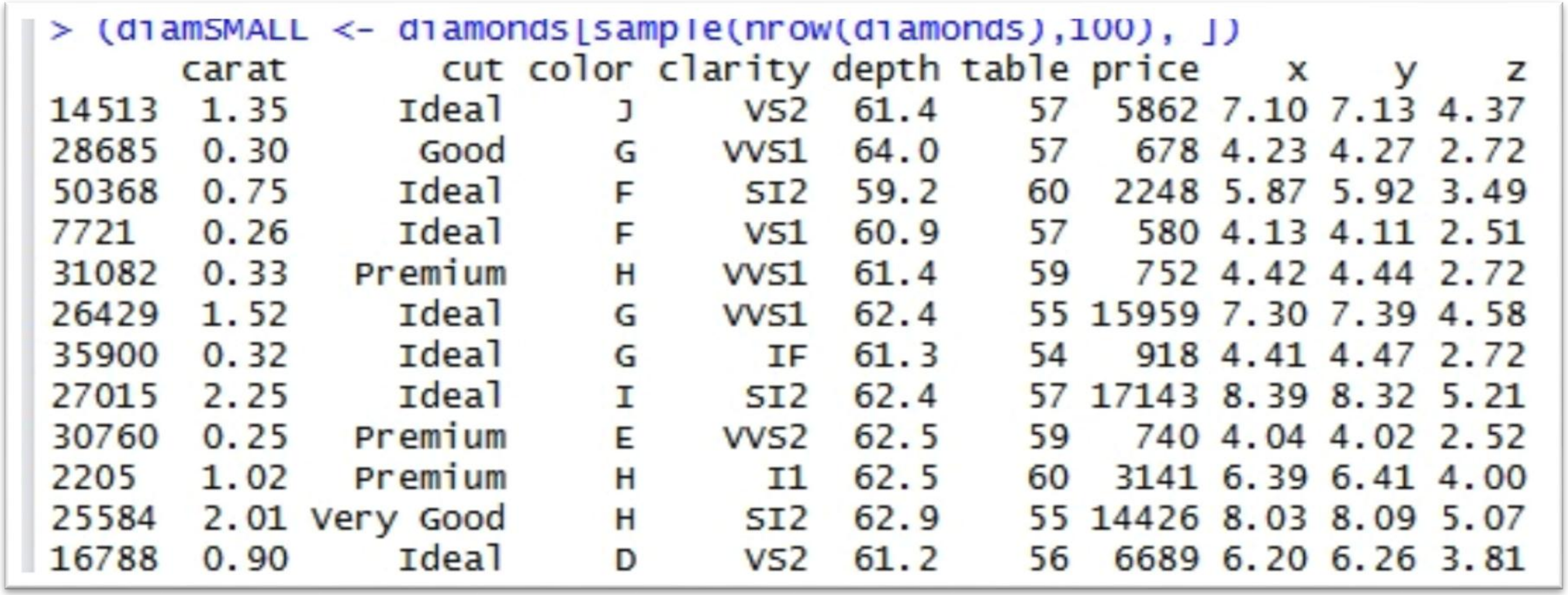
SAMPLING A LARGER DATASET

set.seed(1410) #Non random sample. The 1410 is irrelevant.

#Without the set.seed, each sample generates a random sample from the dataset

#100 rows and all columns and print it

(diamSMALL <- diamonds[**sample**(nrow(diamonds),100),])



```
> (diamSMALL <- diamonds[sample(nrow(diamonds),100), ])
      carat       cut color clarity depth table price    x    y    z
14513  1.35     Ideal     J     VS2  61.4    57  5862 7.10 7.13 4.37
28685  0.30      Good     G    VVS1  64.0    57   678 4.23 4.27 2.72
50368  0.75     Ideal     F     SI2  59.2    60  2248 5.87 5.92 3.49
7721   0.26     Ideal     F     VS1  60.9    57   580 4.13 4.11 2.51
31082  0.33   Premium     H    VVS1  61.4    59   752 4.42 4.44 2.72
26429  1.52     Ideal     G    VVS1  62.4    55 15959 7.30 7.39 4.58
35900  0.32     Ideal     G      IF  61.3    54   918 4.41 4.47 2.72
27015  2.25     Ideal     I     SI2  62.4    57 17143 8.39 8.32 5.21
30760  0.25   Premium     E    VVS2  62.5    59   740 4.04 4.02 2.52
2205   1.02   Premium     H      I1  62.5    60  3141 6.39 6.41 4.00
25584  2.01 Very Good     H     SI2  62.9    55 14426 8.03 8.09 5.07
16788  0.90     Ideal     D     VS2  61.2    56  6689 6.20 6.26 3.81
```

QPLOT( ) : QUICK PLOT: A GGPLOT2 FUNCTION

Syntax:

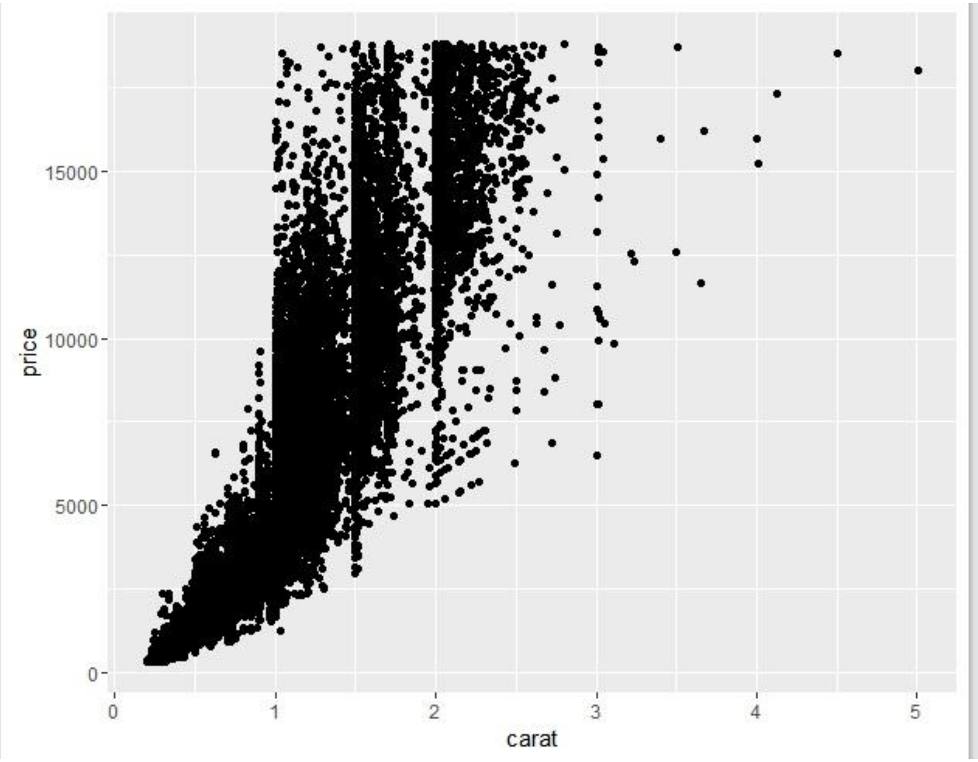
qplot(x, y, data, color, attribute=value, alpha, geom, method, span, formula, xlim, ylim, …)

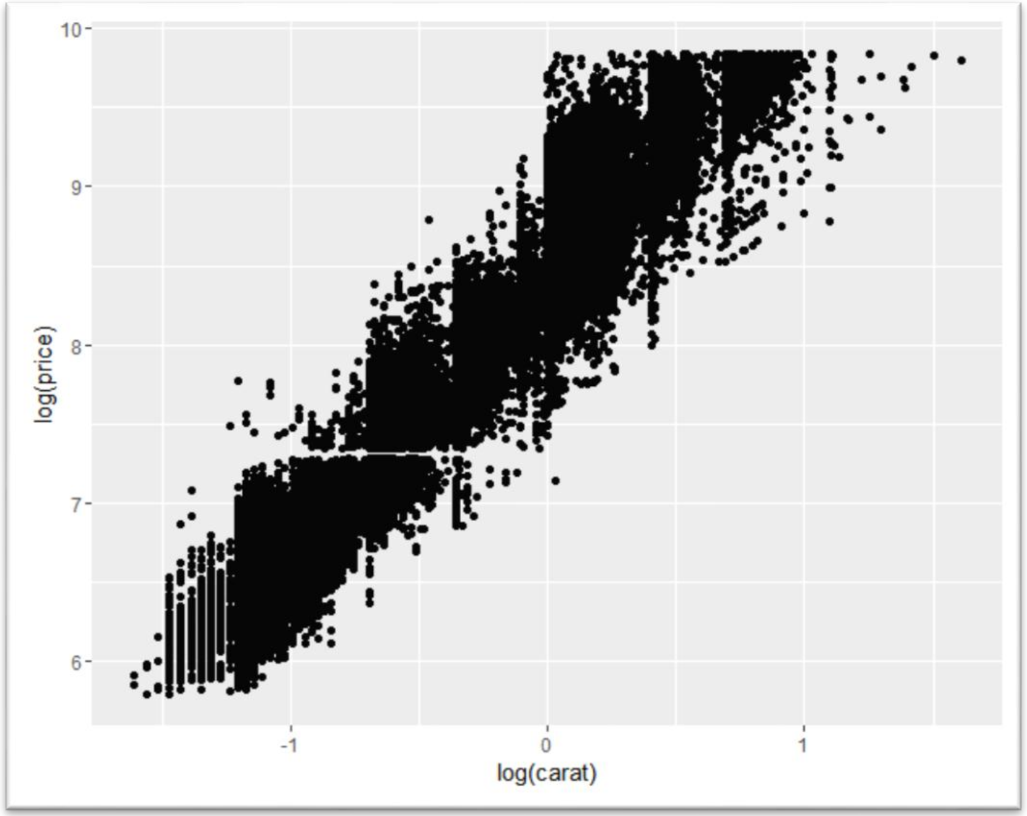
The following examples will explore several uses of qplot, several graph types, and concepts in Grammar of Graphics that relate to each.

QPLOT: SCATTERPLOT

##qplot with log of x and y
qplot(log(carat), log(price), data=diamonds)

qplot(carat, price, data=diamonds)





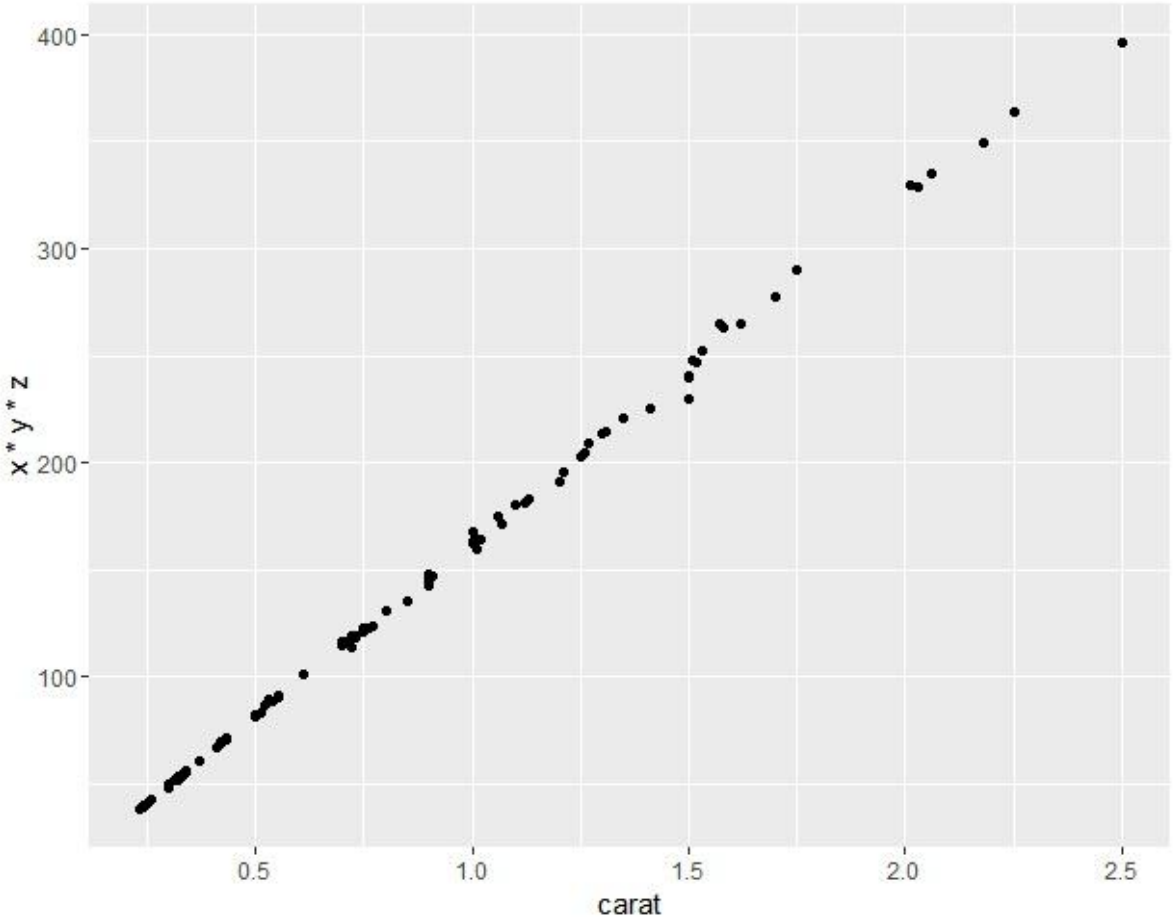
X AND Y OPTIONS IN QPLOT

## qplot will plot functions of data

## Recall that x, y, and z are the dim of ##each diam

qplot(carat, x*y*z, data=diamSMALL)

What does this plot tell us?

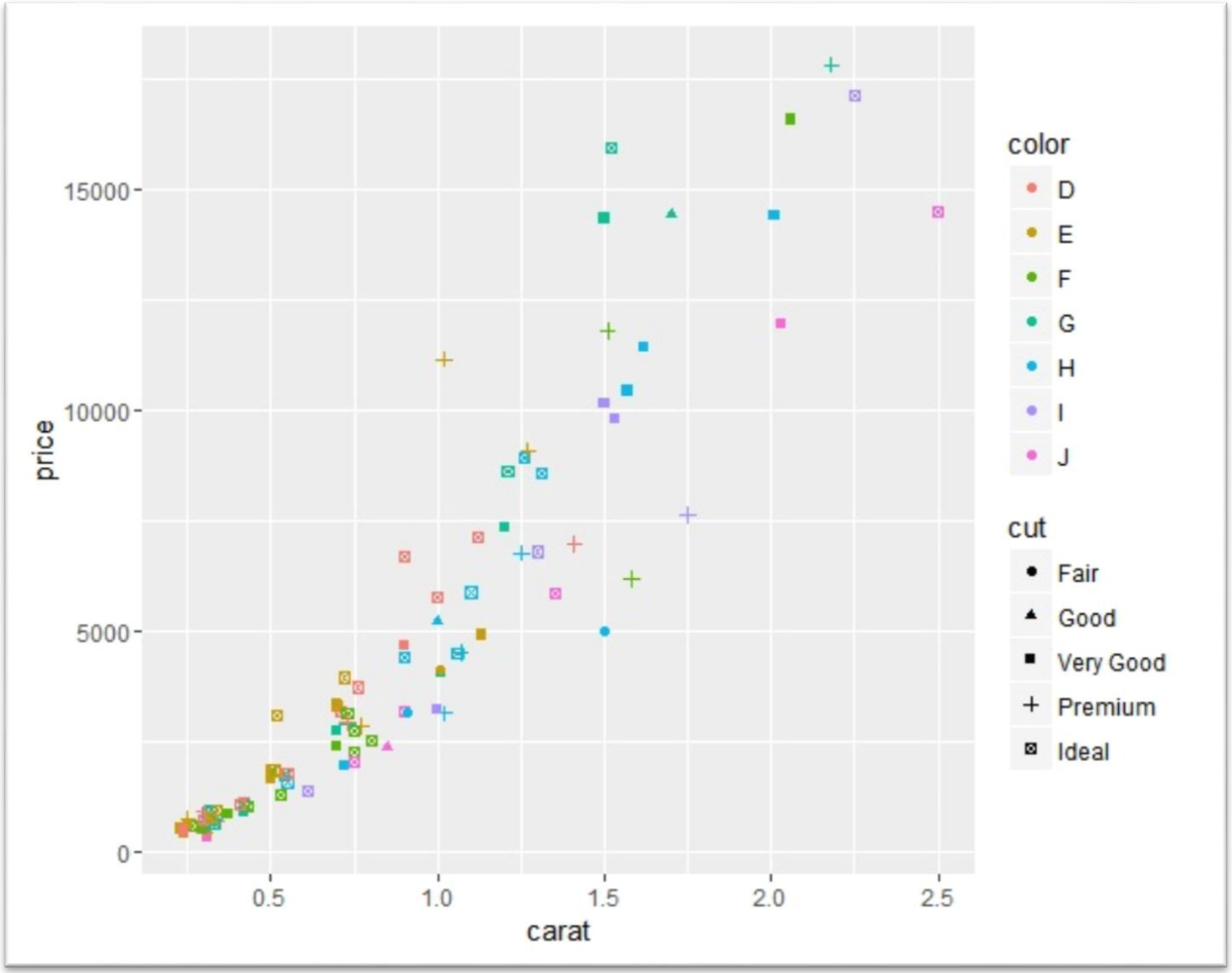


AESTHETICS: COLOR, SHAPE, AND SIZE

## qplot automatically creates a #legend and displays categorized #color

## It can also alter the shape of #the categories

qplot(carat, price, data=diamSMALL, shape=cut, color=color)



AESTHETICS

- Within the Grammar of Graphics: color, shape, and size are called **aesthetics.**
- These are **visual properties** that can affect the way observations are displayed and therefore viewed.
- A difference between plot and qplot is that qplot can automatically assign aesthetics (color, shape, size, etc.) to variables.

```
qplot(carat, price, data=diamonds, shape=cut)

# This will automatically map the shape of the point to the diamond cut and will include a legend.
```

THE I( ) FUNCTION

Aesthetics can also be **manually mapped** using the **I( )** function.

Size and color are both set by hand using "I( )" below. The option, **alpha**=I(1/3) affects the **transparency** of the points.

Example:

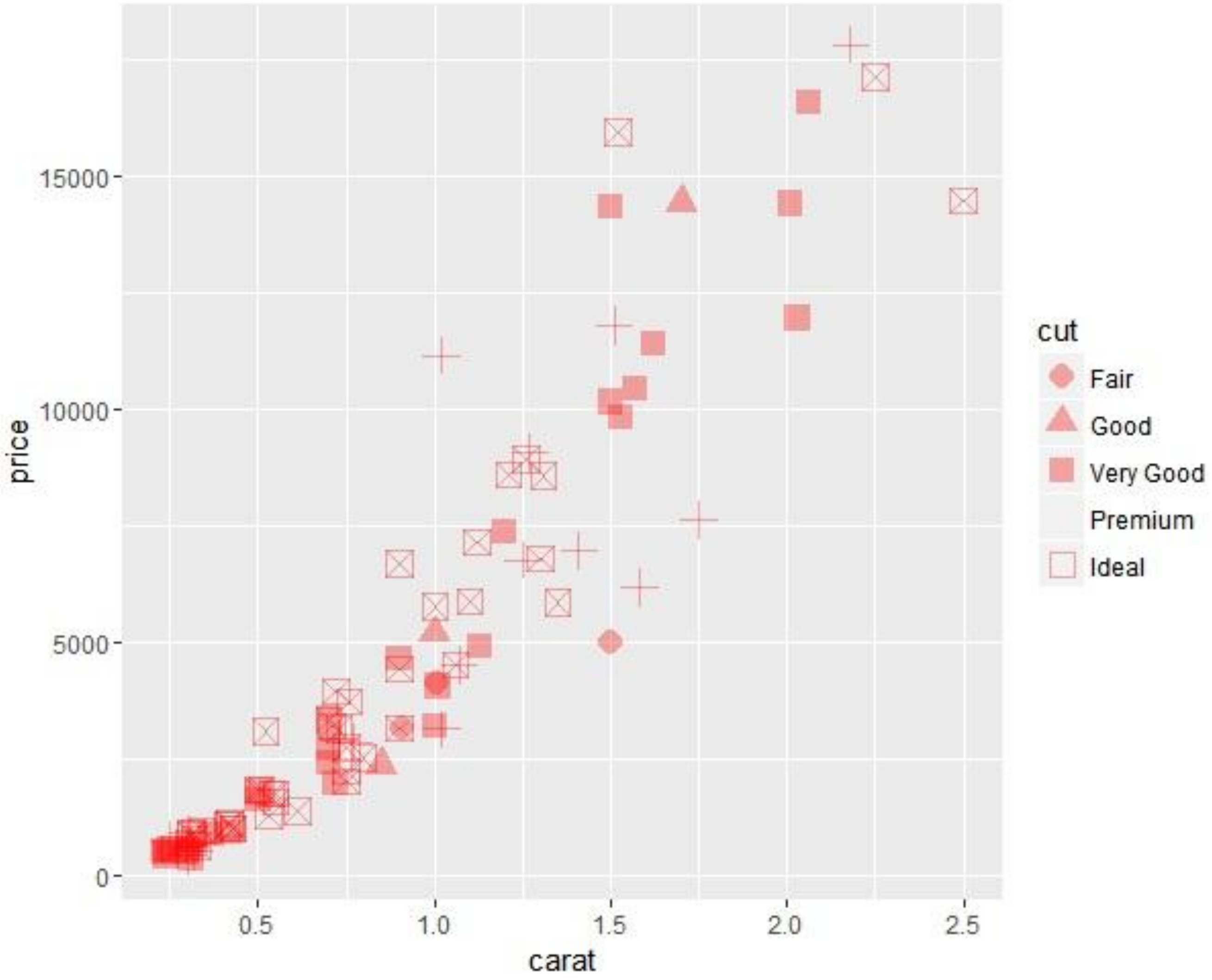
```
qplot(carat, price, data=diamSMALL, shape=cut, color=I("red"), size=I(4),
alpha=I(1/3))
```

Result on next slide…

Some aesthetics work betters with specific variable types.

For example: "color" is better for categorical data.

COLOR=I("RED"), SIZE=I(4), ALPHA=I(1/3)



```
qplot(carat, price, data=diamSMALL, shape=cut, color=I("red"), size=I(4), alpha=I(1/3))
```

GEOMS: GEOMETRIC OBJECTS

Some **geoms** require initial **data transformations**

Example: A histogram is a binning statistic as well as a bar geom.

Most Common Geoms:

1) **geom="point"**: draws points

2) **geom="smooth"**: fits a smooth to the data and displays smooth and standard error

3) **geom="boxplot"**: box and whisker to summarize distribution of points

4) **geom="path"** and **geom="line"**: draws lines between datapoints. Often used to consider "time" versus another variable.

5) **geom="histogram":** is great for continuous variables

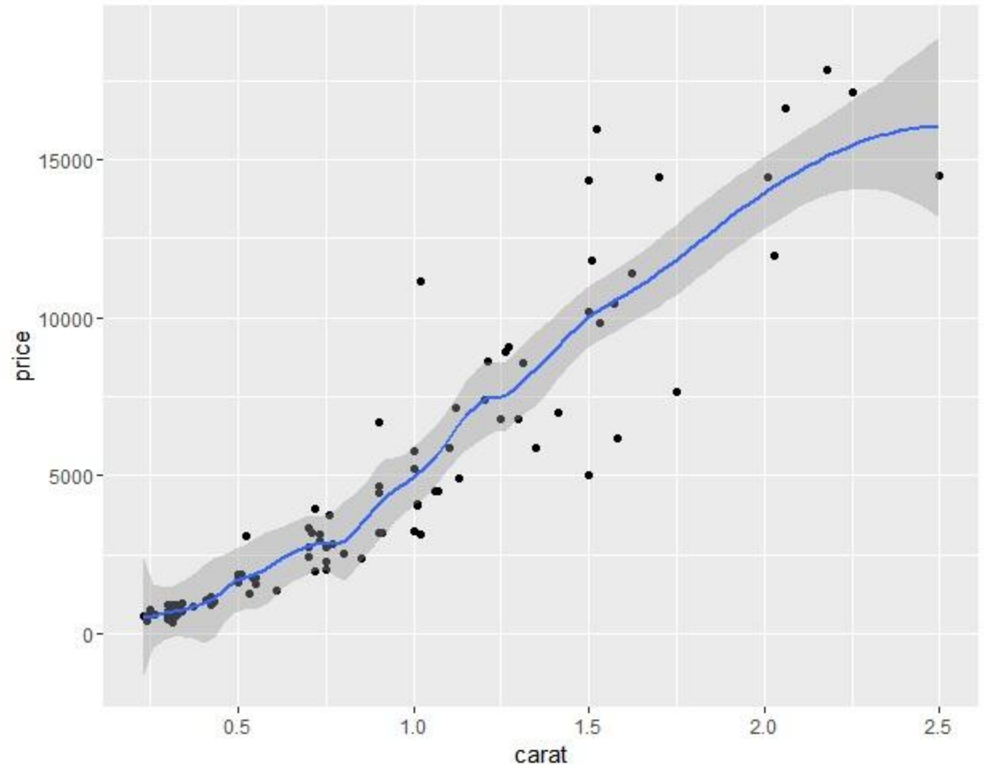
6) **geom="freqpoly":** frequency polygon

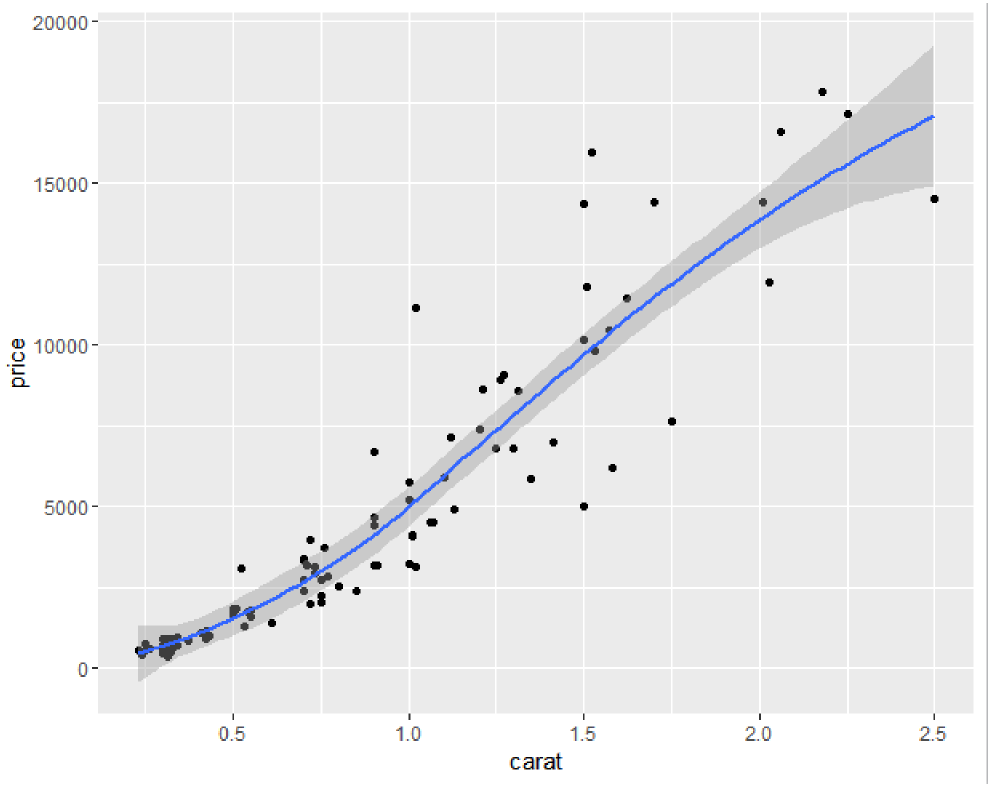
7) **geom="density"**: creates a density plot

8) **geom="bar":** creates a bar graph

GEOM: POINT AND SMOOTH 0<SPAN<1 WITH SPAN→1, LEAST WIGGLY

NOTICE: The newer version of ggplot2 uses **geom_point()** and **geom_smooth().**

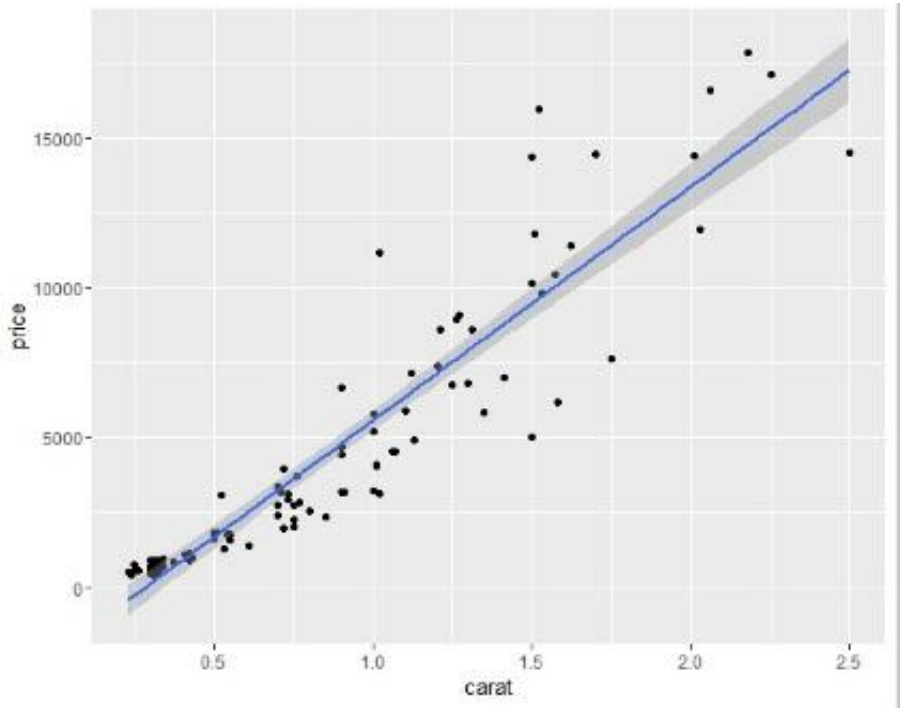


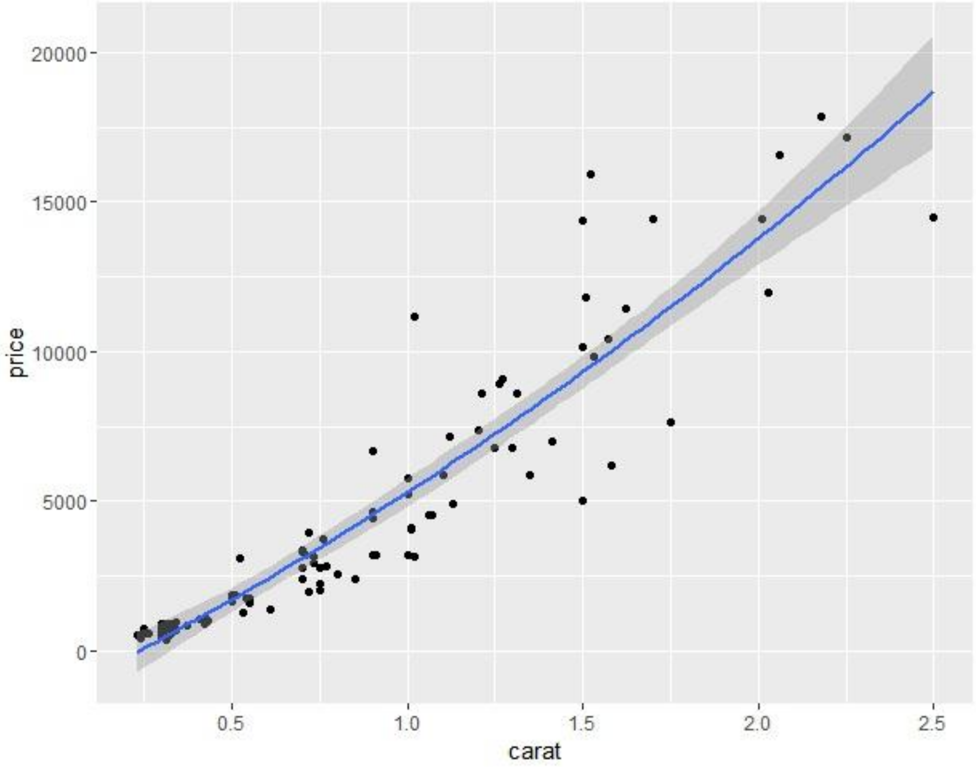


```
qplot(carat, price, data=diamSMALL) +
 geom_point() + geom_smooth(span = .3)
```

```
qplot(carat, price, data=diamSMALL) +
 geom_point() + geom_smooth(span = .8)
```

SMOOTHING WITH
LM: LINEAR MODEL





qplot(carat, price, data=diamSMALL) + geom_point() + geom_smooth(method="lm",formula=y~poly(x,2), span = .8)

qplot(carat, price, data=diamSMALL) + geom_point() + geom_smooth(method="lm", span = .8)

BOXPLOTS AND JITTERED POINTS

Suppose your dataset contains some variables that are categorical and others that are quantitative.

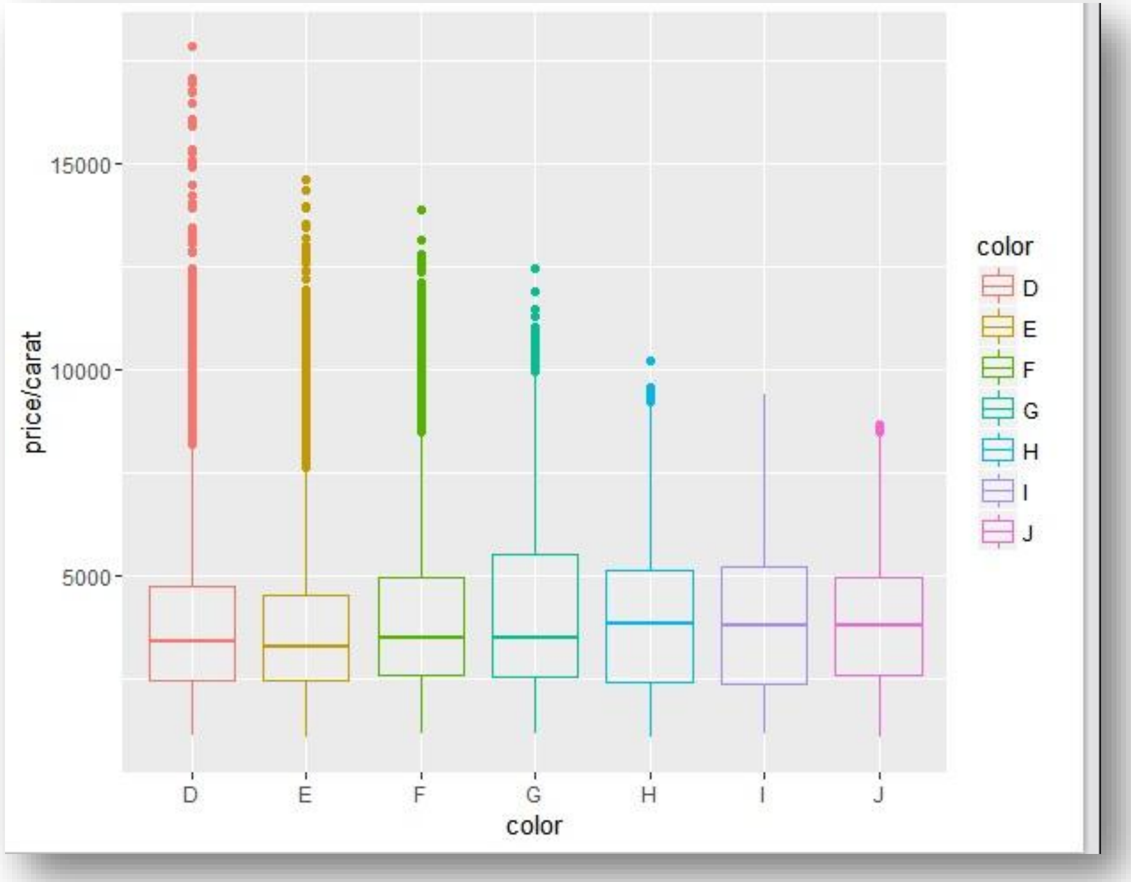
- ❑ One common option is to look at the behavior of the quantitative variable values with respect to the levels of the categorical variables.

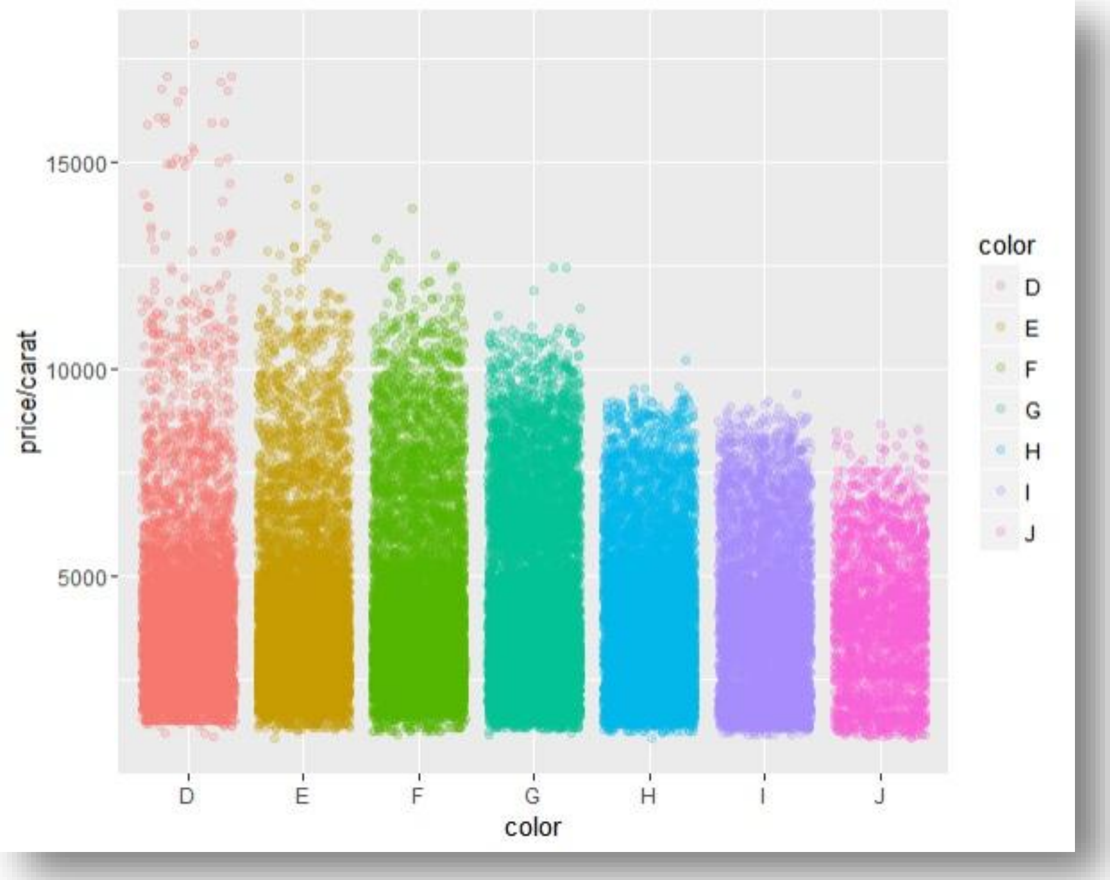
Example: Using the Diamonds data we have been working with, we can look at how price (quantitative and continuous) varies with color.

- Recall that with diamonds, the clearer the better and more "yellow" the lower the price.
- There are two common methods for looking at the variance of continuous data per level of category:

boxplots and **jitter points**…

qplot(color, price/carat, data=diamonds, geom="boxplot", color=color,alpha=I(1/10))





qplot(color, price/carat, data=diamonds, geom="jitter", color=color, alpha=I(1/5))

NOTE: Normalize by carat – why?

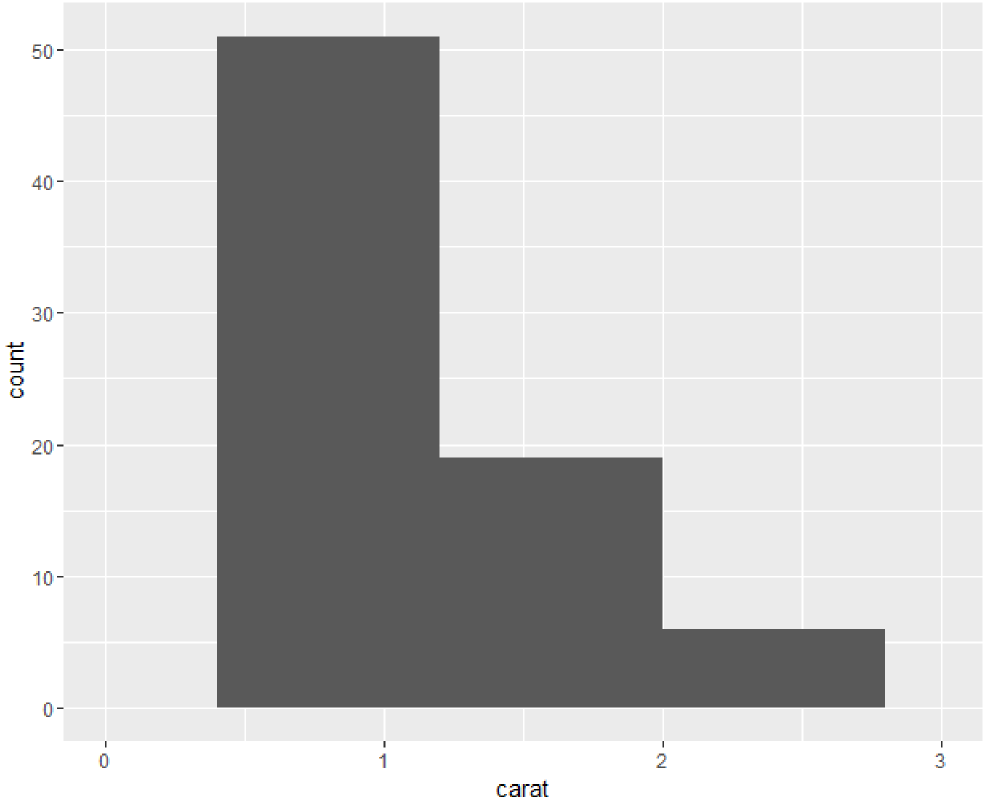
HISTOGRAMS AN DENSITY PLOTS

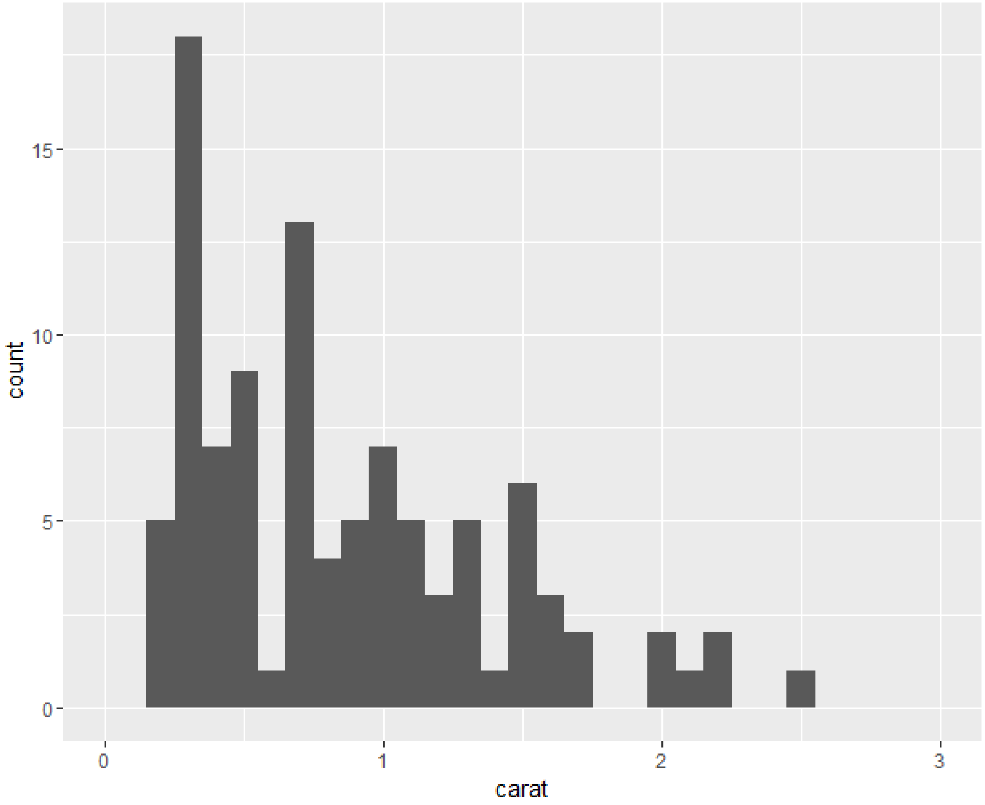
Histograms and density plots can be used to investigate or show the **distribution** of a single variable.

For Density Plots, the “adjust” option is for smoothness, with higher values giving smoother lines.

For the Histogram, the “binwidth” affects smoothness by determining the number of bins. The smaller the binwidth value, the more bins you will have.

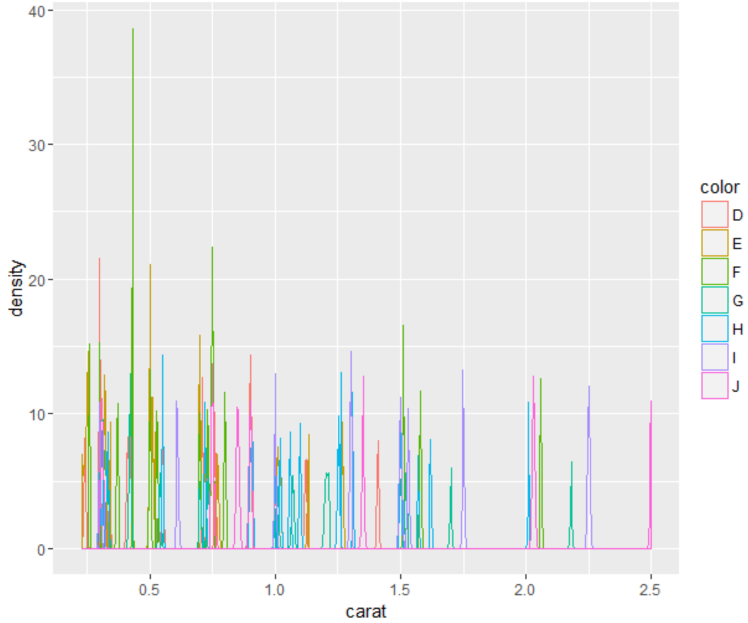
qplot(carat, data=diamSMALL, geom="histogram", binwidth=.8, xlim=c(0,3))

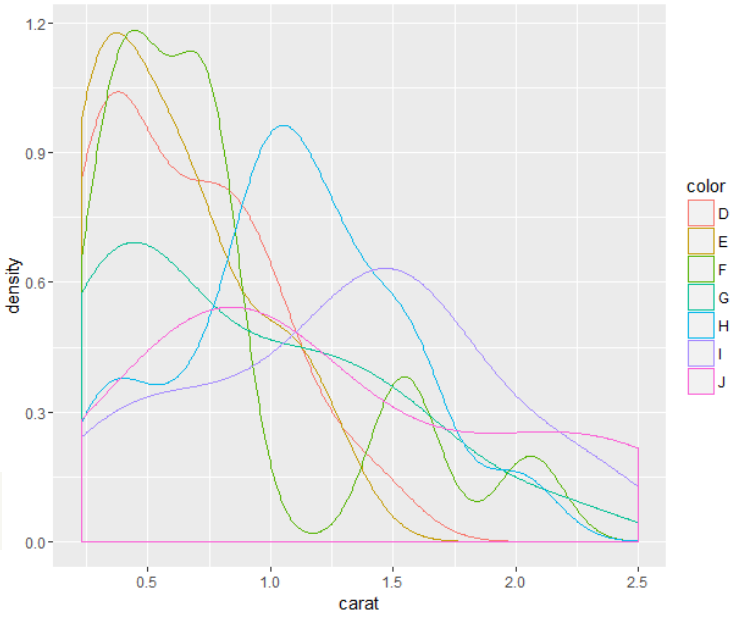


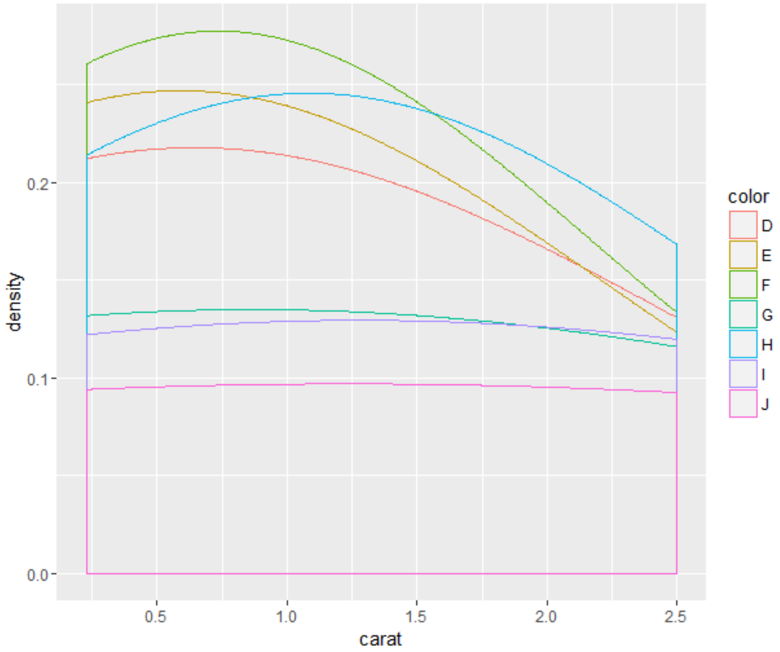


qplot(carat, data=diamSMALL, geom="histogram", binwidth=.1, xlim=c(0,3))

```
QPLOT(CARAT,DATA=DIAMSMALL,GEOM="DENSITY",COLOR=COLOR, ADJUST=.01)
QPLOT(CARAT,DATA=DIAMSMALL,GEOM="DENSITY",COLOR=COLOR, ADJUST=1)
QPLOT(CARAT,DATA=DIAMSMALL,GEOM="DENSITY",COLOR=COLOR, ADJUST=10)
```





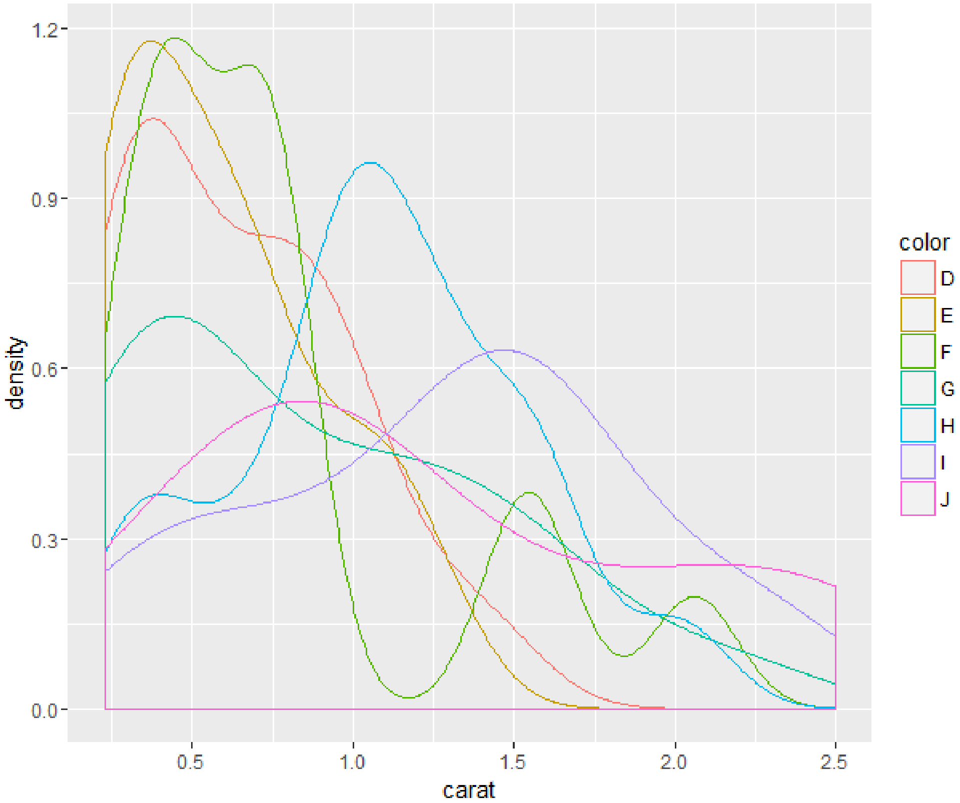


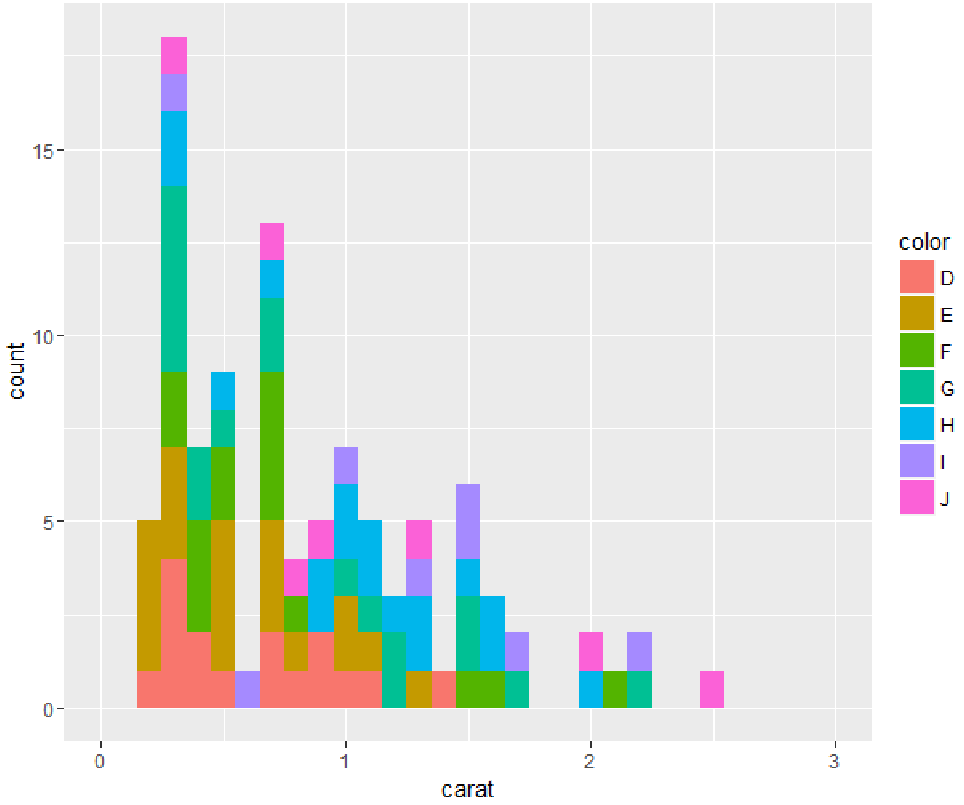
Adjust controls the degree of smoothness

COLOR CAN BE USED AS AN AESTHETIC MAPPING

Mapping a categorical variable to an aesthetic will automatically split up the geom by that variable. This example shows each level of diamond color.

qplot(carat,data=diamSMALL,geom="density",color=color, adjust=1)





qplot(carat, data=diamSMALL, geom="histogram", binwidth=.1, xlim=c(0,3), fill=color)

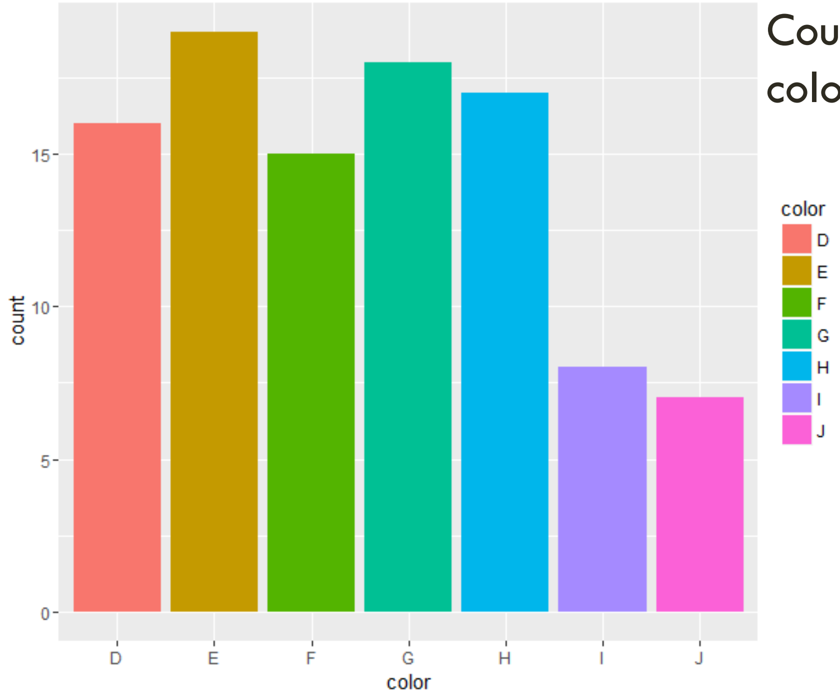
BAR CHARTS

1) A Bar Chart is a discrete analogue of a histogram.

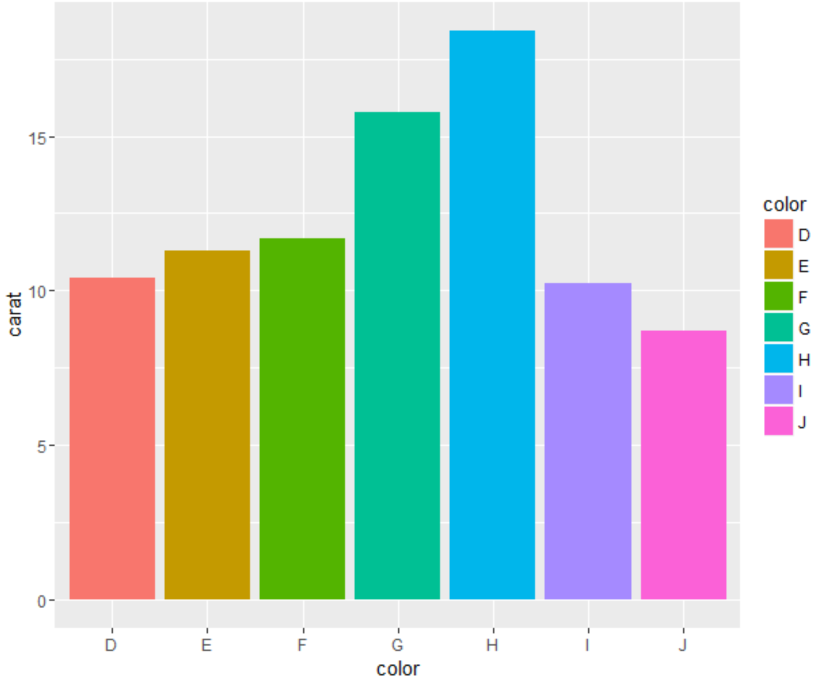
2) The standard bar chart will count up all instances of each category or class

3) It is also an option to choose a continuous variable via the "weight" option.

qplot(color, data=diamSMALL, geom="bar", fill=color)



Count of carats per color.



qplot(color, data=diamSMALL, geom="bar", weight=carat, fill=color)+scale_y_continuous("carat")

TIME SERIES

Time Series plots are often generated with **lines** or **paths.**

Lines join points from left to right. **Paths** join points in the **order they appear** in the dataset.

Therefore: line plot = sorted (by x) path plot

For **Time Series,** the x-axis commonly represents a measure of time – showing how one variable (say y) changes over time (say x).

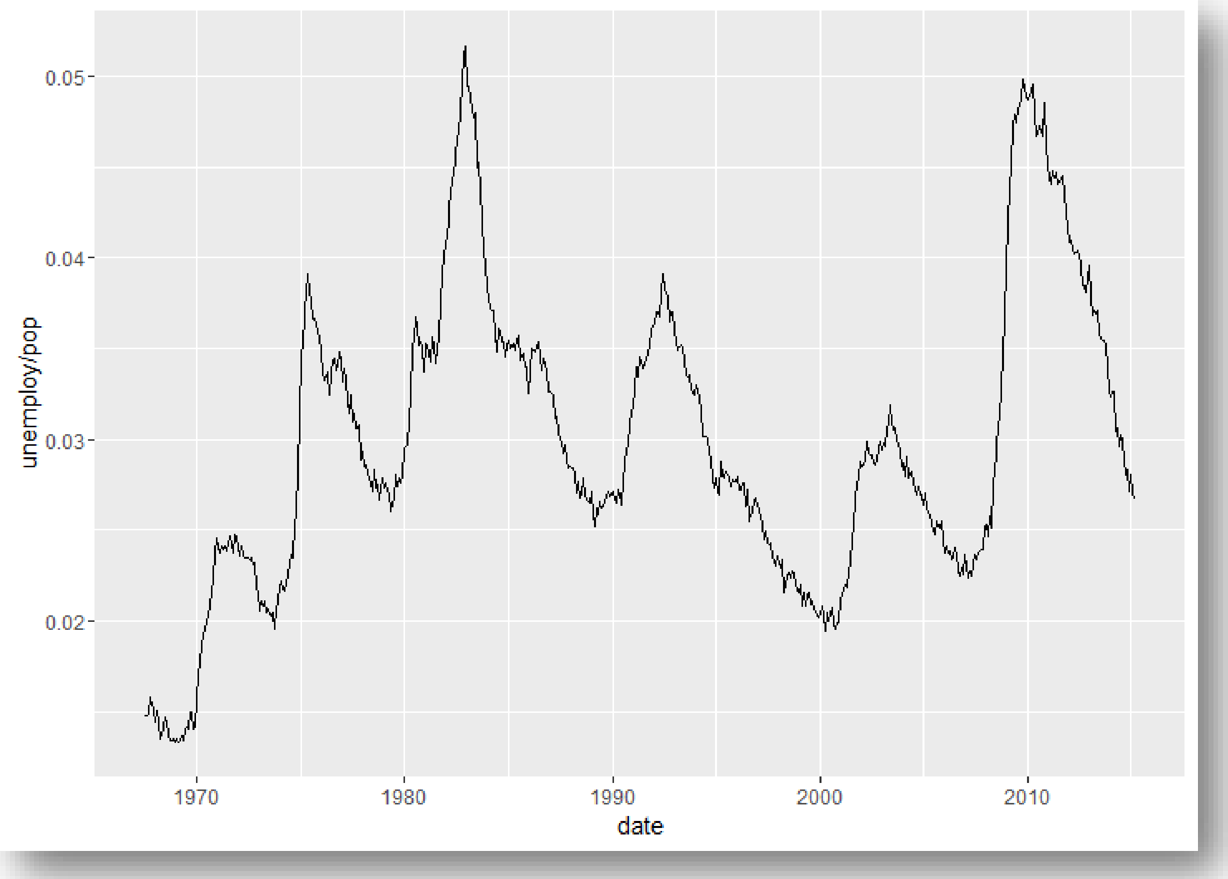
Alternatively, **Path Plots** show how two variables have simultaneously changed over time.

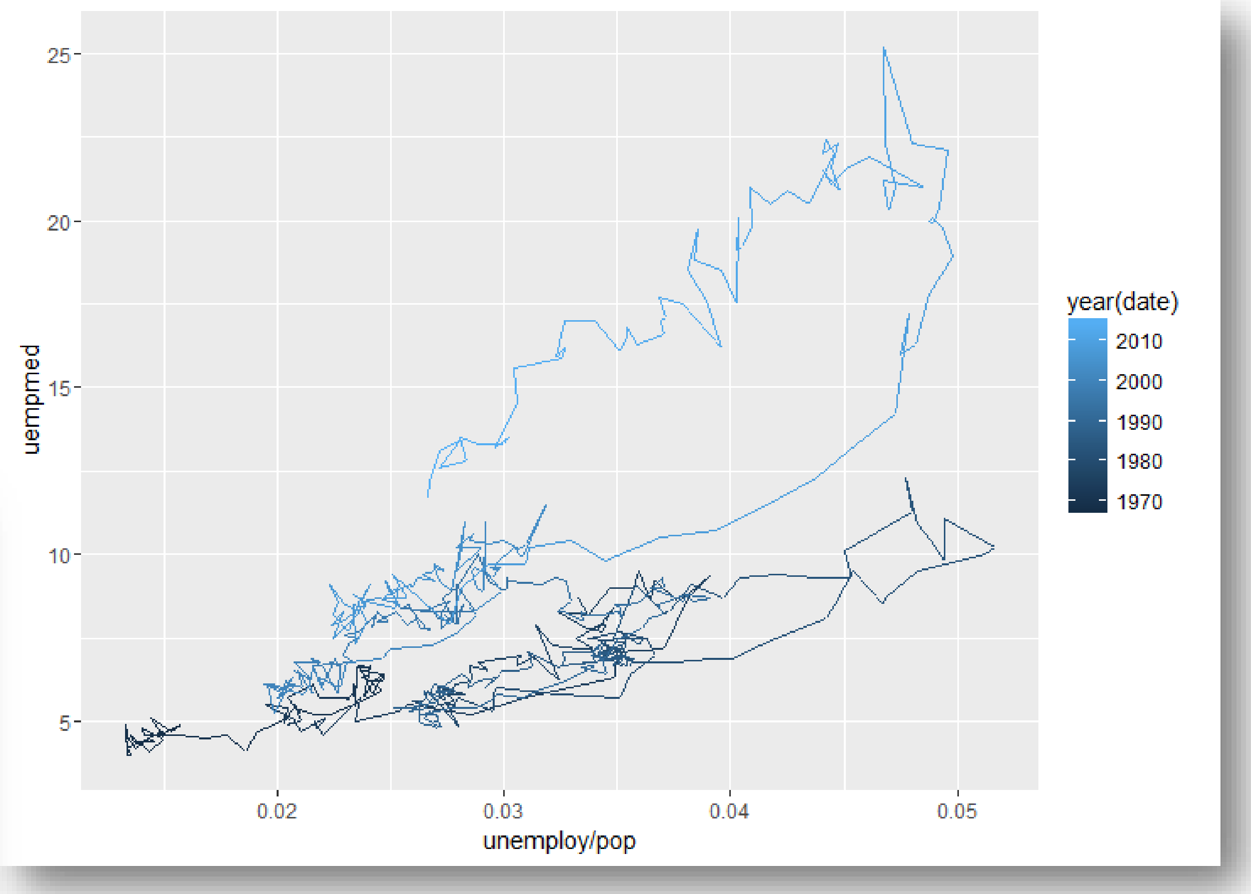
Dates and Times in R:

http://biostat.mc.vanderbilt.edu/wiki/pub/Main/ColeBeck/datestimes.pdf

LINES AND PATHS

```
#Ordered by data
qplot(date,unemploy/pop, data=economics, geom="line")
```





```
year <- function(x) as.POSIXlt(x)$year+1900
qplot(unemploy/pop, uempmed, data=economics,
geom="path", color=year(date))
# unemploy, number of unemployed in thousands,
#uempmed, median duration of unemployment, in week,
#pop, total population, in thousands
```

POSIX AND DATA-TIME IN R

Note:

```
(as.POSIXlt("1967-07-01")$year)
```

prints 67

RE:

https://stat.ethz.ch/R-manual/R-devel/library/base/html/as.POSIXlt.html

http://biostat.mc.vanderbilt.edu/wiki/pub/Main/ColeBeck/datestimes.pdf

https://www.stat.berkeley.edu/~s133/dates.html

FACETING

Thus far, we have looked at both “shape” and “color” as aesthetics.

Faceting splits data into subsets or subgroups and then displays each group.

Faceting Formula:

row_var ~ col_var

Note that the use of the dot will assume ONE row or column:

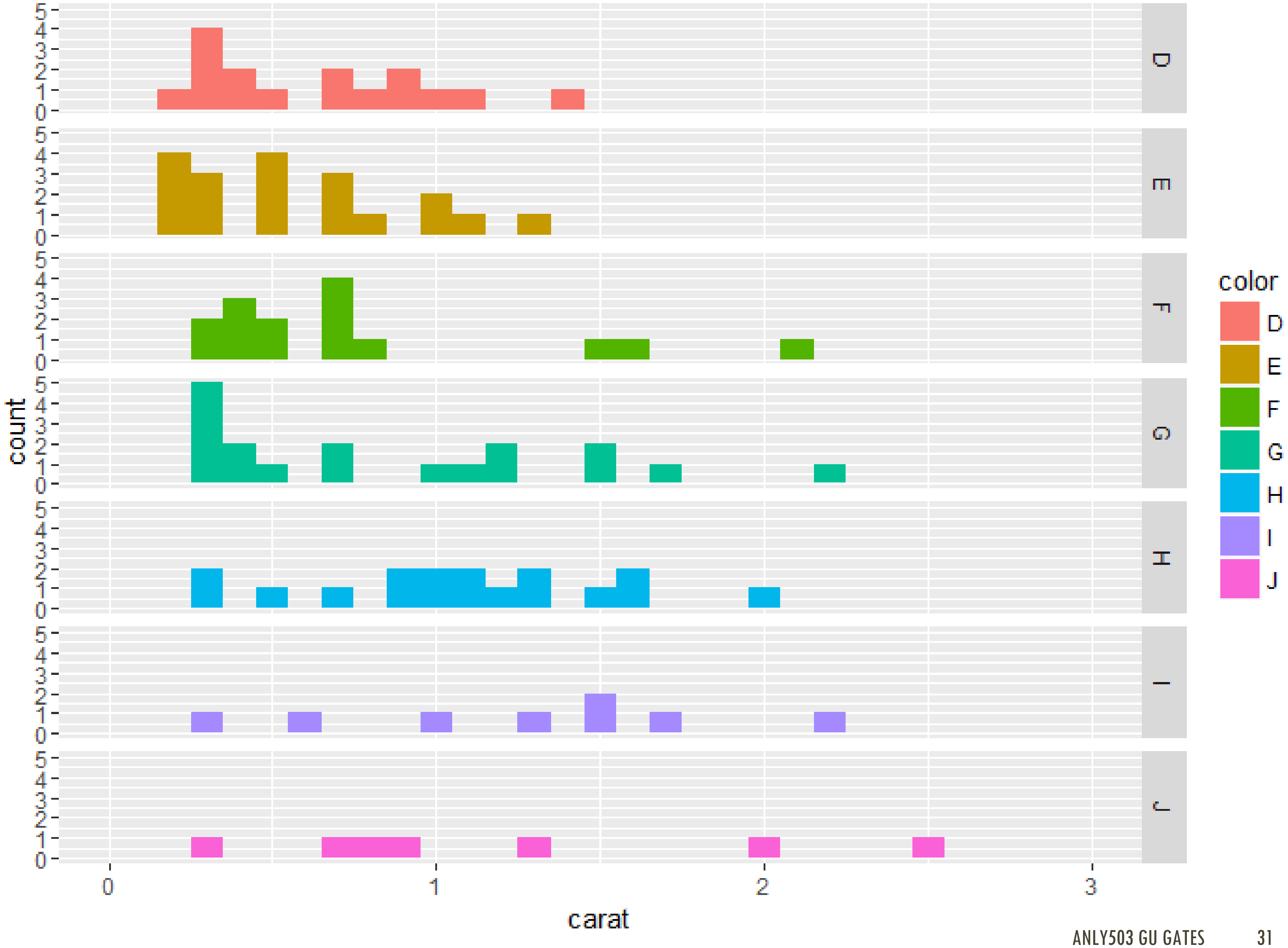
row_var ~ .

# multiple rows and one column.

## FACETING QPLOT(CARAT, DATA=DIAMSMALL, **FACETS= COLOR~.** , GEOM="HISTOGRAM", BINWIDTH=.1, XLIM=C(0,3), FILL=COLOR)

RE:
https://plot.ly/ggplot2/facet/

http://www.cookbook-r.com/Graphs/Facets_(ggplot2)/



LAYERING THEORY

Gates

LAYERING WITH GGPLOT

Layering: Building a plot in stages by adding different elements or "Layers".

Each layer can come from a different dataset and can have its own aesthetic mapping.

Note that using **qplot() automates** many of the layerings – such as the plot object, the displaying of the result, and default values.

ggplot() will permit full use and control over plot layers.

GGPLOT( ) AND EXPLICIT LAYERING

Quick Form:

p <- ggplot(dataset, aes(x=, y=, color=) + geom_XXX(mapping, dataset, …, geom, position)

or

p <- ggplot(dataset, aes(x=, y=, color=) + stat_XXX(mapping, dataset, …, stat, position)

geom_XXX can be geom_histogram or geom_bar or geom_line, etc.

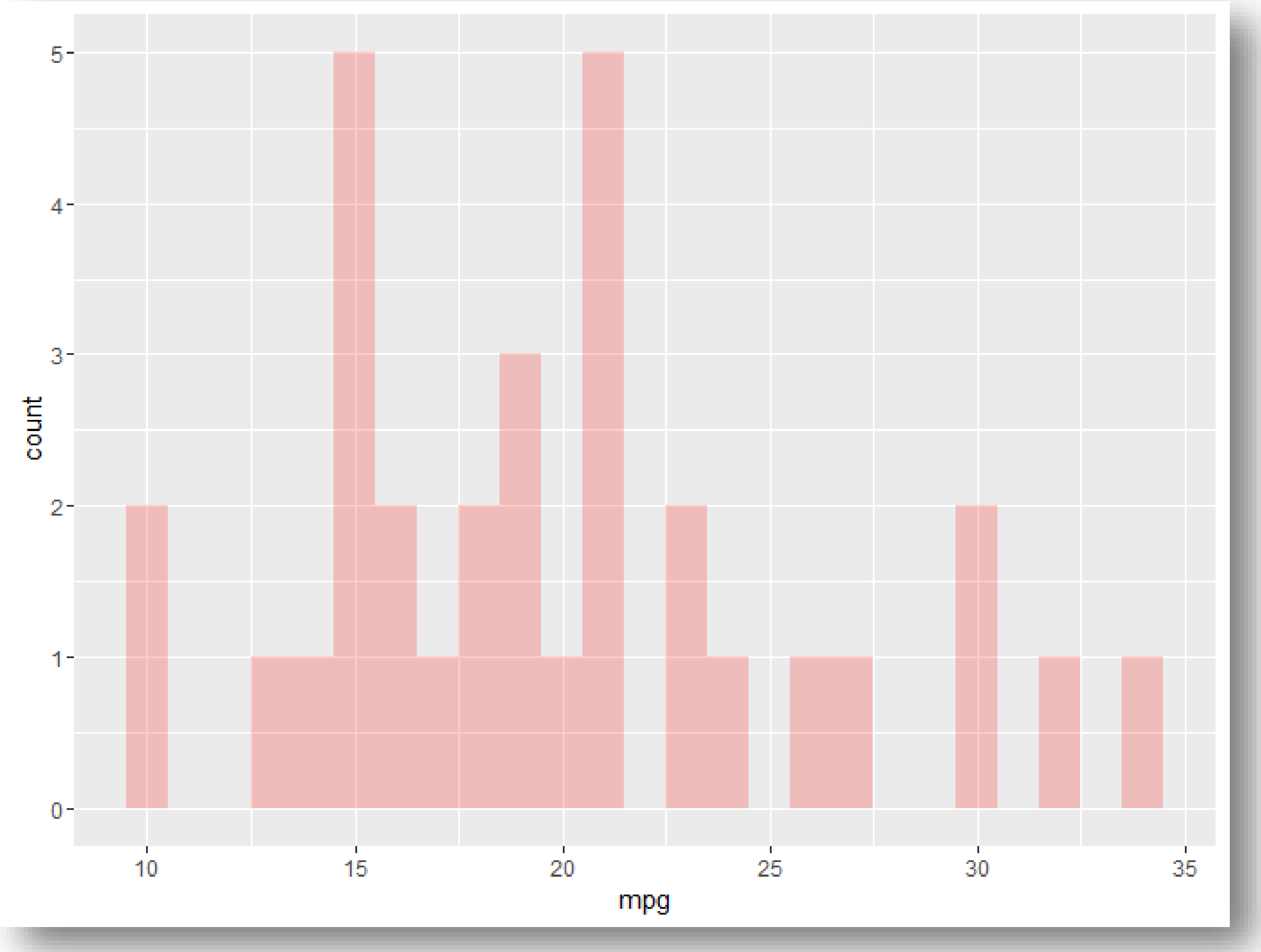
Note that every geom is associated with a default stat and every stat is associated with a default geom. Therefore, it is only necessary to define either stat or geom (unless you want to control both outside of the defaults.

For full control, the syntax is:

p <-   ggplot(dataset, aes(..))   #first define plot p, then add layers to p with "+"

p <-   p + layer (geom=, geom_params=list(), stat=, stat_params=list())

```
p <- ggplot(mtcars, aes(mpg)) + geom_histogram(binwidth=1, fill=alpha("red", .2))
p
```

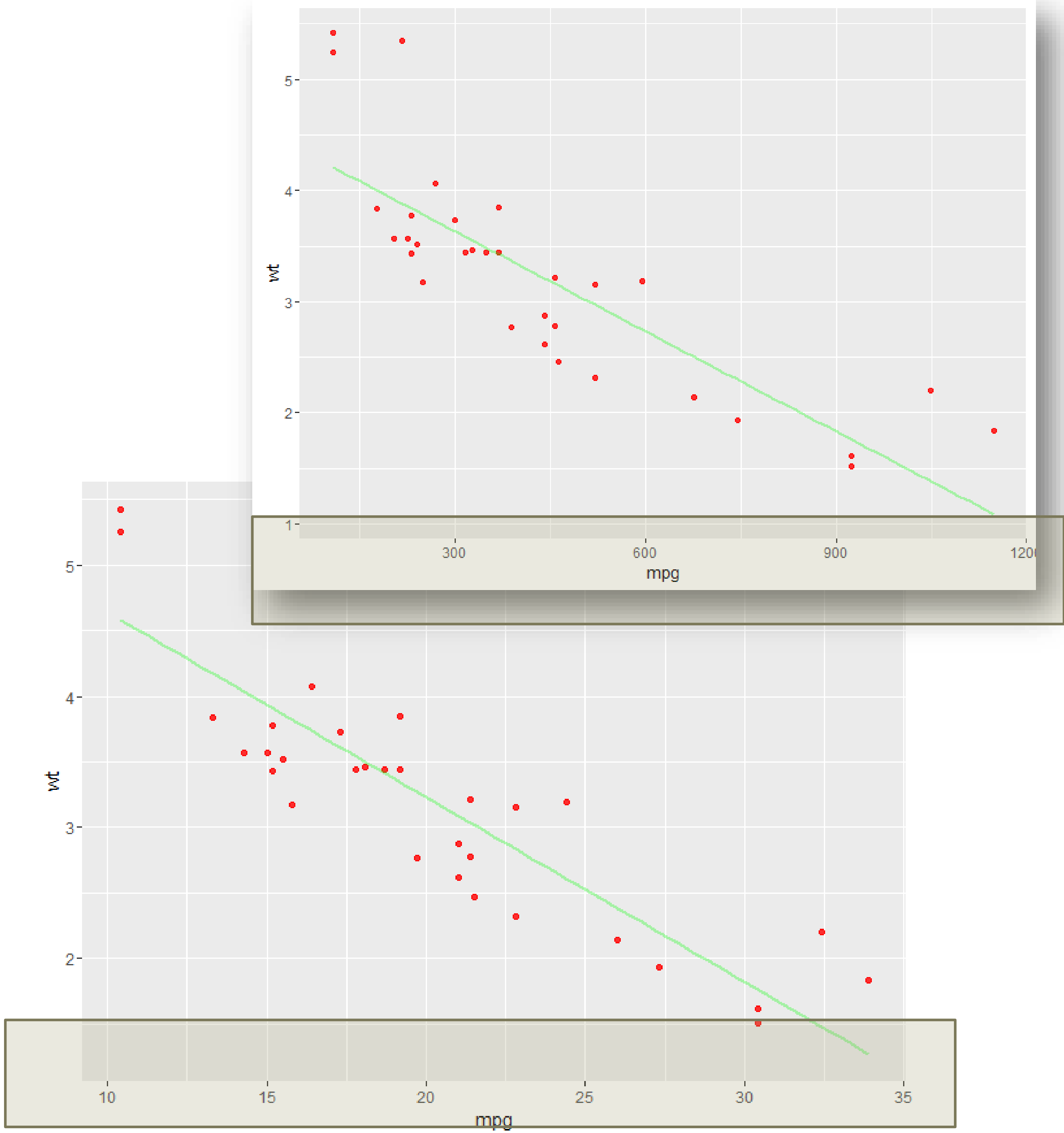


TRANSFORMATIONS %+%

```
bestfit <- geom_smooth(method="lm", se=F,
color=alpha("green", .3), size=1)

p <- ggplot(mtcars, aes(x=mpg, y=wt)) +
geom_point(color=alpha("red", .8))

+ bestfit

p   #plot p

newcars<- transform(mtcars, mpg=mpg^2)

p %+% newcars   # layer on the transformation to p

p    #plot p
```

NOTE: %+% is an infix function such that

```
`%+%` <- function(a, b) paste0(a, b)

"new" %+% " string"

## Results is… "new string“

# With ggplot, %+% replaces the current dataframe
```



AESTHETIC MAPPINGS

Aesthetics: things we can see in a plot

Mapping: methods for displaying aesthetics.

Aesthetic mappings can be default or modified using the “+”.

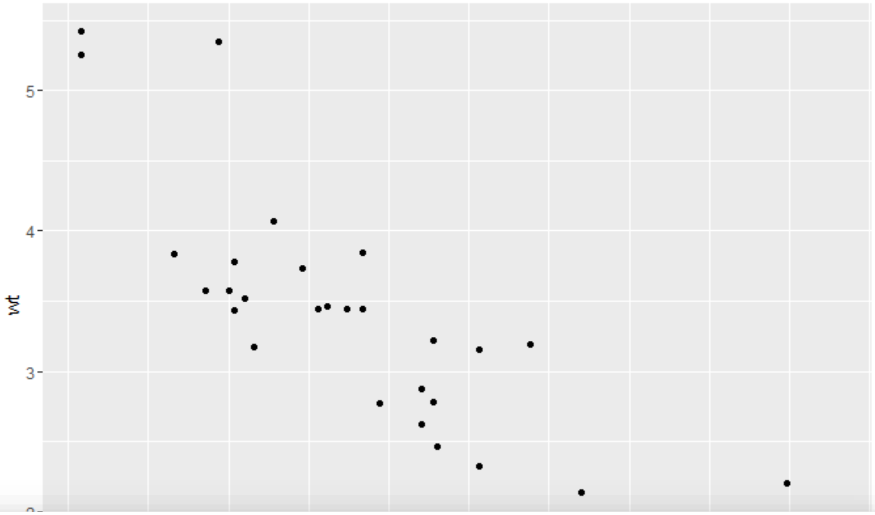
Example

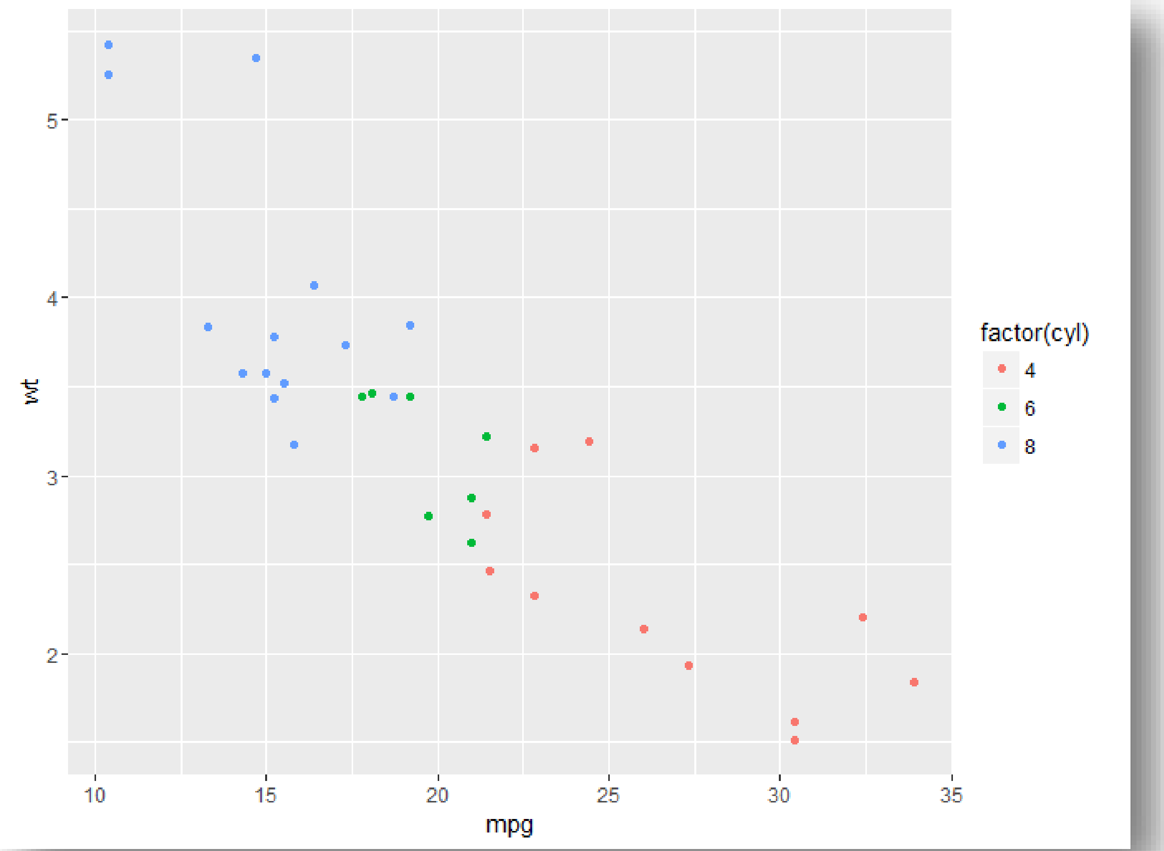
p <- ggplot(mtcars, aes(x=mpg, y=wt))

p + geom_point()

OR

p + geom_point(aes(color=factor(cyl)))





GROUPING

Geoms can be “individual” or “collective”.

1) **Individual**: has a distinct graphical object (such as a point) for each row of data

2) **Collective**: graphical object that represents multiple rows of data (such as a bar)

Cases:

Multiple groups with one aesthetic

Different groups on different layers

Override default grouping

DATA FOR FOLLOWING EXAMPLES: OXBOYS

install.packages("nlme")

# restart R Studio

#nlme: linear/nonlin /mixed models

library(nlme)

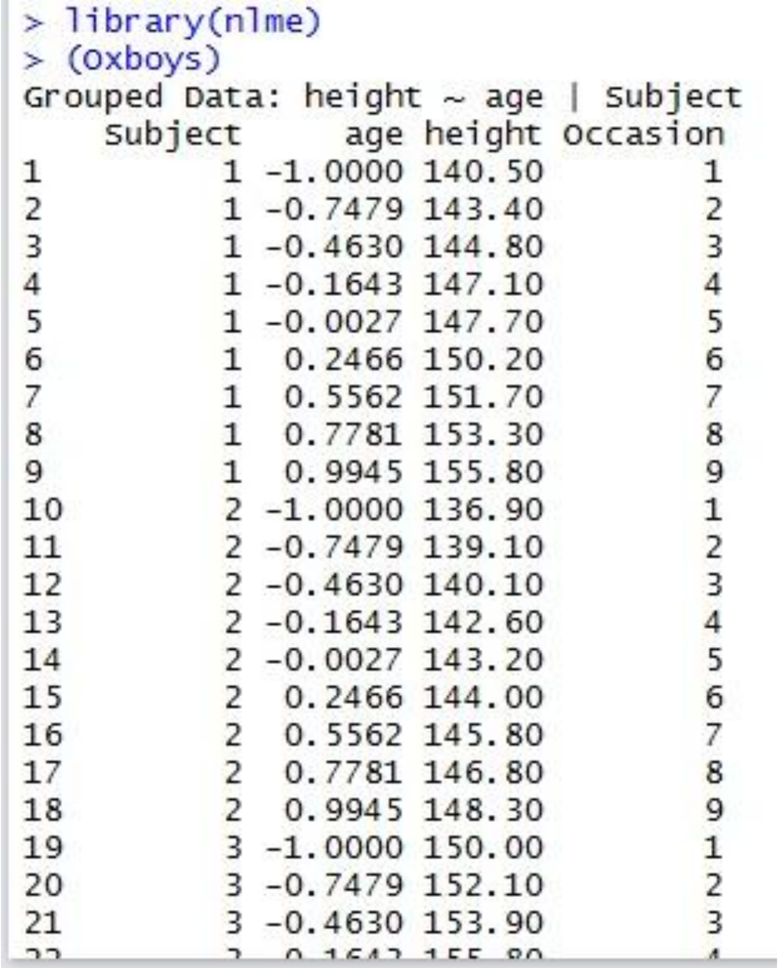
(head(Oxboys))

#Occasion is when measure collected

#This is longitudinal data

# age is standardized

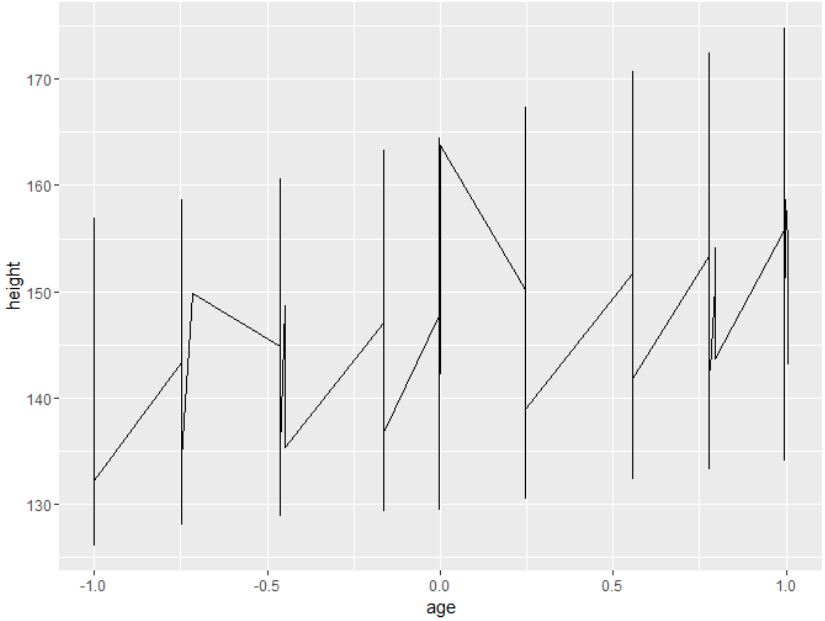
https://stat.ethz.ch/R-manual/R-devel/library/nlme/html/Oxboys.html

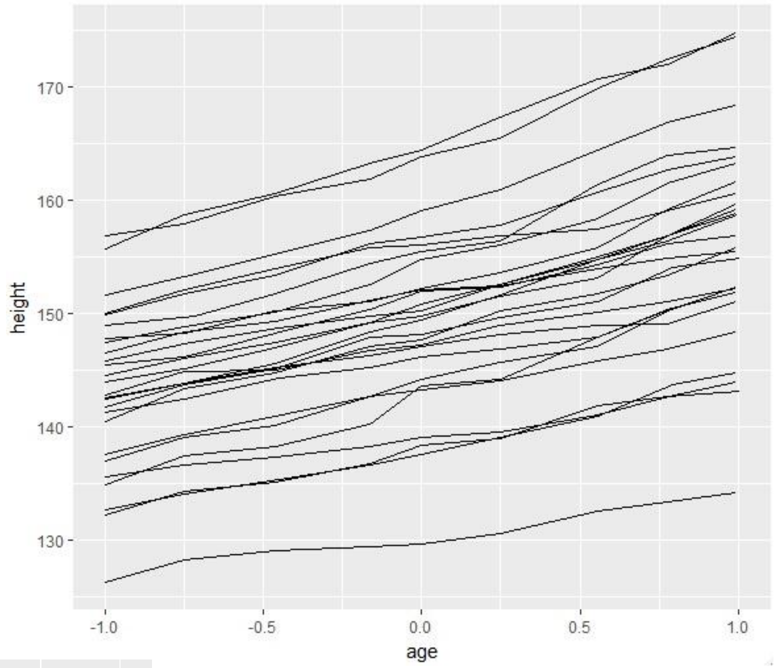


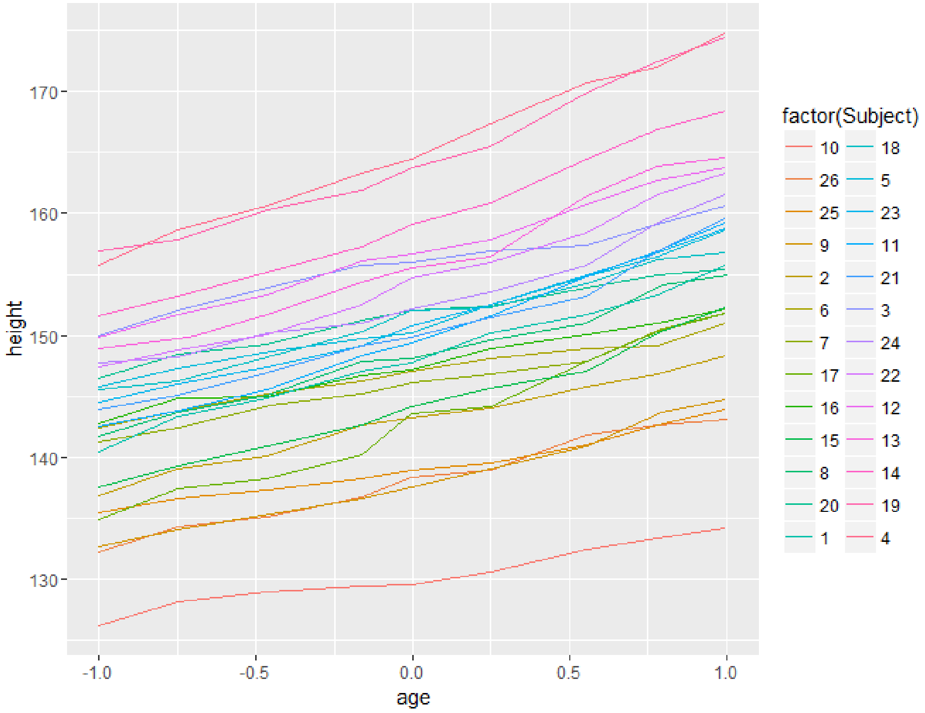
```
> library(nlme)
> (Oxboys)
Grouped Data: height ~ age | Subject
    Subject     age height Occasion
1         1 -1.0000 140.50        1
2         1 -0.7479 143.40        2
3         1 -0.4630 144.80        3
4         1 -0.1643 147.10        4
5         1 -0.0027 147.70        5
6         1  0.2466 150.20        6
7         1  0.5562 151.70        7
8         1  0.7781 153.30        8
9         1  0.9945 155.80        9
10        2 -1.0000 136.90        1
11        2 -0.7479 139.10        2
12        2 -0.4630 140.10        3
13        2 -0.1643 142.60        4
14        2 -0.0027 143.20        5
15        2  0.2466 144.00        6
16        2  0.5562 145.80        7
17        2  0.7781 146.80        8
18        2  0.9945 148.30        9
19        3 -1.0000 150.00        1
20        3 -0.7479 152.10        2
21        3 -0.4630 153.90        3
```

LAYERS

```
ggplot(Oxboys, aes(age, height)) + geom_line()  #Figure 1

ggplot(Oxboys, aes(age, height, group = Subject)) + geom_line() #Figure 2

ggplot(Oxboys, aes(age, height, group = Subject, color=factor(Subject))) + geom_line()   #Figure 3
```


height
170
160
150
140
130
-1.0
-0.5
0.0
0.5
1.0
age

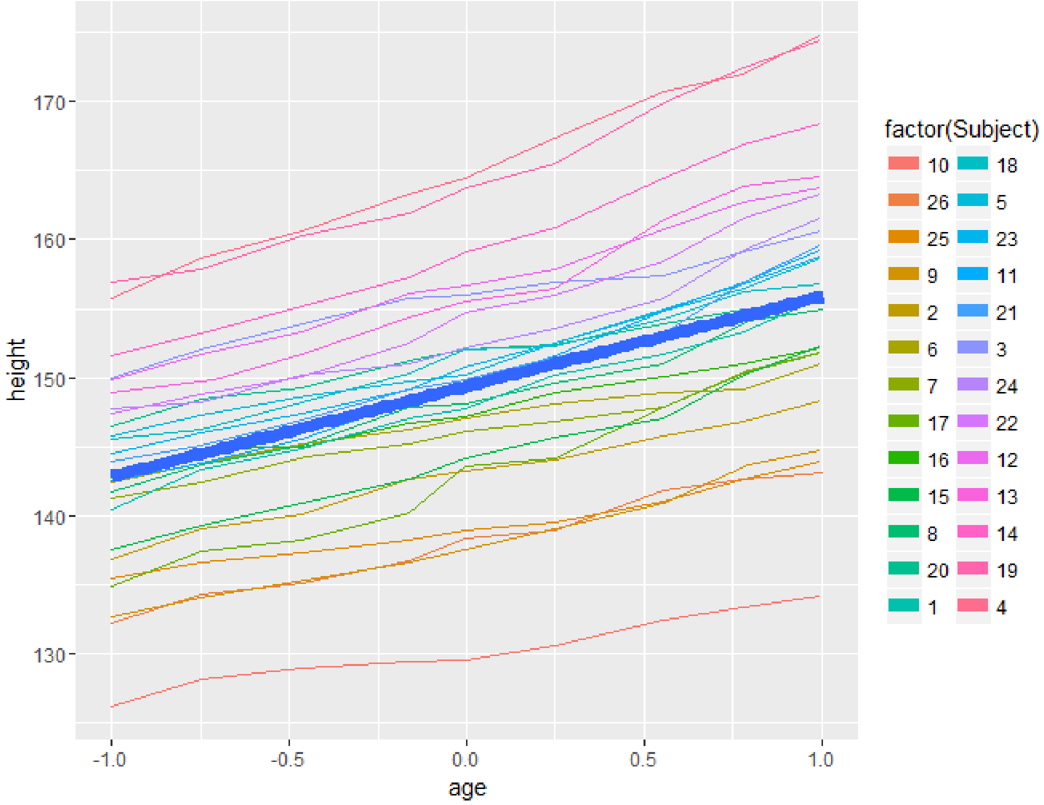

height
170
160
150
140
130
-1.0
-0.5
0.0
0.5
1.0
age


height
170
160
150
140
130
-1.0
-0.5
0.0
0.5
1.0
age
factor(Subject)
10
26
25
9
2
6
7
17
16
15
8
20
1
18
5
23
11
21
3
24
22
12
13
14
19
4

DIFFERENT GROUPS ON DIFFERENT LAYERS

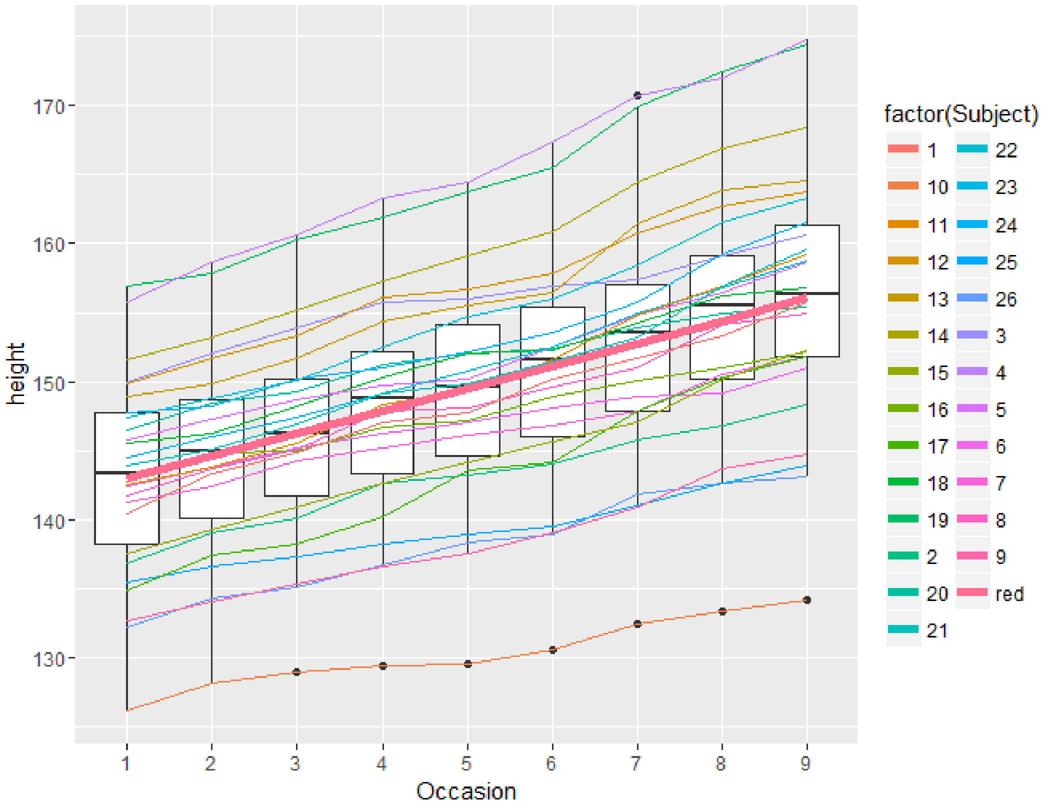
Here, we will add a single smooth line to the plot that represents AGE and HEIGHT for ALL boys.

```
ggplot(Oxboys, aes(age, height)) + geom_line()

p <- ggplot(Oxboys, aes(age, height, group =
Subject, color=factor(Subject))) + geom_line()

#Create a best line of fit for all boys
# add onto p the smooth lm line
p <- p + geom_smooth(aes(group=1), method="lm",
se=F, size=3)


p    #plot p
```

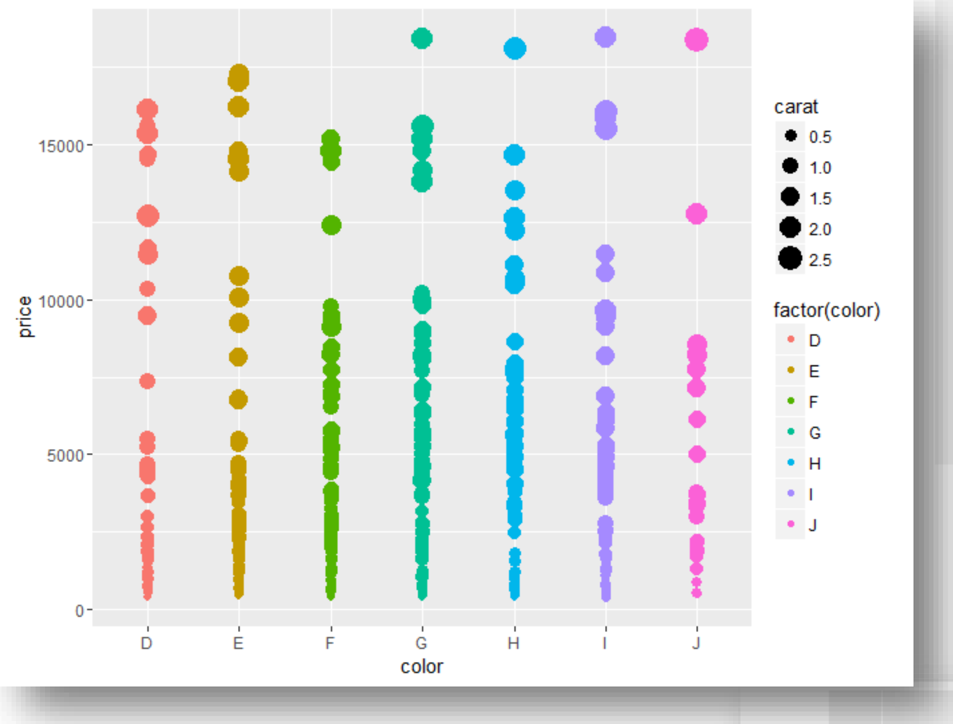


OVERRIDE DEFAULT GROUPING

```
##Overriding default grouping

#No need to specify group because "Occasion" is discrete

boysbox1 <- ggplot(Oxboys, aes(Occasion,
height))+geom_boxplot()

#To overlay individual trajectories, override the default
#grouping for that layer

boysbox2 <- boysbox1 + geom_line(aes(group=Subject,
color=factor(Subject)))

#Add a layer that is a smooth

boysbox3 <-boysbox2 + geom_smooth(aes(group=1,
color="red"), method="lm", se=F, size=2)

boysbox3   # plot it
```



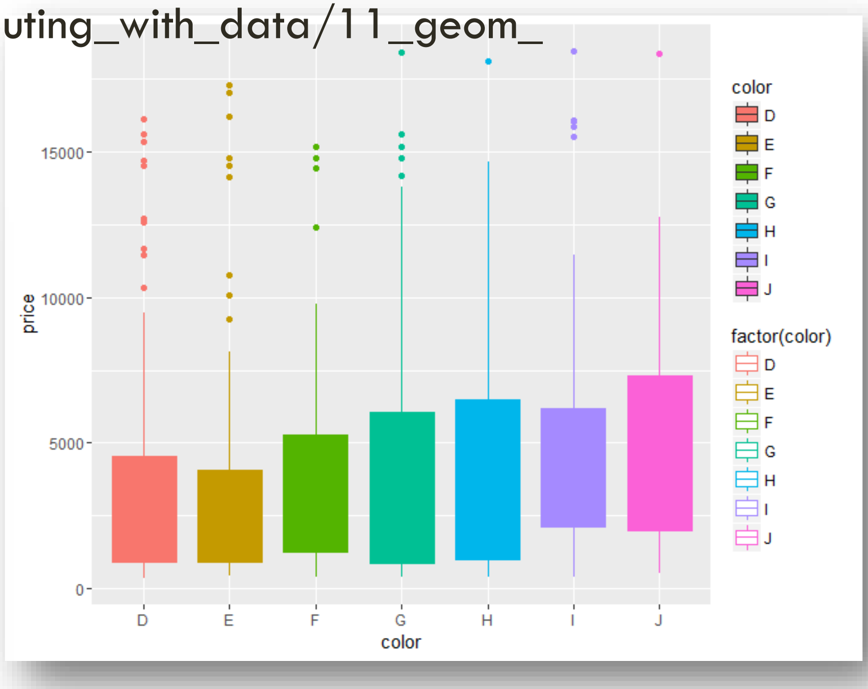
Occasion
an ordered factor - the result of converting age from a continuous variable to a count so these slightly unbalanced data can be analyzed as balanced. Age is normalized -1to1



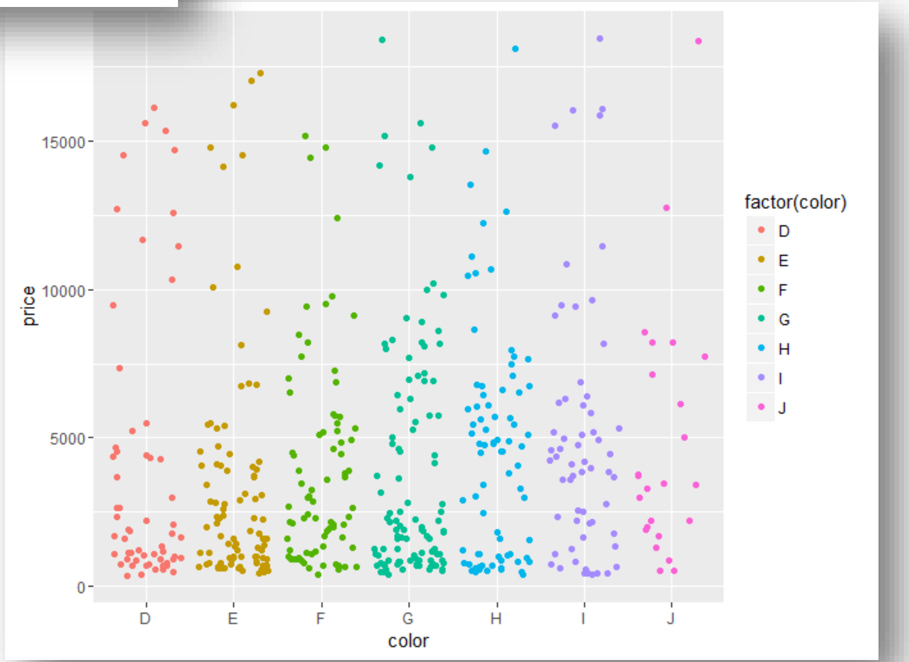
#Geoms – geometric objects – control plots&layers

```
diamMEDIUM <- diamonds[sample(nrow(diamonds),500), ]
baseplot <- ggplot(data = diamMEDIUM, aes(x = color, y = price,
color=factor(color)))
## Three individual plots….
baseplot + geom_point(aes(size=carat))
baseplot + geom_jitter()
baseplot + geom_boxplot(aes(fill=color))
```

https://www3.nd.edu/~steve/computing_with_data/11_geom_examples/ggplot_examples.html



See pages 56&57 in Wickham for Tables 4.2 and 4.3 that include lists of Geoms and Default Stats and Aesthetics



STAT – A STATISTICAL TRANSFORMATION STAT_XXX( )

1) Transforms the data.

Example: a **Smoother** will find the mean of y for each x.

2) A “Stat” must be location and scale invariant so that:

$f(x + a) = f(x) + a$ AND

$f(bx) = b\,f(x)$

set	set + 5	set* 10
30	35	300
40	45	400
50	55	500
60	65	600
70	75	700
80	85	800
90	95	900
mean	mean	mean
60	65	600
	which is 60 + 5	which is 60 * 10

STAT

A “Stat” takes a dataset as input and returns a dataset as output

A “Stat” can add/generate variables and then aesthetics can be mapped to the new variables.

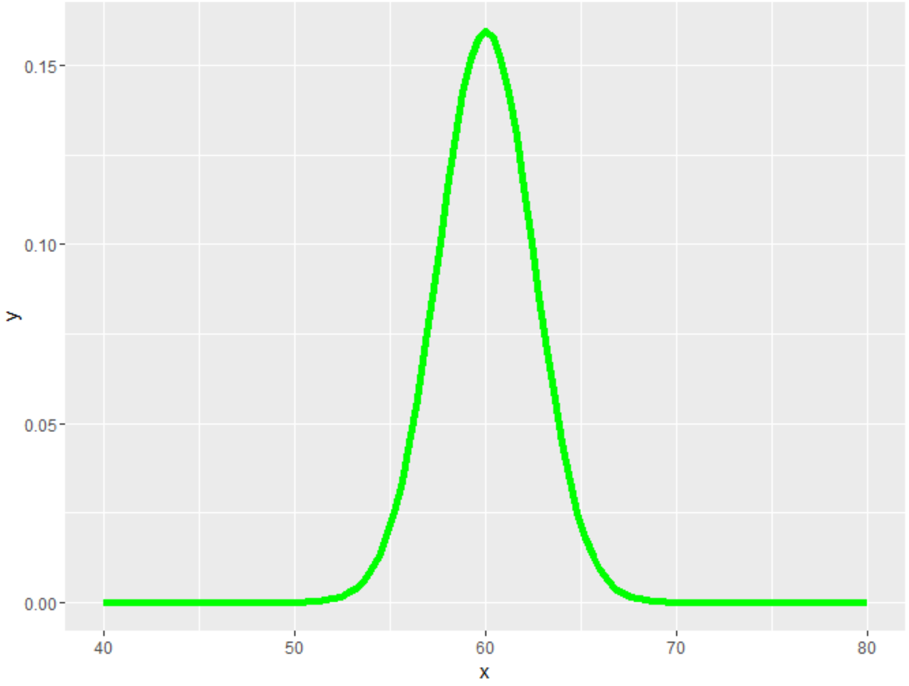
Example: “stat_bin”

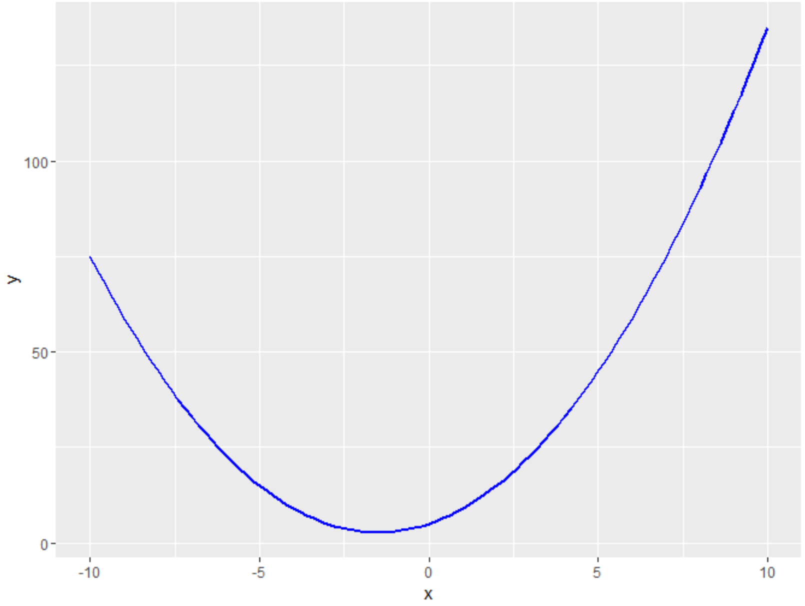
- used to make histograms
- creates variables: count, density, and x – the center of each bin
- these variables can be referenced using the “.. variable name ..”

```
ggplot(diamonds, aes(carat)) + geom_histogram(aes(y = ..count..), binwidth=.5, fill="blue")
```

STAT EXAMPLES

```
#STATS

# Plot a normal curve

base1 <- ggplot(  data.frame(x = c(40, 80)), aes(x))

base1 + stat_function(fun = dnorm, args = list(mean = 60, sd = 2.5), color="green", size=2)


# Using a custom function

base2 <- ggplot(data.frame(x = c(-10, 10)), aes(x))

newf <- function(x) {x ^ 2 + 3*x + 5}

base2 + stat_function(fun = newf, color="blue", size=1)
```





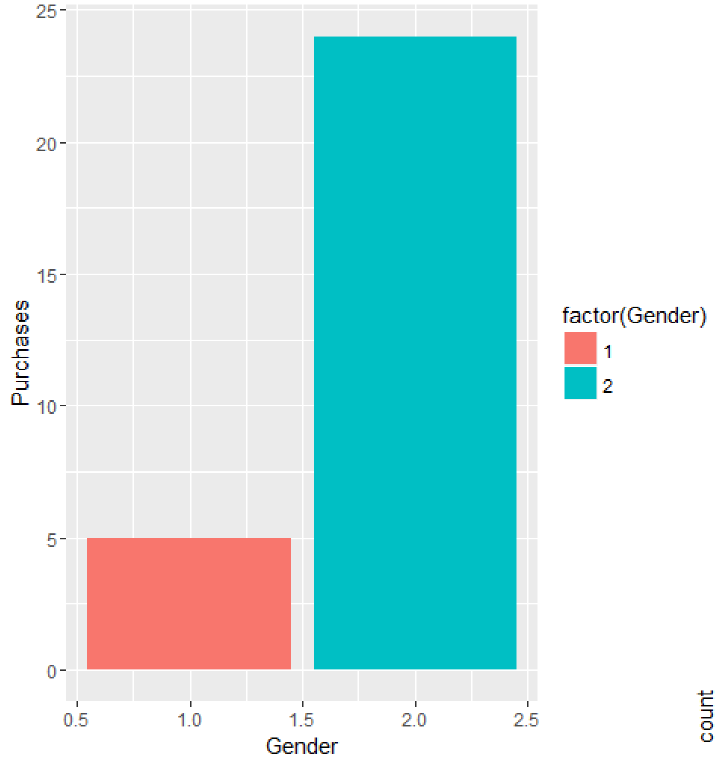
BASIC PLOTS AND IDENTITY

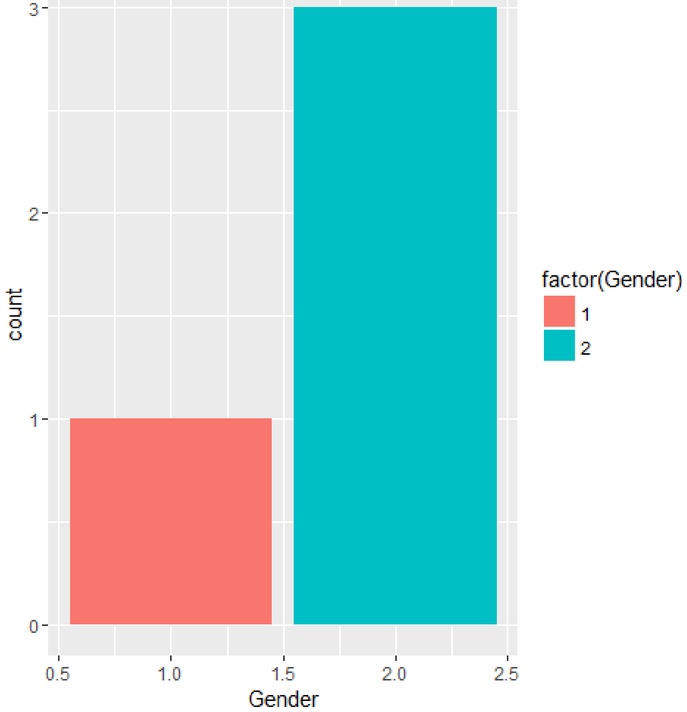
geom_bar using and not using “identity”

```
##geom area
name=c("Bob", "Sally", "Jan", "Annie")
Age=c(34,21,56,54)
Gender=c(1, 2, 2, 2)
Purchases=c(5, 10, 11, 3)
df=data.frame(name, Age, Gender)
(df)
```

# Using identity, x is the center of the bar and y is the height so its 2D

```
myplot <- ggplot(df, aes(x=Gender, y=Purchases, fill=factor(Gender)))
myplot + geom_bar(stat="identity")
myplot2 <- ggplot(df, aes(x=Gender, fill=factor(Gender))) + geom_bar()
myplot2
```





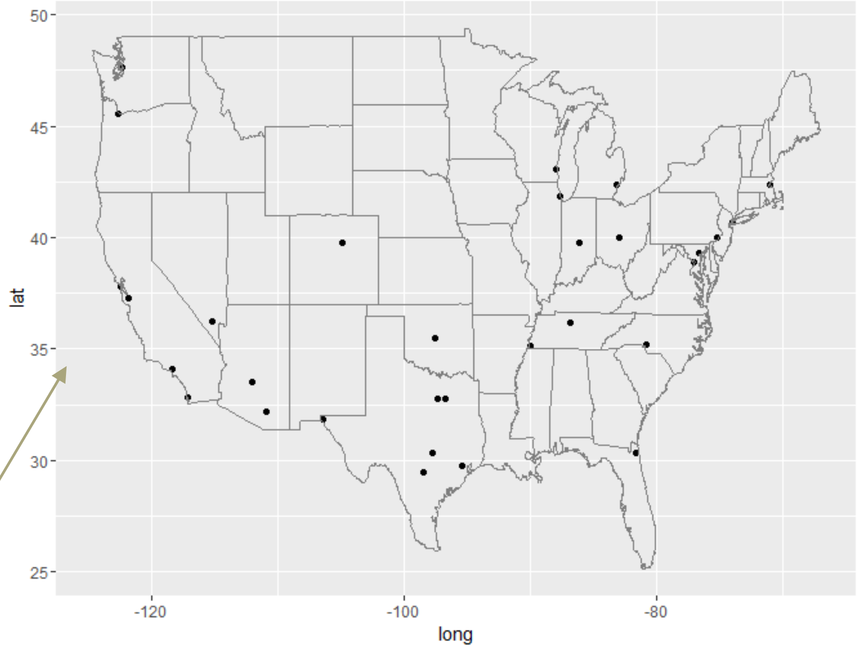
SURFACE AND MAPS: DRAWING MAPS

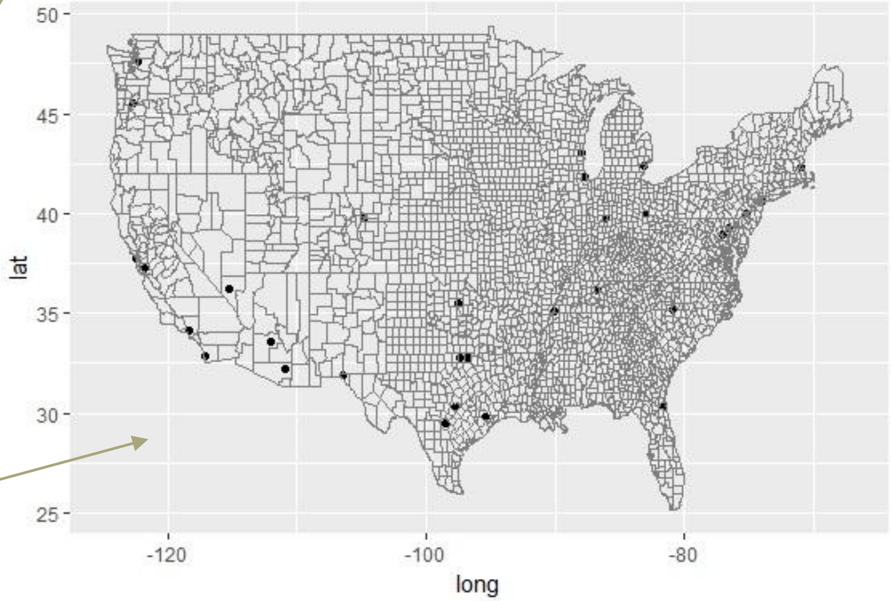
The MAPS available in the maps package are:

Map of	Map Package Name
France	france
Italy	Italy
New Zealand	nz
USA County	county
USA state	state
USA borders	usa
Entire World	world

SURFACE AND MAPS: DRAWING MAPS

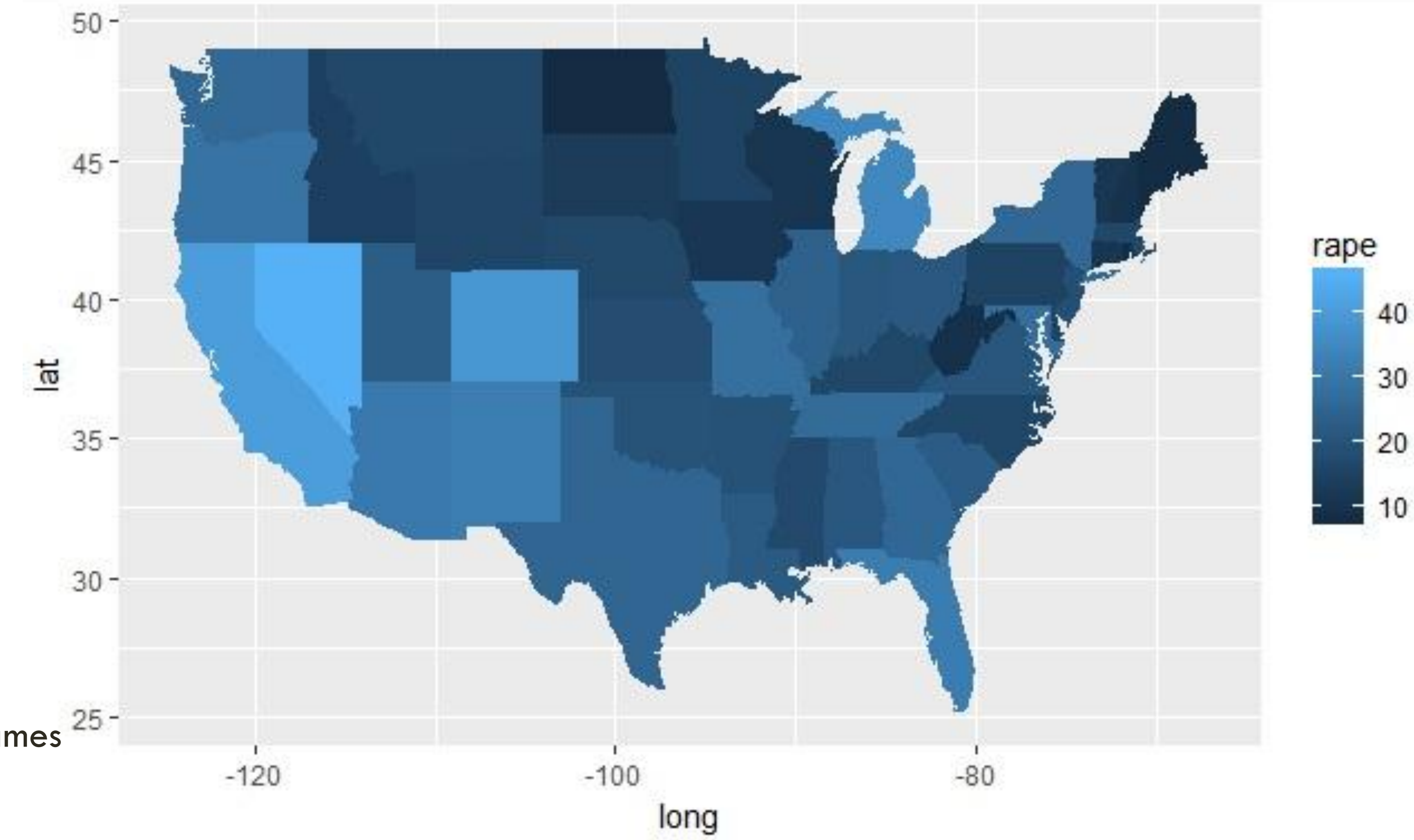
```
##MAPS

## On the command line:

## install.packages("maps")

#Spatial

library(maps)

data(us.cities)

big_cities <- subset(us.cities, pop > 500000)

qplot(long,lat, data=big_cities) + borders("state")

#OR

qplot(long,lat, data=big_cities) + borders(“county")
```





SURFACE AND MAPS: DRAWING MAPS

```
##Choropleth
## Use map_data() to convert map to data frame
## dataframe can be merge() with data
library(maps)
states <- map_data("state")
arrests <- USArrests
names(arrests) <- tolower(names(arrests)) # columns names
#tolower – convert to lower case
arrests$region <- tolower(rownames(USArrests))
choro <- merge(states, arrests, by="region")
#Reorder rows
choro <- choro[order(choro$order),]  # choro has a feature called order
#qplot(long, lat, data=choro, group=group, fill=assault, geom="polygon")
#group signifies the state
qplot(long,lat,data=choro, group=group,fill=rape, geom="polygon")
```



POSITIONING AND FACETING

Four components that control position:

1) position adjustment – Section 4.8

2) position scales – Section 6.4.2

3) faceting - automatically layering out multiple plots (small multiples) on a page

4) coordinate systems – most common is Cartesian – there is also polar, map, etc.

FACETING

Two types of facets in ggplot

1) facet_grid – produces a 2D panel (matrix) of subplots

2) facet_wrap – produces a 1D ribbon of subplots that “wrap” to form a 2D matrix

facet_grid

A1	**B1**	**C1**
A2	**B2**	**C2**
A3	**B3**	**C3**

facet_wrap

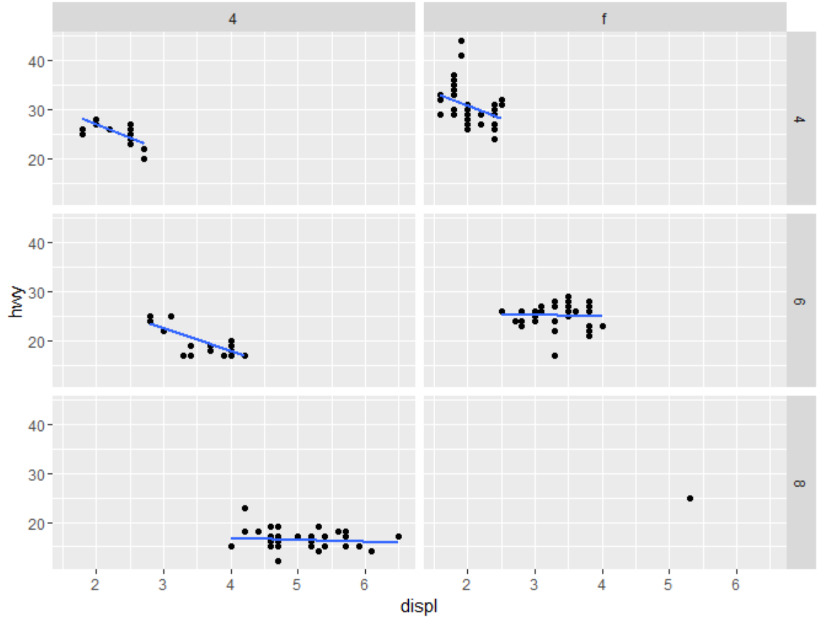
1	**2**	**3**
4	**5**	**6**
7	**8**	**9**

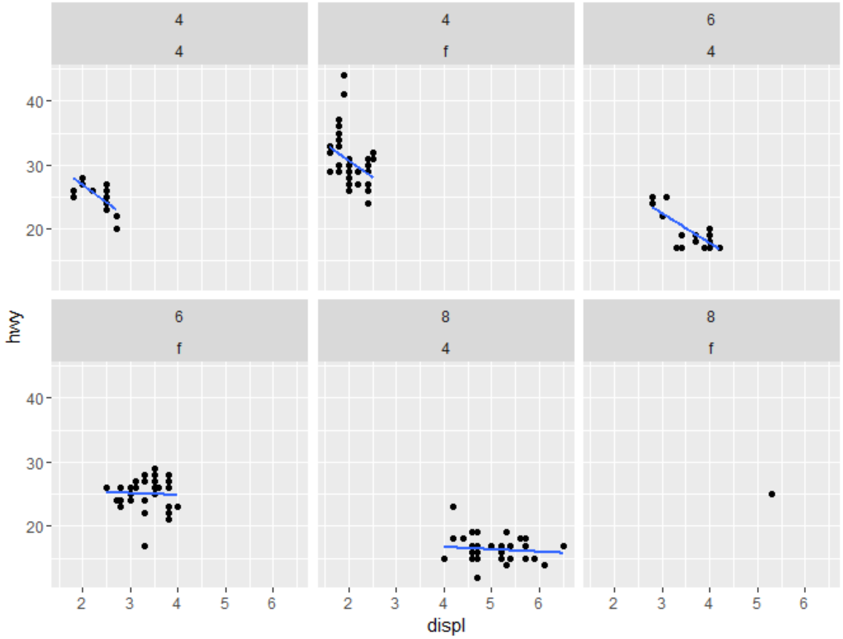
FACET EXAMPLES

```
## Facet Grid & Wrap

(mpg2 <- subset(mpg, cyl!=5 & drv %in%
c("4", "f")))

p <- qplot(displ,hwy,data=mpg2) +
geom_smooth(method="lm", se=F)

p + facet_grid(cyl ~ drv)

p + facet_wrap(cyl ~ drv)
```

NOTE: **mpg** is a dataset epa. It contains variables such as cyl (cylinders on the car) and drv (“4” is 4 wheel drive, “f” is “front wheel drive”)

NOTE2: “%in%” is a binary operator to check if element is in vector.





DATA TRANSFORMATION

Common data transformations:

1) Normalize the data: $xi / \sum xi$

2) Take the log of the data: $\log xi$

3) Take the power of the data: $xi \wedge (1/c)$ where c is a constant

4) Use the Box-Cox on the data: $(xi^{\wedge}c – 1) / c$ where c is not 0 and is constant

5) Binning data – such as for a histogram

6) Grouping data – such as merging/ remapping

SCALE & COLOR

We will return to these topics –

but be sure to review the R book (Wickham Chapters 6 & 7)

DISTRIBUTIONS

1) One-dim continuous: histogram

- Important attribute: binwidth

2) Compare distributions between groups:

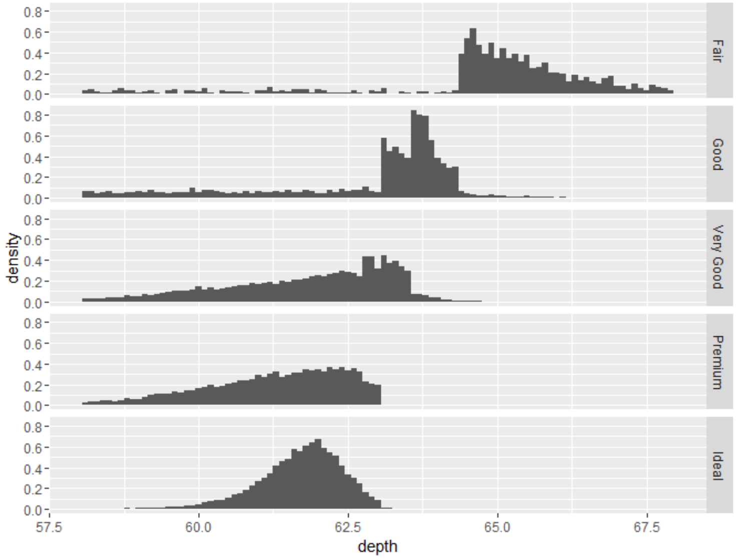
- create multiples of the histogram with **facets = .~ var**
- use a frequency polygon **goem="freqpoly"**
- create a conditional density plot with **position = "fill"**

Examples

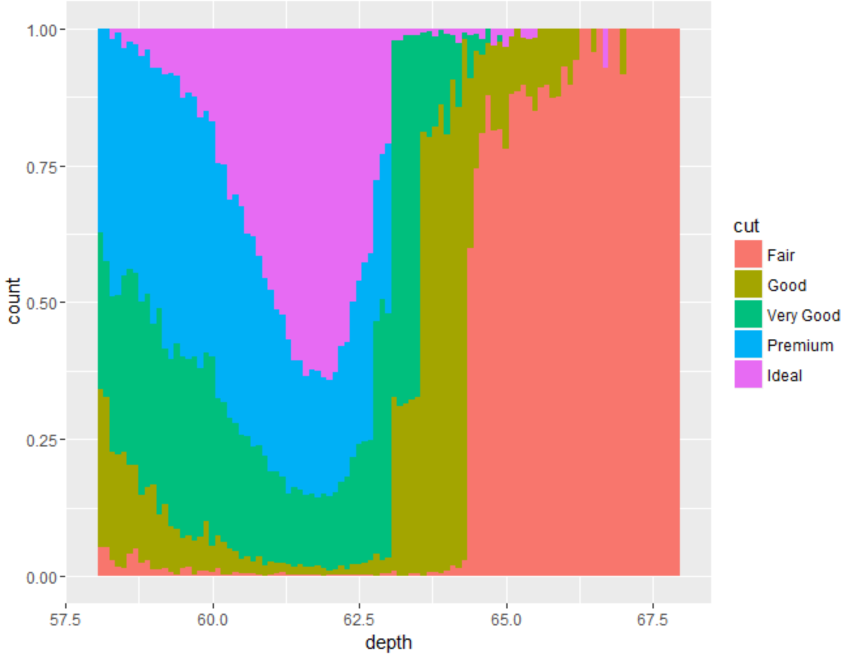
```
#density

#examples with diamonds depth

diam_depth_dist <- ggplot(diamonds, aes(depth)) + xlim(58,68)

# Recall: The names of "generated" variables must use the ..var..

diam_depth_dist + geom_histogram(aes(y=..density..), binwidth=.1) + facet_grid(cut ~ .)

diam_depth_dist + geom_histogram(aes(fill=cut), binwidth=.1, position="fill")

diam_depth_dist + geom_freqpoly(aes(y=..density.., color=cut), binwidth=.1)
```

USING POSITION AND TRANSPARENCY

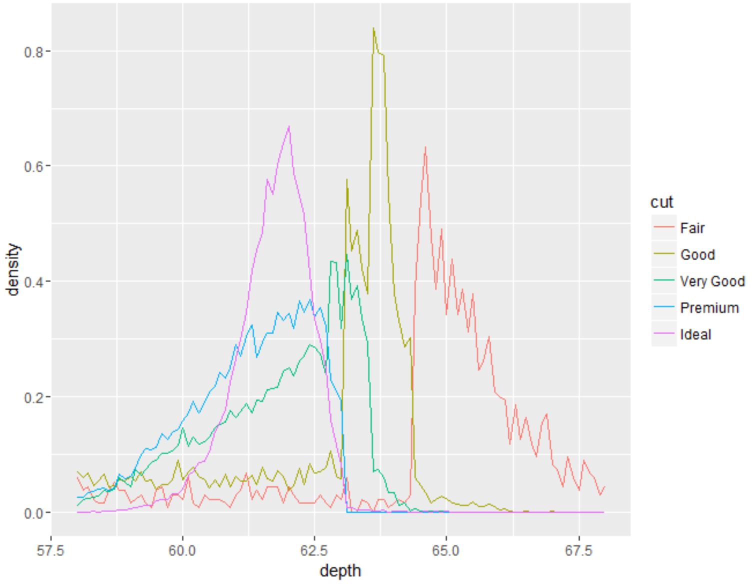
diam_depth_dist <- ggplot(diamonds, aes(depth)) + xlim(58,68)   # This is the base



diam_depth_dist + geom_histogram(aes(y=..density..), binwidth=.1) + facet_grid(cut ~ .)

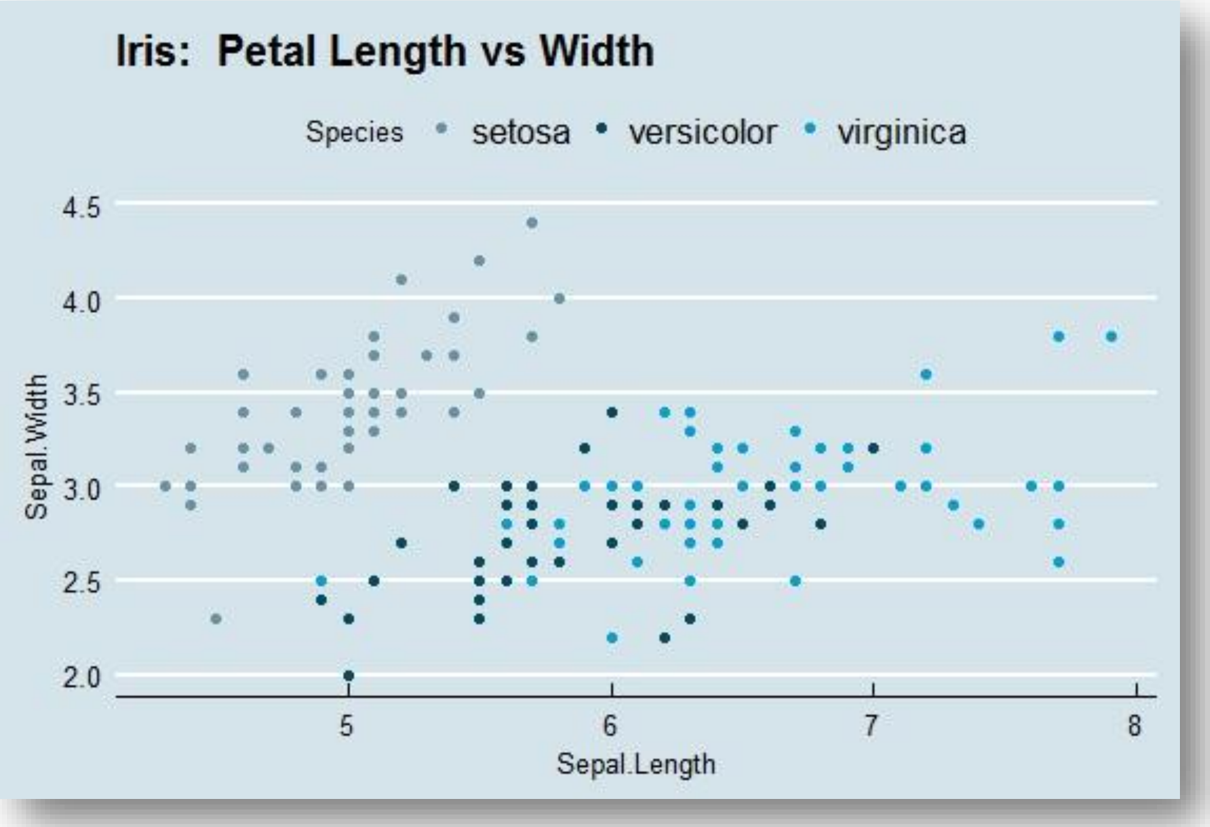


diam_depth_dist + geom_histogram(aes(fill=cut), binwidth=.1, position="fill")



diam_depth_dist + geom_freqpoly(aes(y=..density.., color=cut), binwidth=.1)

USING GGTHEMES



```
## install.packages("ggthemes")

## library(ggthemes)


IrisPlot <- ggplot(iris,

  aes(Sepal.Length, Sepal.Width,

      color=Species)) + geom_point()

IrisPlot + theme_economist() + scale_color_economist() + ggtitle("Iris:  Petal Length vs Width")
```