

Data Analytics

CS301

Exploratory Data Analysis (Continued)

Week 5: 12th Feb
Spring 2020
Oliver BONHAM-CARTER

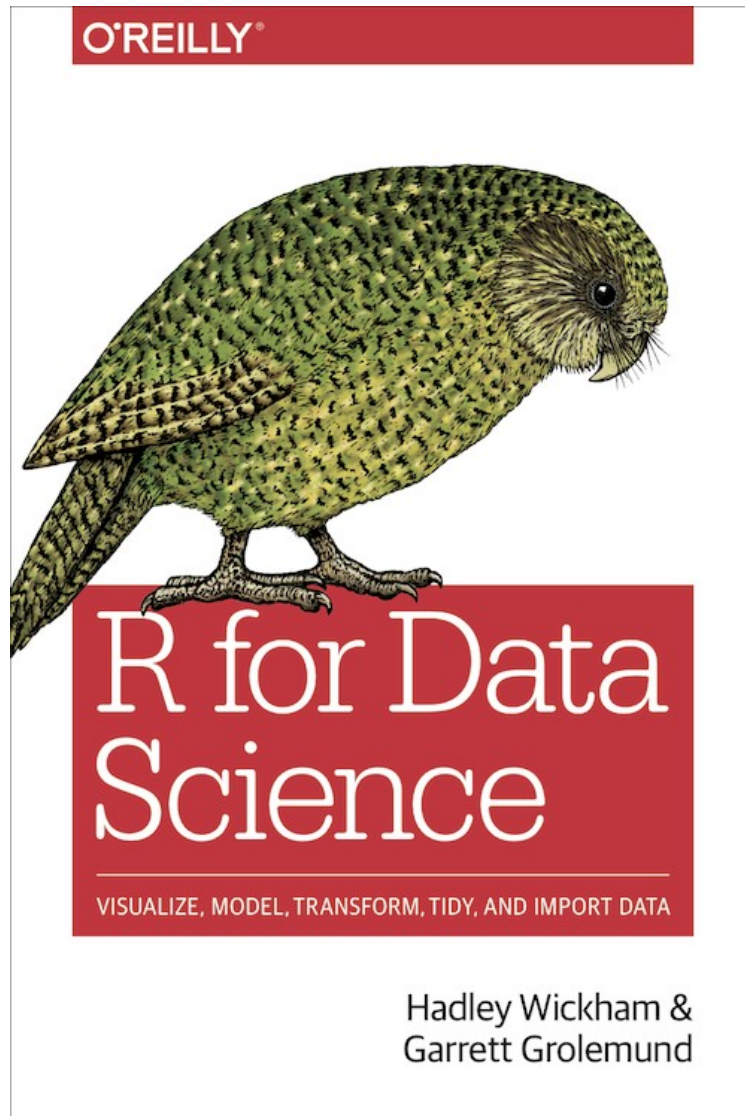
On Exam 1, 2:30pm, 17th Feb 2020

- Online exam: multiple choice, matching, True and False.
- Up through week 5: Google Analytics, Exploratory Data Analysis Material, tibble and Tidy Data
- **Study your slides and notes.**
- **Turn to the book to provide further detail as needed.**
- Be familiar with the graphing code in R. You do not have to write code, but you will have to recognize working code.
- Be familiar with concepts discussed in class such as problems using bin-widths, working with outliers or missing information, categorical variables and others discussion points.

On Exam 1, 2:30pm, 17th Feb 2020

- Topics
 - Google Analytics
 - Web traffic Information: understanding terms and reading plots, analysis depending on goals of a type of website
 - Data Visualizations: types, meanings and when to use
 - R Statistics: recognizing the code to make particular plots
 - Basic syntax and methods
 - Library features:
 - Tidyverse, nycflights13, tibble
 - Concepts:
 - Exploratory data analysis
 - Tidy data manipulation
 - Ggplot, and others

Where in the Web? Where in the Book?



- Note the chapter differences!
- Book:
 - Chap 5: Exploratory Data Analysis
- Web:
 - <http://r4ds.had.co.nz/exploratory-data-analysis.html>
 - Chap 7: Exploratory Data Analysis



Missing Data Points?

MISSING





Missing Data Entries

- Missing data in R appears as **NA**.
- **NA** is not a string or a numeric value, but an indicator of missing data.
- Let's create vectors with missing values to test

```
library(tidyverse)
library(tibble)
x1 <- c(1, 4, 3, NA, 7)
x2 <- c("a", "B", NA, "NA")
is.na(x1)
is.na(x2)
```

Spot
missing
data



Missing Data Entries

- What to do when elements of your data go missing?
- **Why not just DROP the ENTIRE ROW, as well as to drop all the value contained by its other variables as well??**

```
diamonds2 <- diamonds %>% filter(between(y, 3, 20))
```

This is a shortcut for $y \geq 3$ & $y \leq 20$

```
View(diamonds2)
```

```
# compare to the the size of original dataset
```

```
View(diamonds)
```

```
# Note: Good data may have been lost by dropping rows.
```



IfElse(): Condition Statement

```
y = ifelse(y < 3 | y > 20, NA, y)
```

- **Function**
- **Test Condition**
- **If True, then assign this**
- **If False, then assign this**



Data: *Diamond*

The book recommends to *mark* the data as bad or missing.

```
diamonds2 <- diamonds %>%
```

```
  mutate(y = ifelse(y < 3 | y > 20, NA, y))
```

syntax: `ifelse(test, yes, no)`

Inspect each value of *y*. If the *y* is not between 3 and 20, then *y* = NA, else *y* = *y*



We Plot All Non-NA Values

Missing, outliers values marked as NA

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

compared to, no removed missing or outlier values

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

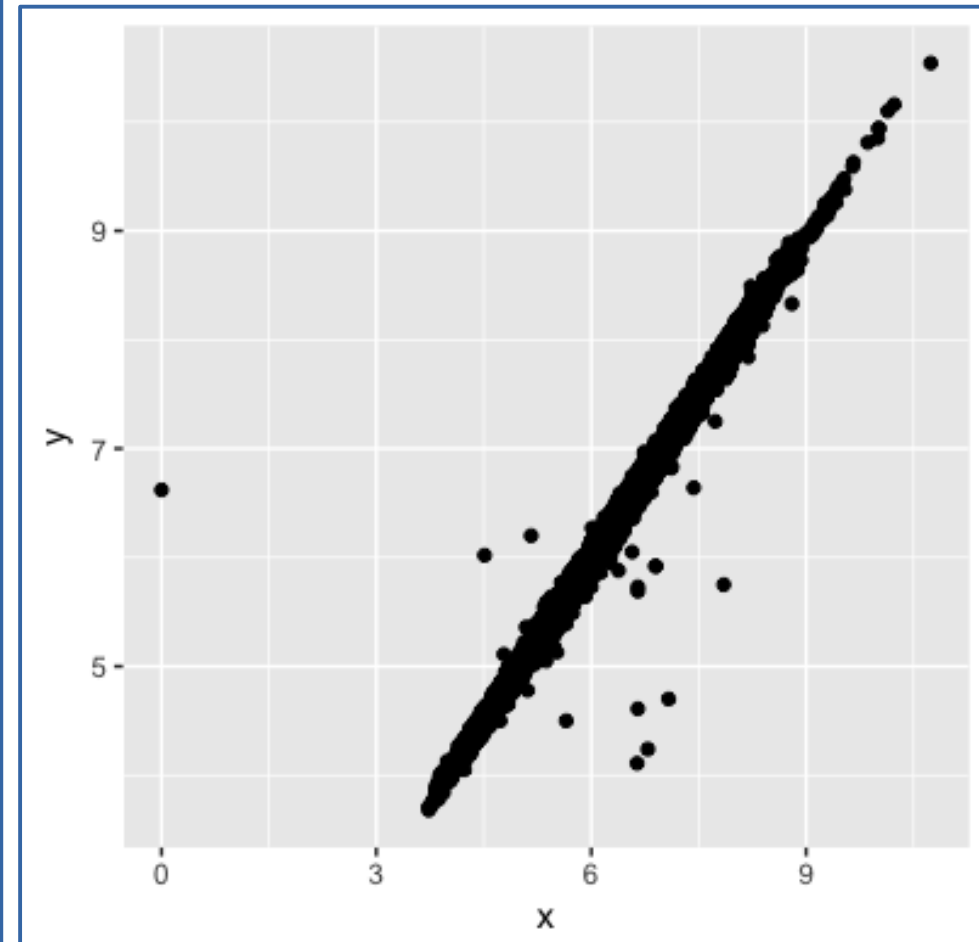
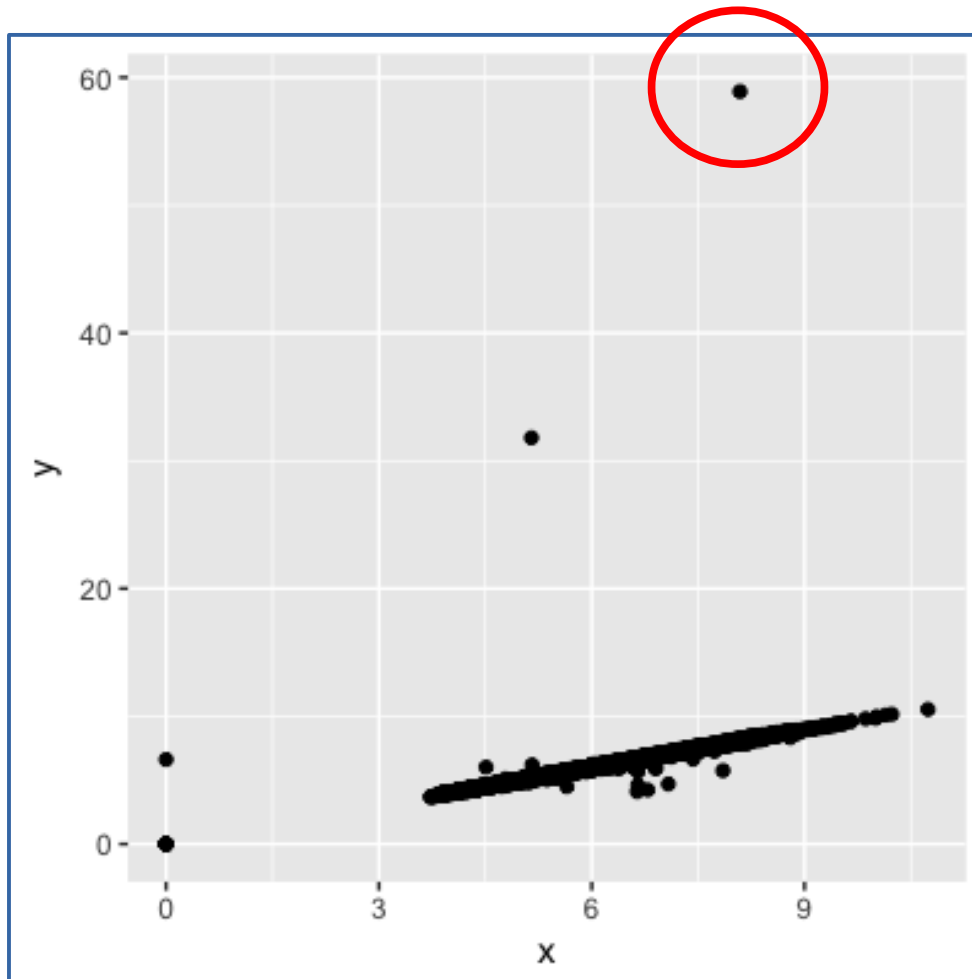


Missing Values, continued

```
# remove the outliers for y (i.e., y<3 and y >20)
library(tidyverse)
diamonds2 <- diamonds %>% mutate(y = ifelse(y < 3 | y > 20, NA, y))
ggplot(data = diamonds2, mapping = aes(x = x, y = y)) + geom_point()
```

**Add the ifelse() code
directly to your other code.**

Trimmed Data, Slightly Different Plot...



- Left: **WITH** outliers
- Above: **NO** outliers



Data: *Diamond*

Can you use the below code to further trim outliers or missing data?

Plot your new graphic after using *ifelse()*

```
diamonds3 <- diamonds %>%  
  mutate(y = ifelse(y < ## | y > ##, NA, y))
```

THINK

Missing Values May Have Their Own Meanings

- Does missing flight arrival-time data indicate canceled flights?



A photograph of an airport flight information display board. The board has four columns: TIME, DESTINATION, GATE#, and STATUS. All flights listed are canceled, with the word 'CANCELLED' appearing in the STATUS column. The GATE# column contains dashes (---) for all flights.

TIME	DESTINATION	GATE#	STATUS
12:00	COPENHAGEN	---	CANCELLED
12:15	PARIS	---	CANCELLED
12:25	LONDON	---	CANCELLED
13:20	FRANKFURT	---	CANCELLED
13:45	ZURICH	---	CANCELLED
14:35	BRUSSELS	---	CANCELLED
15:00	MILAN	---	CANCELLED
16:25	KYIV	---	CANCELLED
16:55	MOSKOW	---	CANCELLED



Missing Values May Have Their Own Meanings

```
# install the flights data, if necessary.  
install.packages("nycflights13")  
library(tidyverse, nycflights13)  
flights <- nycflights13::flights  
View(flights)  
  
# Where are the missing values  
flights$dep_time
```

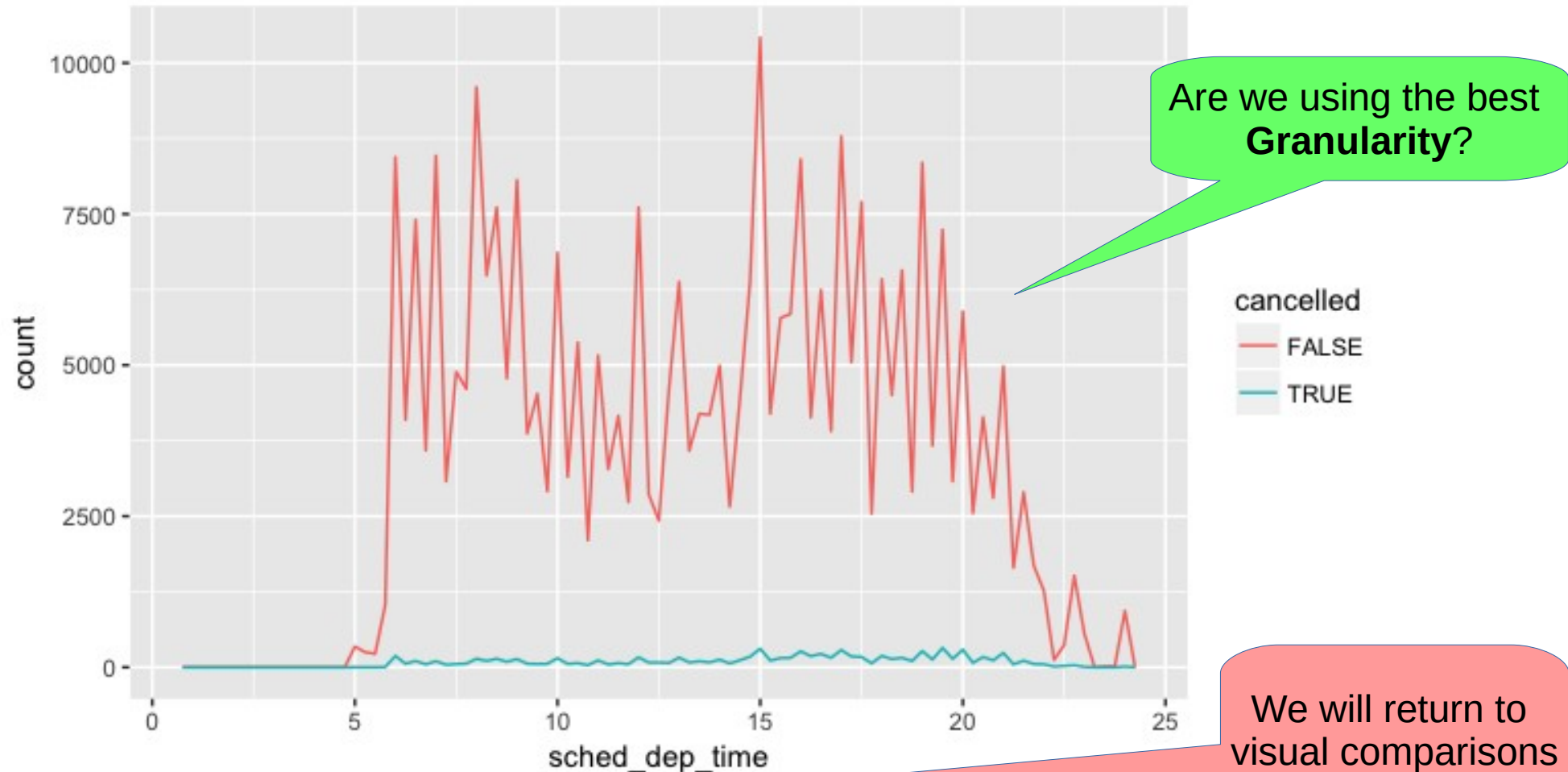


The Distribution of a **Continuous** Variable, Aggregated By a **Categorical** variable

```
# compare the scheduled departure times for cancelled and non-  
cancelled times  
flights %>%  
  mutate(  
    cancelled = is.na(dep_time),  
    sched_hour = sched_dep_time %/% 100,  
    sched_min = sched_dep_time %% 100,  
    sched_dep_time = sched_hour + sched_min / 60  
  ) %>%  
  ggplot(mapping = aes(sched_dep_time)) +  
  geom_freqpoly(mapping = aes(colour = cancelled),  
    binwidth = 1/4)
```




Potential Pitfalls in Theory



- We get an slight idea of when cancellations happen
- Many more non-cancelled flights than cancelled flights: *does the business side of flying introduce a bias for not-canceling flights?*



Covariation

covariance



co·var·i·ance

/ˌkōˈverēəns/

noun

1. MATHEMATICS

the property of a function of retaining its form when the variables are linearly transformed.

2. STATISTICS

the mean value of the product of the deviations of two variates from their respective means.

- **Covariation** is the tendency for the values of two or more variables to vary together in a related way.
- Study covariation by visualizing relationships between two or more variables.
- Pay attention to your variables to know how best to visualize these variables



Covariation

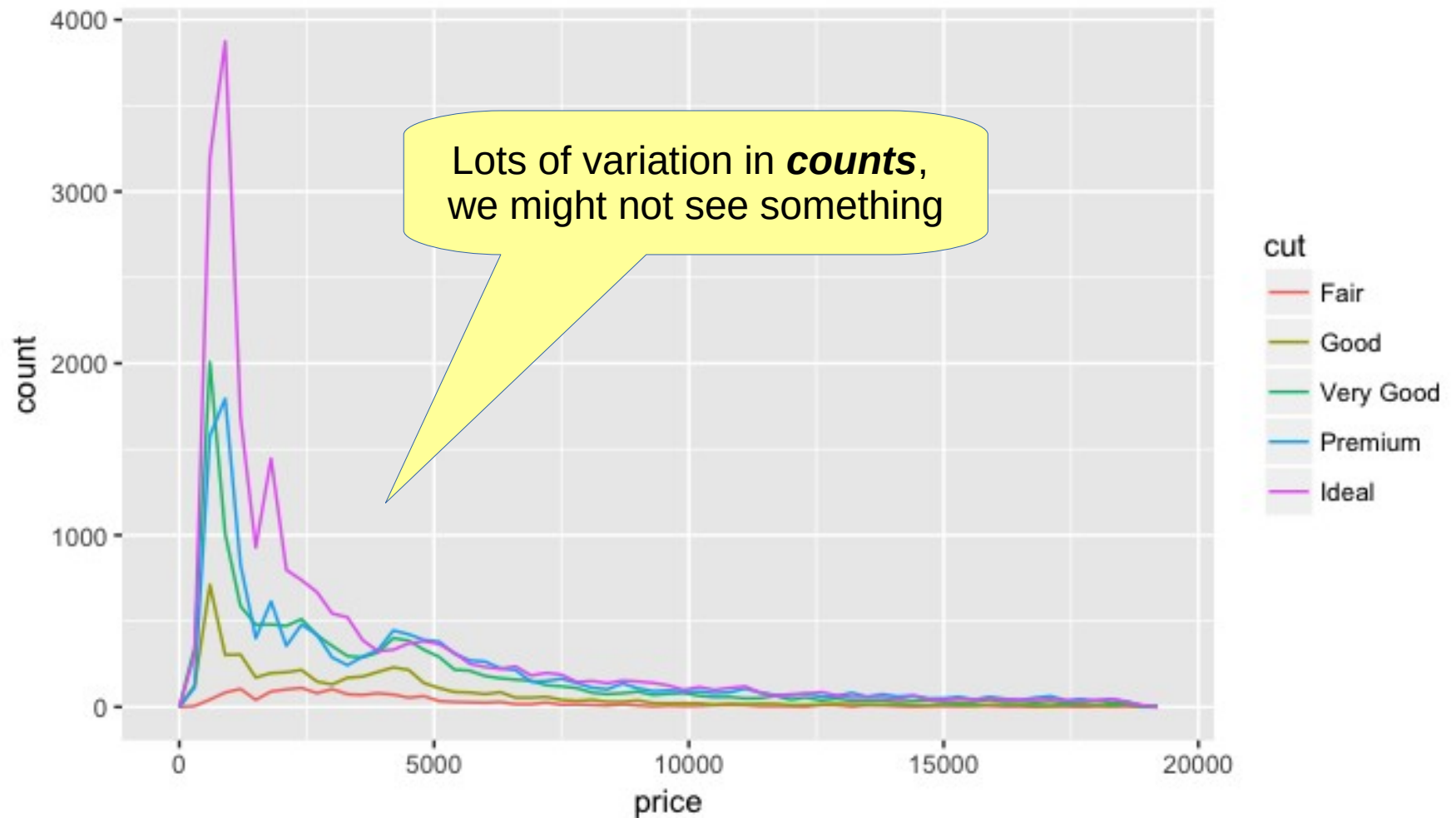
- How do the prices of diamonds vary with quality?

Plot the count of each cut quality according to price.

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



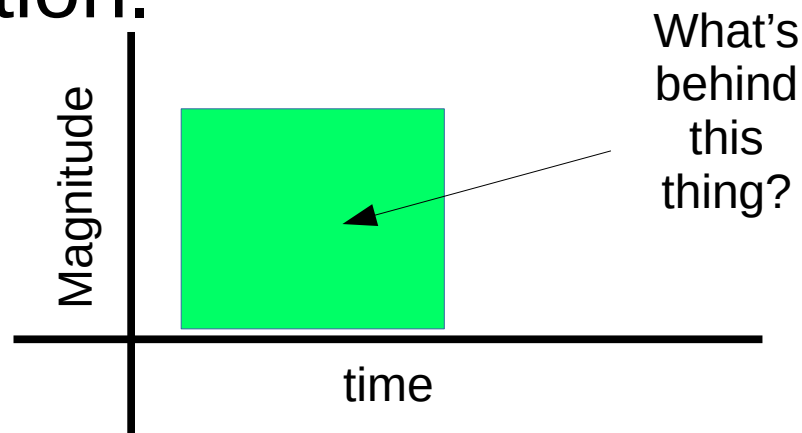
The Plot of the Diamond Counts





This Plot May Make It Hard To See The Phenomenon

- The counts variable seems to have values from all over the range.
- This is noise in our plot
- If one group is much smaller than the others, then it is hard to see the differences in its distribution.





Let's Change Our Plotting

Does a histogram help?

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

#Note: **Density**, is the count standardised so that the area under each frequency polygon is one.

change the axis to see behind them.

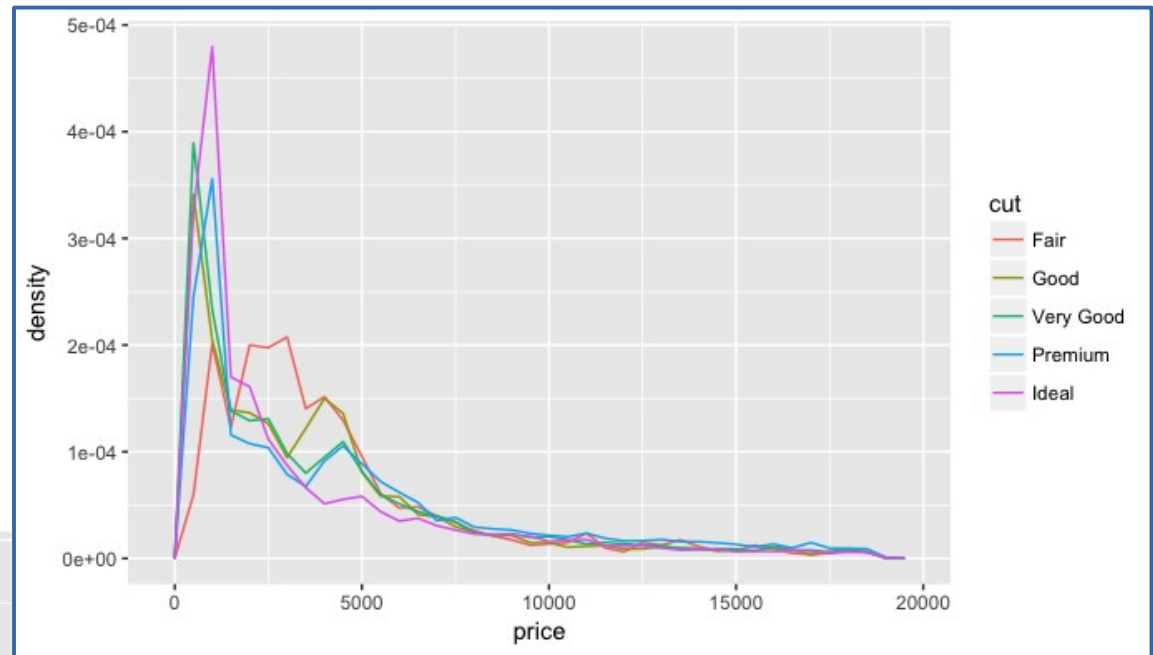
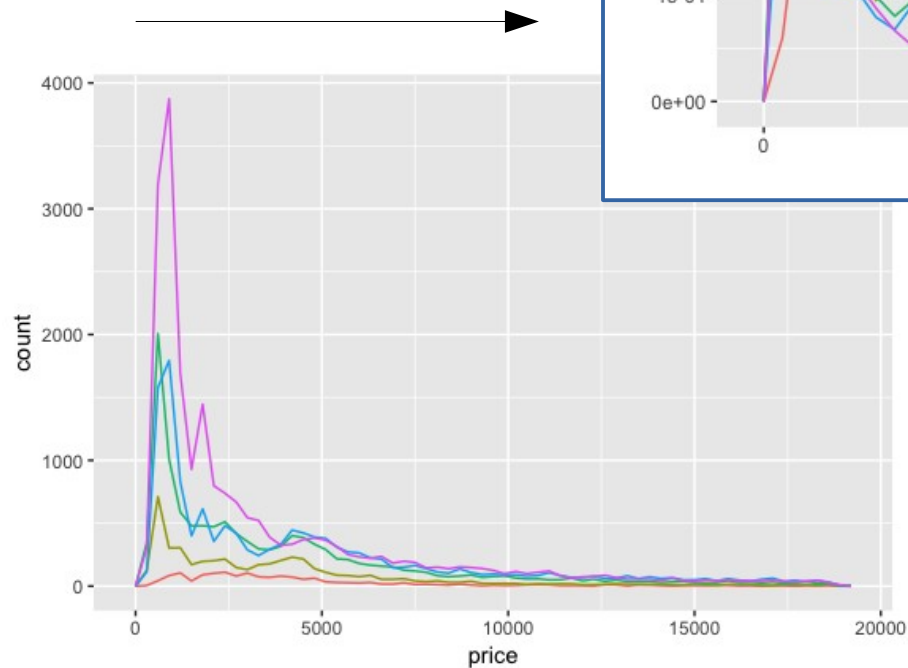
```
ggplot(data = diamonds, mapping = aes(x = price, y  
= ..density..)) + geom_freqpoly(mapping = aes(colour  
= cut), binwidth = 500)
```

Normalize Your View!



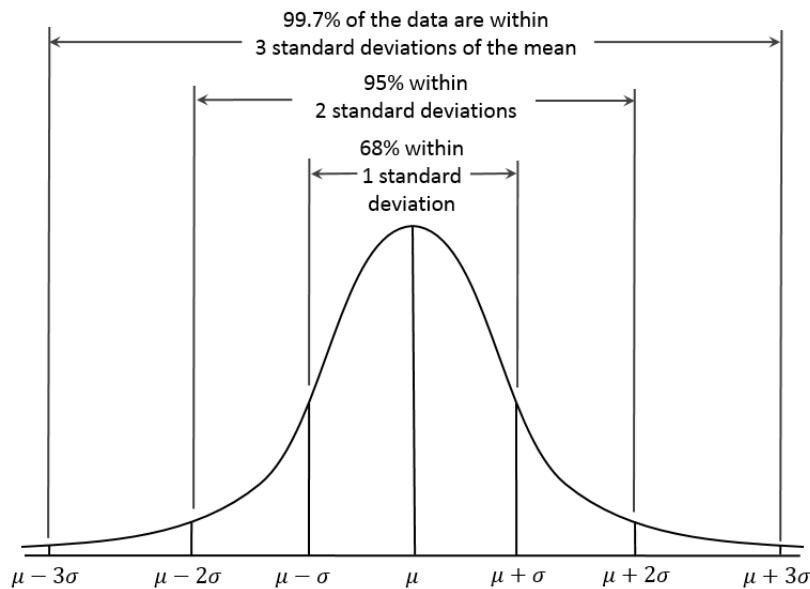
Different Plots

```
mapping = aes(  
  x = price,  
  y = ..density..  
)
```

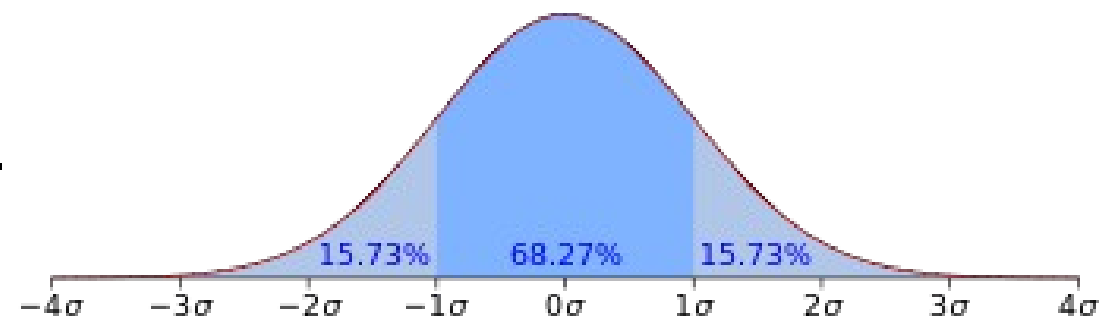
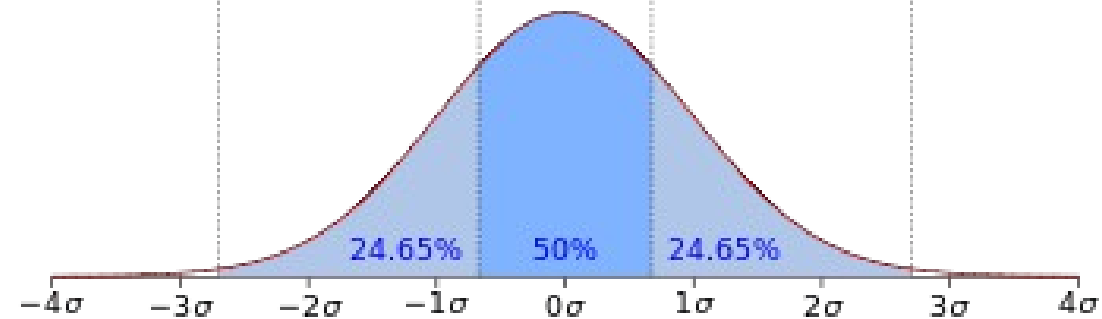
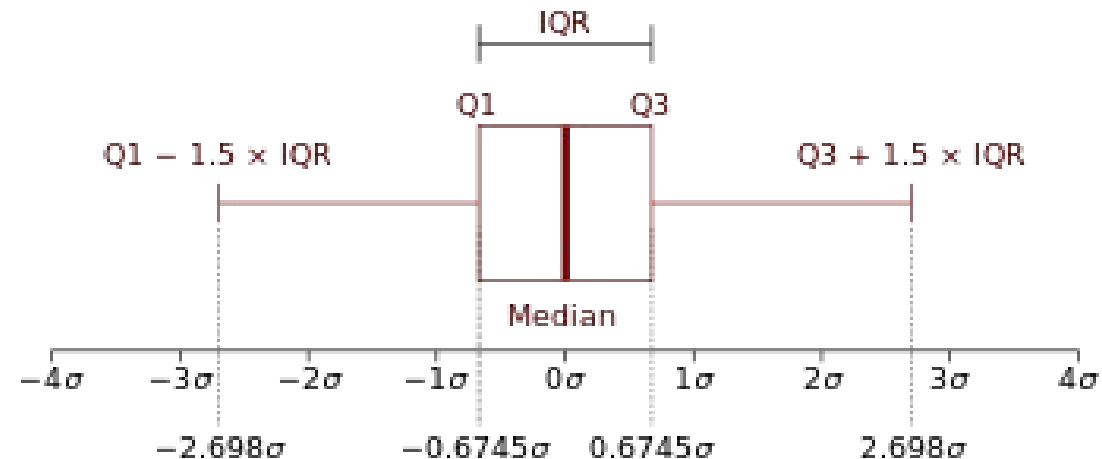


```
mapping = aes(  
  x = price,  
)
```

Box Plots



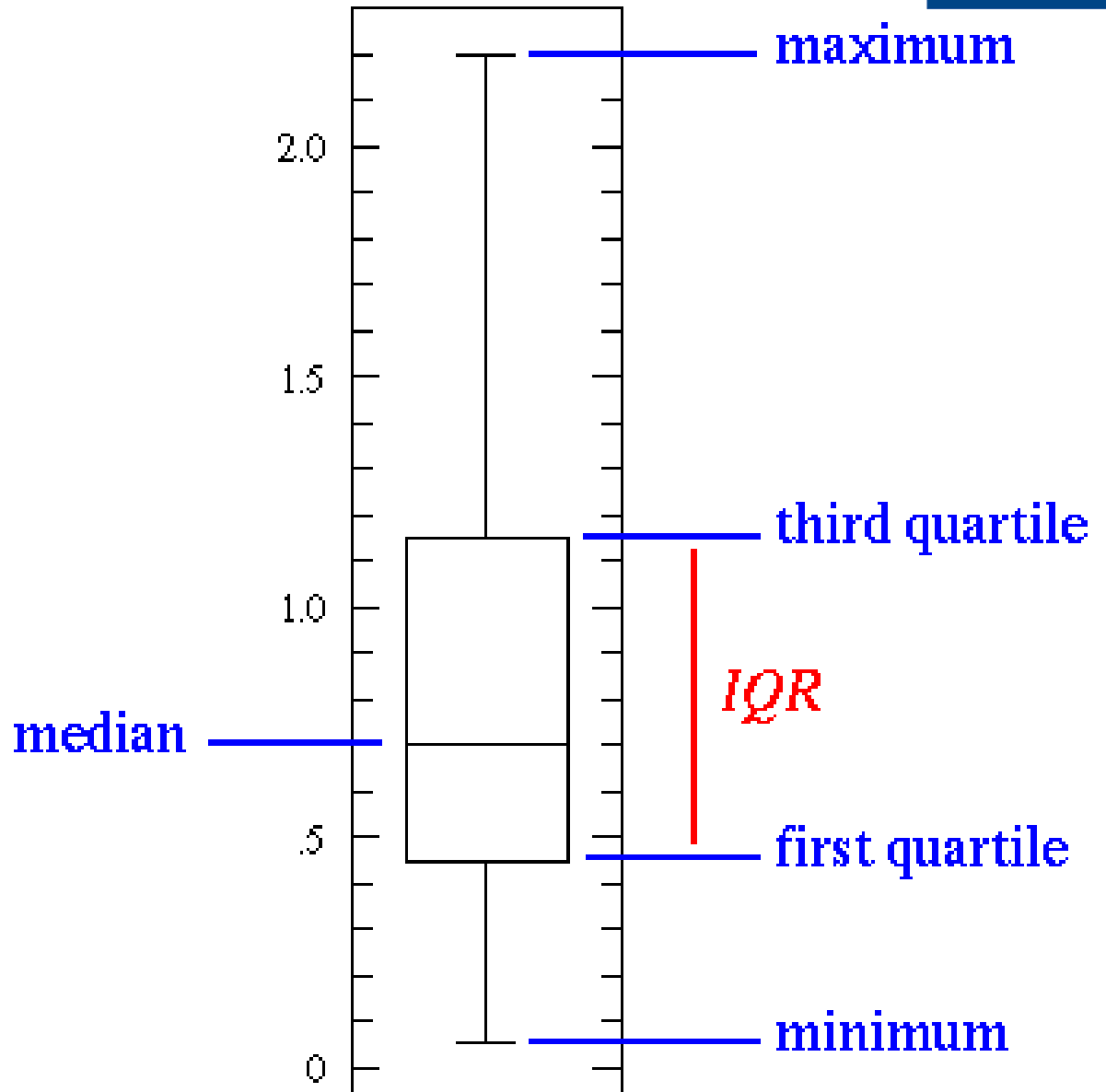
- For the normal distribution, the values less than one standard deviation away from the mean account for 68.27% of the set; while two standard deviations from the mean account for 95.45%; and three standard deviations account for 99.73%.





Explore Data Using Box Plots

Standardized way of displaying the distribution of data based on the five number summary: minimum, first quartile, median, third quartile, and maximum





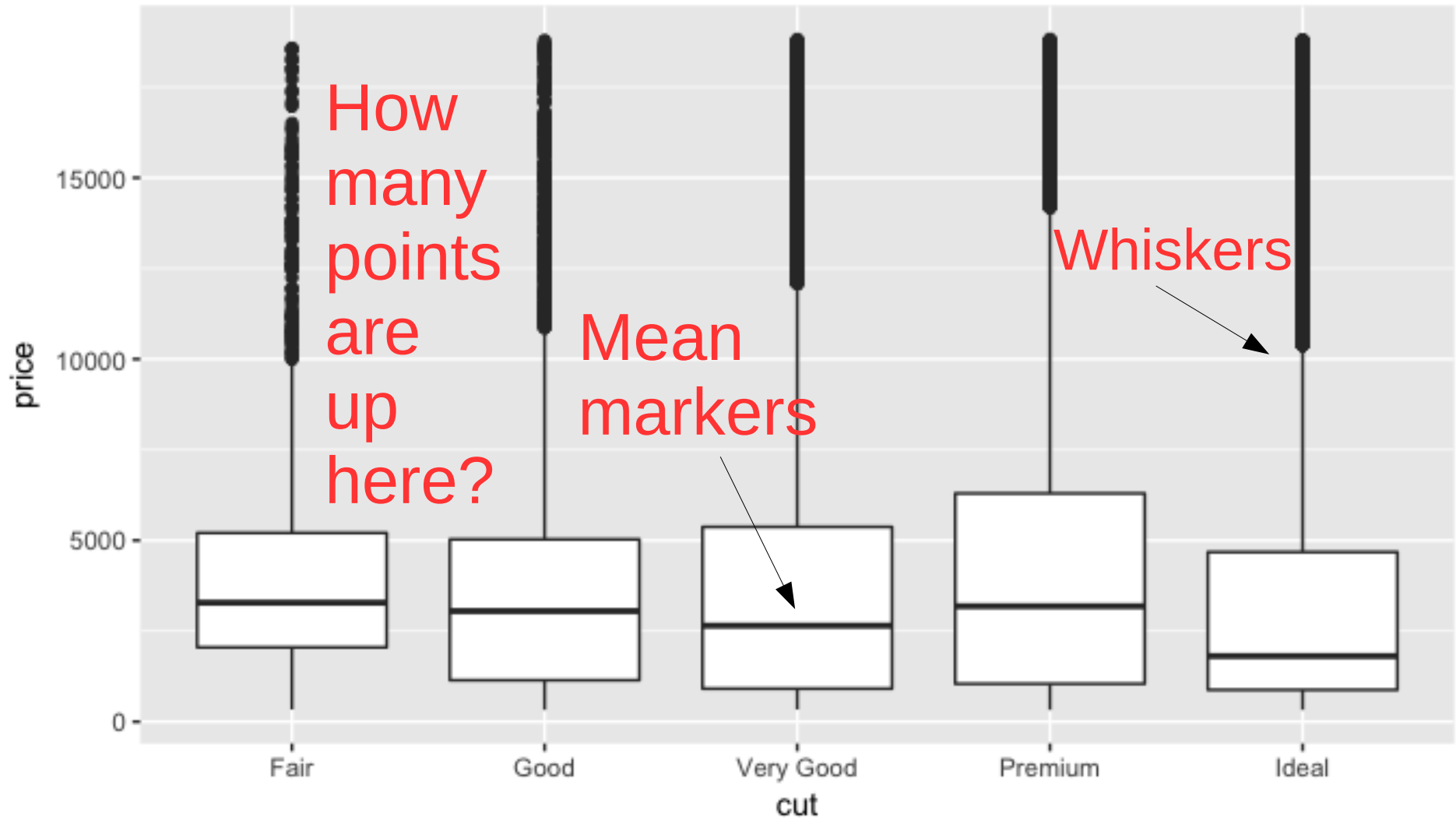
Explore Data Using Box Plots

Make a box plot to describe covariance between cut and price.

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



Explore Data Using Box Plots





Box Plots: Pros and Cons

- Much less information about the *cut* distribution.
- Boxplots much more compact for convenient comparison
- Be careful, we could incorrectly conclude that better quality diamonds are cheaper on average.





Two Categorical Variables

Visualize the covariation between categorical variables with a “Plot of Dots” to determine observations.

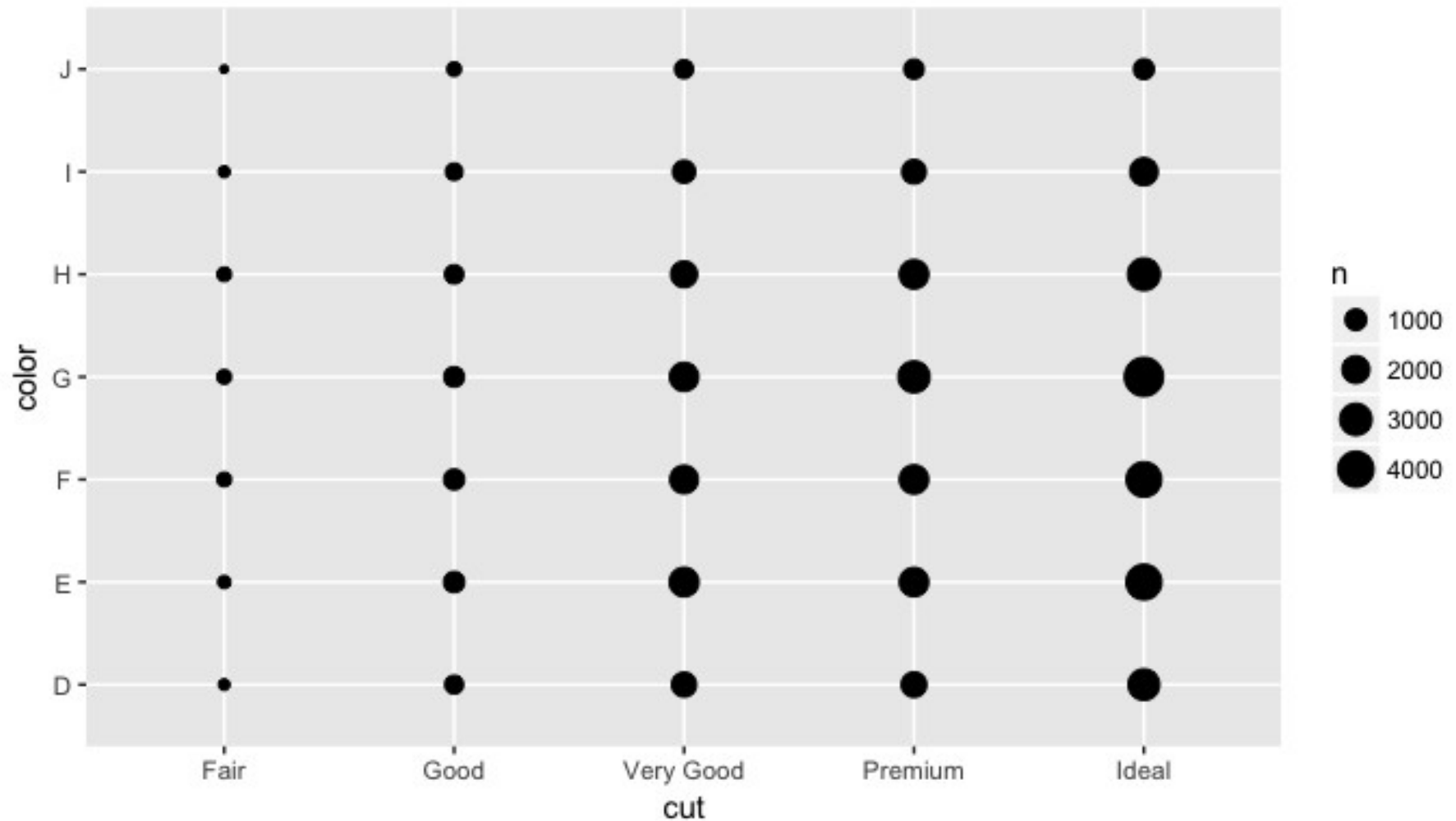
```
ggplot(data = diamonds) +  
geom_count(mapping = aes(x = cut, y = color))
```

Note: The size of each circle in the plot displays how many observations occurred at each combination of values

```
# Get exact text details of the plot  
diamonds %>% count(color, cut)
```



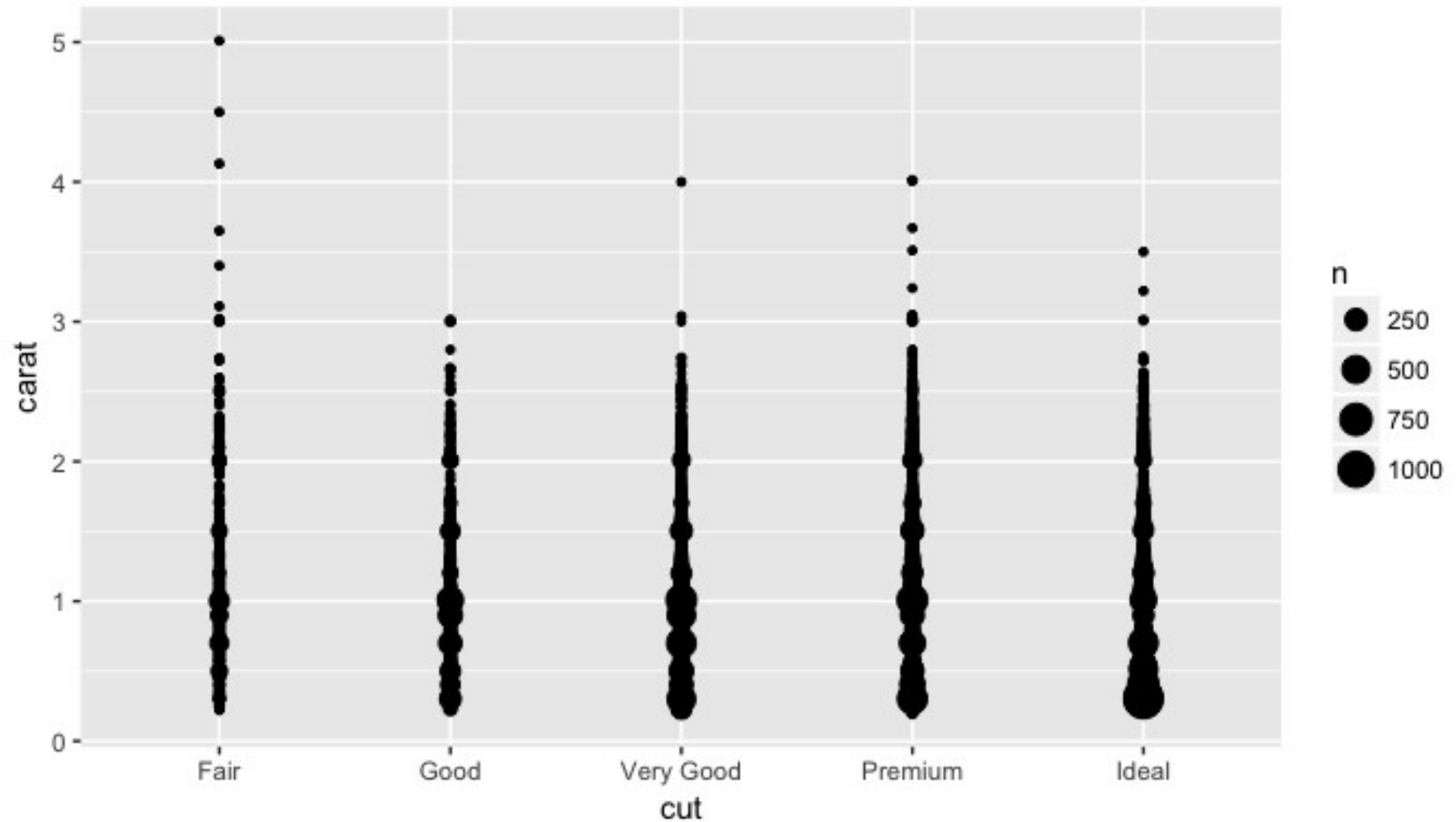
Mini Distributions: Cut vs Color



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



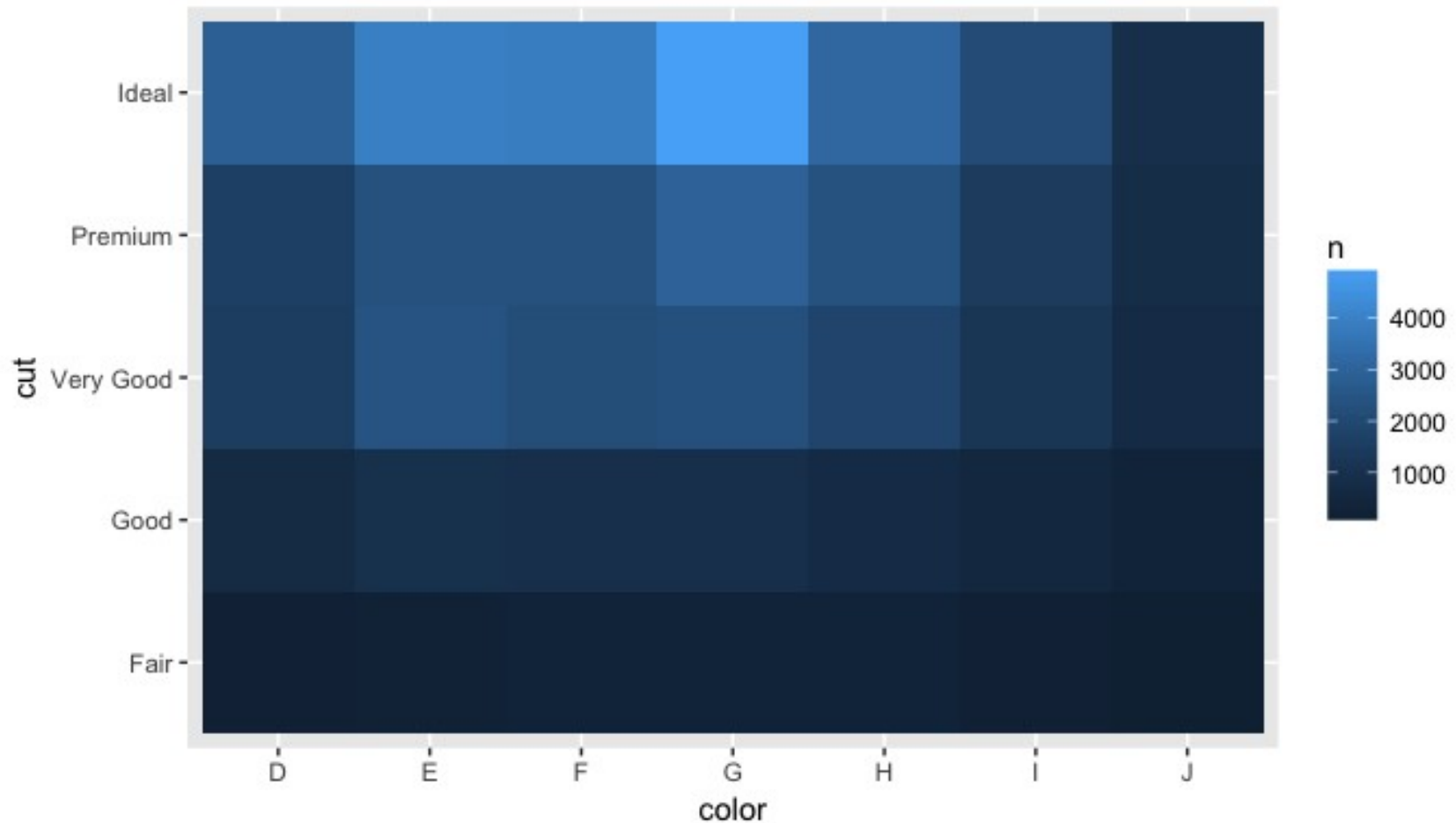
Mini Distributions: Cut vs Carat



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



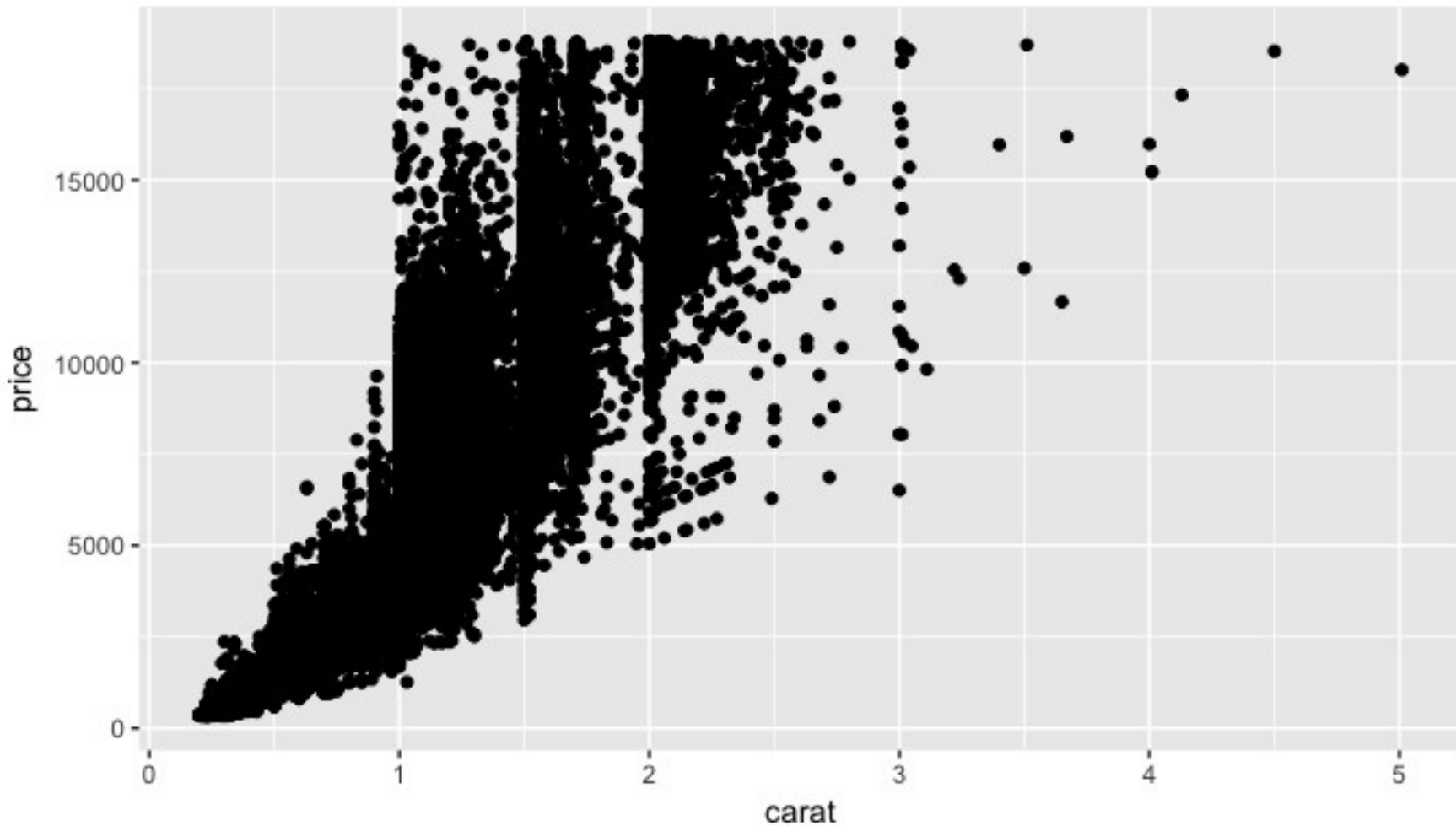
Mini Distributions: Cut vs Color



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



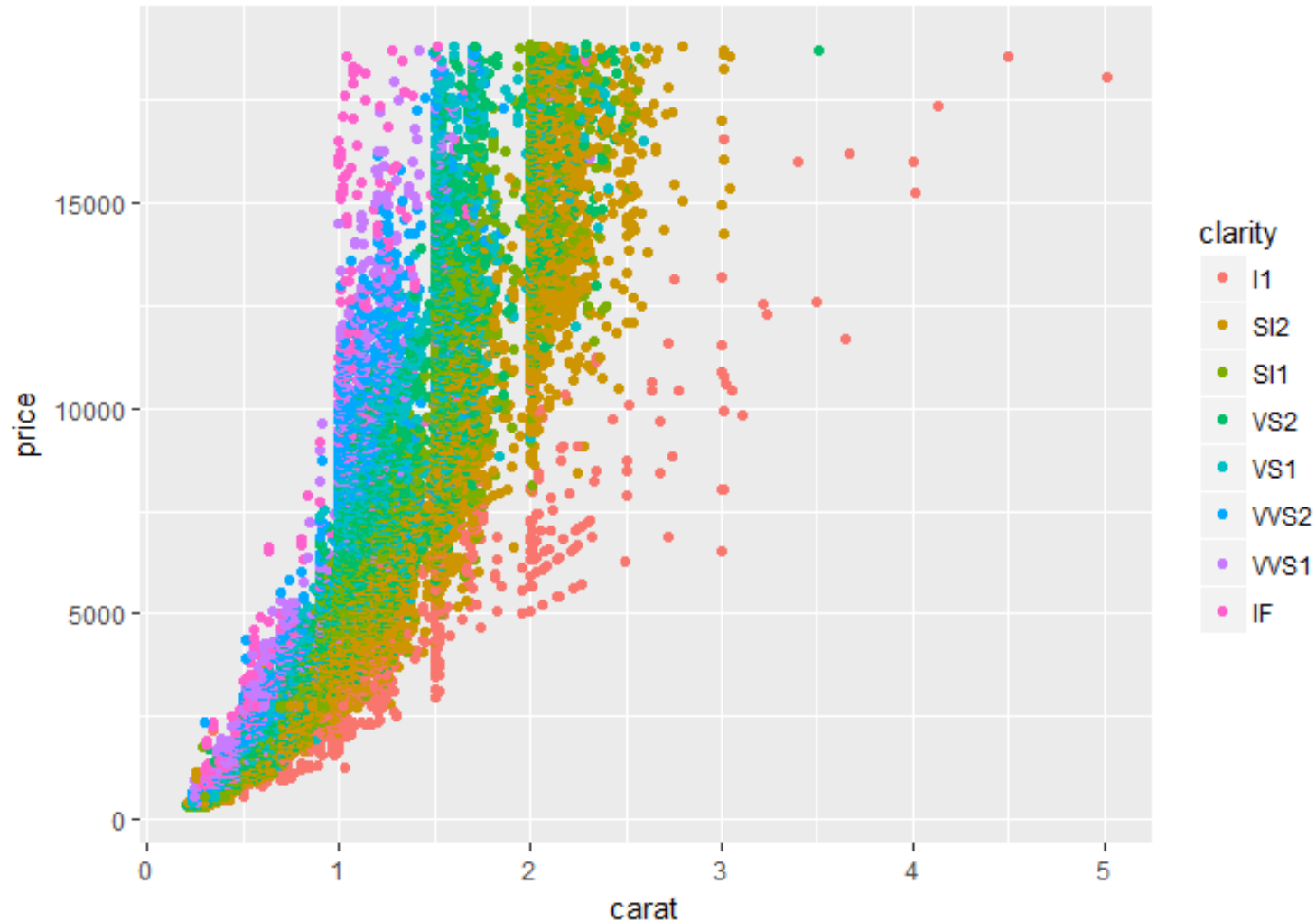

Mini Distributions: Carat vs Price



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



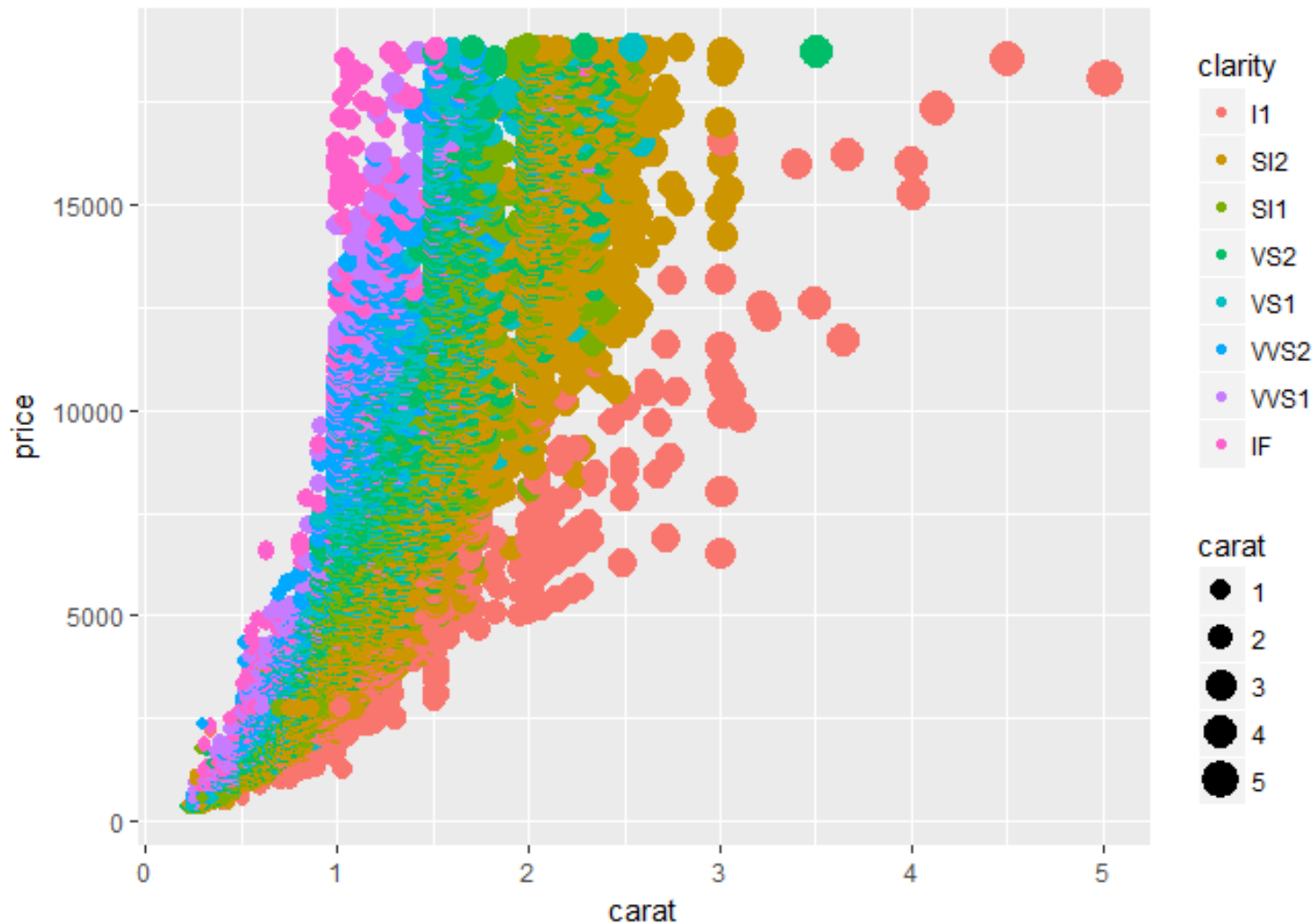
Mini Distributions: Carat vs Price



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



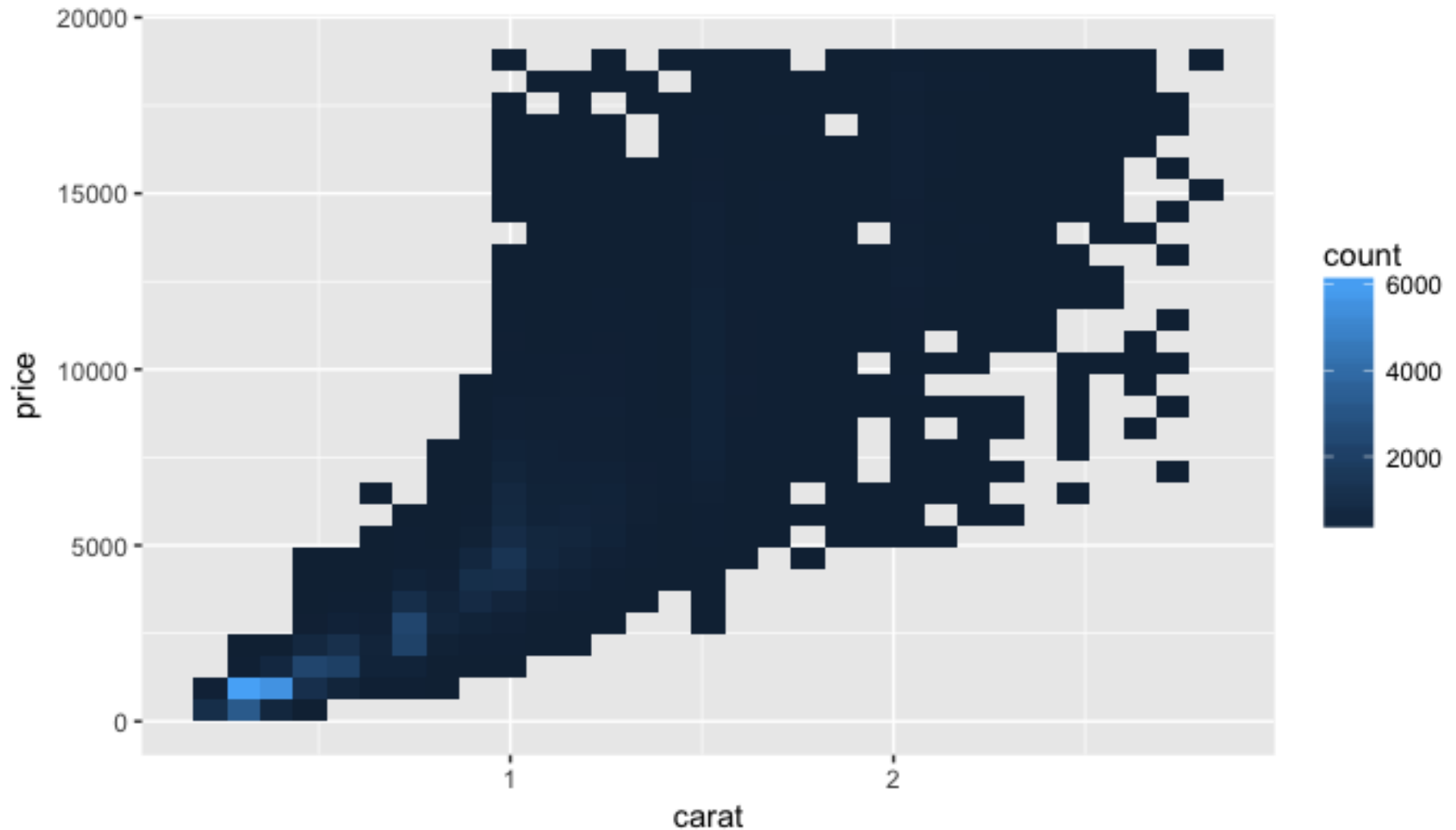
Mini Distributions: Carat vs Price



```
ggplot(data = diamonds) + geom_point(mapping =  
aes(x = carat, y = price,color= clarity, size = carat))
```



Mini Distributions: Carat vs Price



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



Consider This: *plots*

Can you plot your diamond dataset with and without outliers?

Compare your plots.

Play with the binwidths: What is a good size of bin to use?

Time permitting, can you find another dataset to trim outliers?

```
diamonds3 <- diamonds %>%  
  mutate(y = ifelse(y < ## | y > ##, NA, y))
```

THINK