Bollinger HW 3 COM 521

Beth Bollinger

1/12/2017

### PC4. Get to know your data! Do whatever is necessary to summarize the new dataset.

summary(week3\_dataset.beth)  
names(week3\_dataset.beth)  
dim(week3\_dataset.beth)

Above, you can see the summary statistics for this new dataset. In particular, there are 100 rows and 5 columns, as can be seen by the code dim.

## What are the appropriate summary statistics to report for each variable?

summary(week3\_dataset.beth$x)  
sd(week3\_dataset.beth$x)  
range(week3\_dataset.beth$x)

As shown above, for the variable 'x,' there is a minimum value of -40.85, maximum of 173.40, median of 41.79, mean of 43.10, and standard deviation of 41.66937

summary(week3\_dataset.beth$j)  
sd(week3\_dataset.beth$j)  
range(week3\_dataset.beth$j)

As shown above, for the variable 'j,' there is a minimum value of 0, maximum of 1, median of 0, mean of 0.43, and standard deviation of 0.4975699. Clearly, this variable is a 0 or 1 (yes/no, present/not present) nominal variable.

summary(week3\_dataset.beth$i)  
sd(week3\_dataset.beth$i)  
range(week3\_dataset.beth$i)

As shown above, for the variable 'i,' there is a minimum value of 0, maximum of 1, median of 0, mean of 0.45, and standard deviation of 0.5. Clearly, this variable is a 0 or 1 (yes/no, present/not present) nominal variable.

summary(week3\_dataset.beth$k)  
sd(week3\_dataset.beth$k)  
range(week3\_dataset.beth$k)

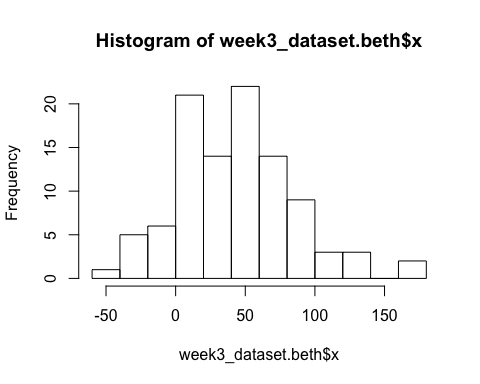
As shown above, for the variable 'k,' there is a minimum value of 0, maximum of 3, median of 1, mean of 1.47, and standard deviation of 0.797154.

summary(week3\_dataset.beth$y)  
sd(week3\_dataset.beth$y)  
range(week3\_dataset.beth$y)

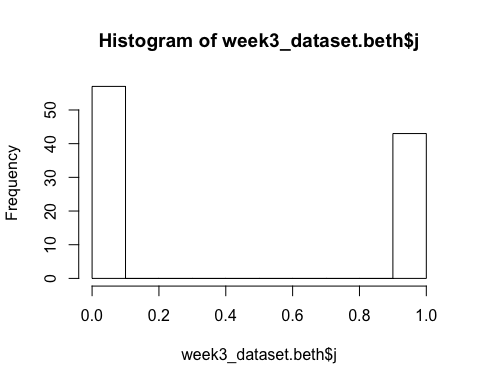
As shown above, for the variable 'y,' there is a minimum value of -199.9, maximum of 793.5, median of 257.4, mean of 259.3, and standard deviation of 196.7348.

## Draw histograms for all of the variables to get a sense of what it looks like.

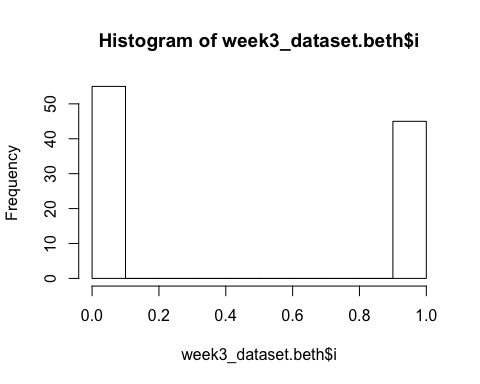
hist(week3\_dataset.beth$x)



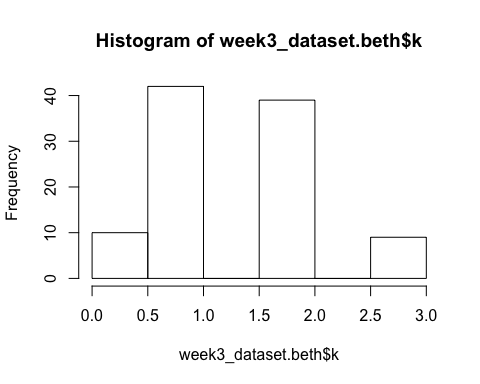
hist(week3\_dataset.beth$j)



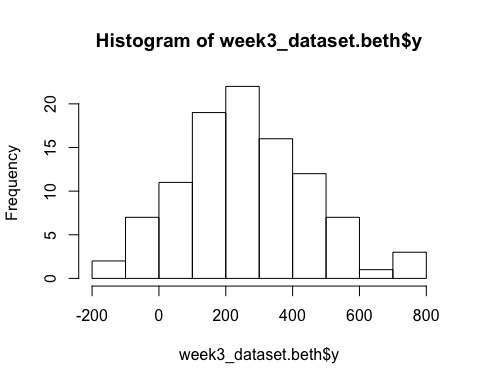
hist(week3\_dataset.beth$i)



hist(week3\_dataset.beth$k)



hist(week3\_dataset.beth$y)



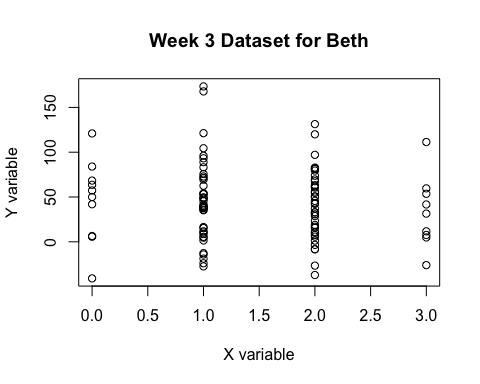
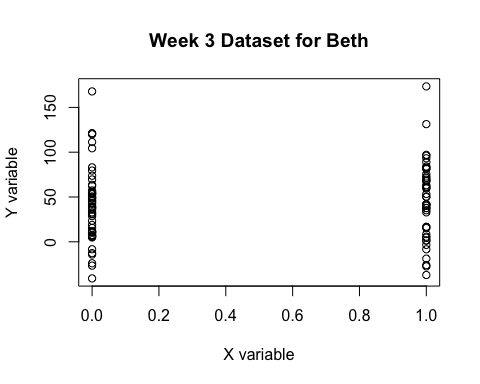
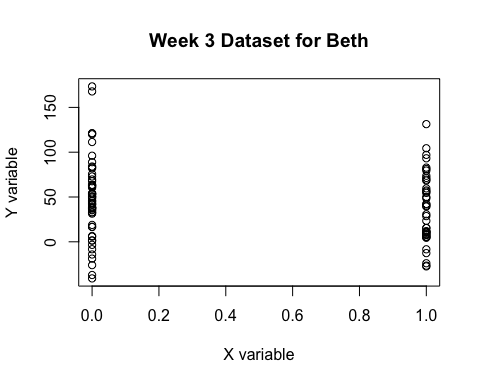
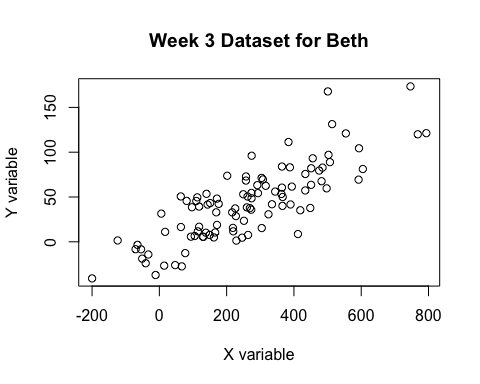
### PC5. Compare the week2.dataset vector with the first column (x) of the data frame. I mentioned in the video lecture that they are similar? Do you agree? How similar? Write R code to demonstrate or support this answer convincingly.

summary(week2.dataset)  
summary(week3\_dataset.beth$x)

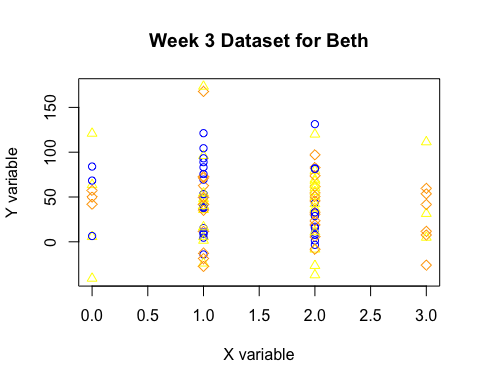
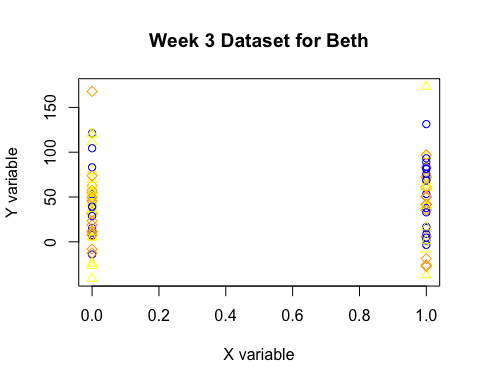
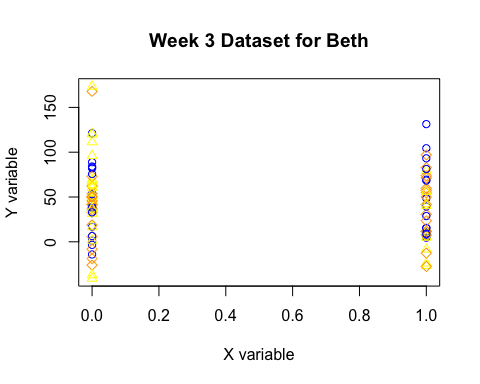
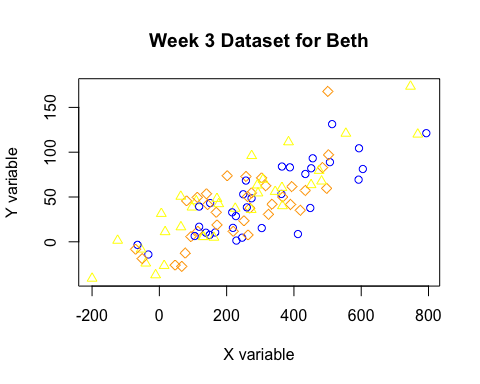
As shown above when you run the summary for each variable, both variables are similar, because their min, median, mean, and max are identical.

### PC6. Visualize the data using ggplot2 and the geom\_point() function. Graphing the x on the x-axis and y on the y-axis seem pretty reasonable! If only it were always so easy! Graph i, j, and k on other dimensions (e.g., color, shape, and size seems reasonable). Did you run into any trouble? How would you work around this?

ggplot2::geom\_point(plot(x~y\*j\*i\*k, data = week3\_dataset.beth, ylab="Y variable", xlab="X variable", main="Week 3 Dataset for Beth"))



ggplot2::geom\_point(plot(x~y\*j\*i\*k, data = week3\_dataset.beth, col=(c("orange", "blue", "yellow")), pch=c(5,1,2), ylab="Y variable", xlab="X variable", main="Week 3 Dataset for Beth"))

 The first plot above shows each of the variables plotted together on the x and y axis. The second plot above adds change to color and shape. I ran into trouble, mapping each variable. I was able to give each one a different shape (theoretically), but I had trouble mapping each one as a point on the graph.

### PC7. A very common step when you import and prepare for data analysis is going to be cleaning and coding data. Some of that is needed here. As is very common, i, j are really dichotomous "true/false" variables but they are coded as 0 and 1 in this dataset. Recode these columns as logical. The variable k is really a categorical variable. Recode this as a factor and change the numbers into textual "meaning" to make it easier. Here's the relevant piece of the codebook (i.e., mapping): 0=none, 1=some, 2=lots, 3=all. The goals is to end up with a factor where those text strings are the levels of the factor. I haven't shown you how to do exactly this but you can solve this with things I have shown you. Or you can try to find a recipe online.

bethdata3i <- week3\_dataset.beth$i  
bethdata3i.f <- factor(bethdata3i, labels = c("FALSE", "TRUE"))  
is.factor(bethdata3i.f)  
levels(bethdata3i.f)  
  
bethdata3j <- week3\_dataset.beth$j  
bethdata3j.f <- factor(bethdata3j, labels = c("FALSE", "TRUE"))  
is.factor(bethdata3j.f)  
levels(bethdata3j.f)  
  
bethdata3k <- week3\_dataset.beth$k  
bethdata3k.f <- factor(bethdata3k, labels = c("None", "Some", "Lots", "All"))  
is.factor(bethdata3k.f)  
levels(bethdata3k.f)

### PC8. Take column i and set it equal to NA when if it is FALSE (i.e., 0). Then set all the values that are NA back to 1. Sorry for the busy work! ;)

bethina <- week3\_dataset.beth$i  
bethina[bethina == 0] <-NA  
bethina[is.na(bethina)] <- 1

As can be shown above, by making all the numbers NA that were zero, that can be done by telling it NA. Then, telling it that those which are NA need to become 1, it switches it so now all numbers are 1 in this variable, "i".

### PC9. Once you have recoded your data, generate new summaries for those three variables. Also, go back and regenerate the visualizations. How have these changed? How are these different from the summary detail you presented above?

summary(bethdata3i.f)  
sd(bethdata3i.f)

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm = na.rm): Calling var(x) on a factor x is deprecated and will become an error.  
## Use something like 'all(duplicated(x)[-1L])' to test for a constant vector.

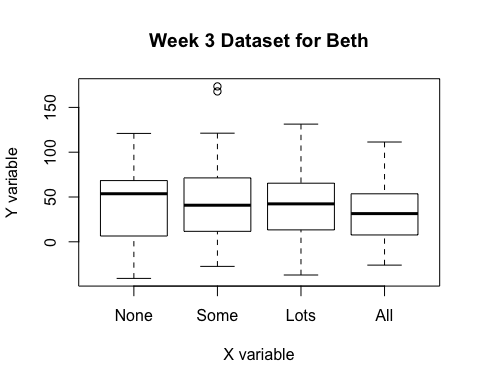
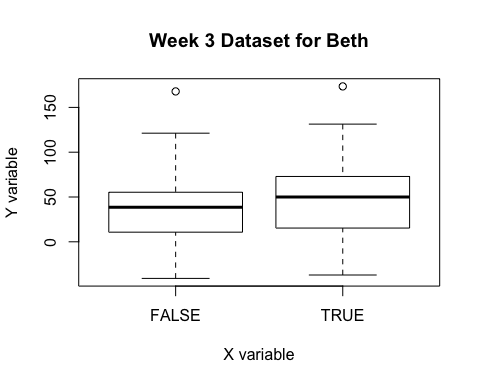
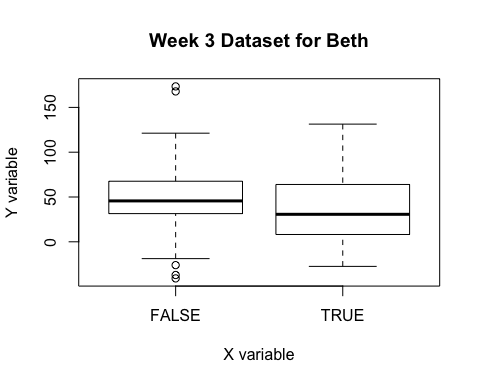
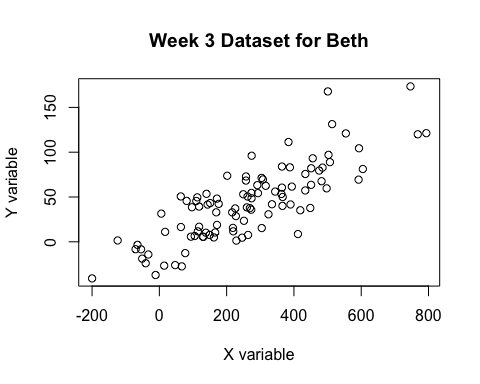
summary(bethdata3j.f)  
sd(bethdata3j.f)

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm = na.rm): Calling var(x) on a factor x is deprecated and will become an error.  
## Use something like 'all(duplicated(x)[-1L])' to test for a constant vector.

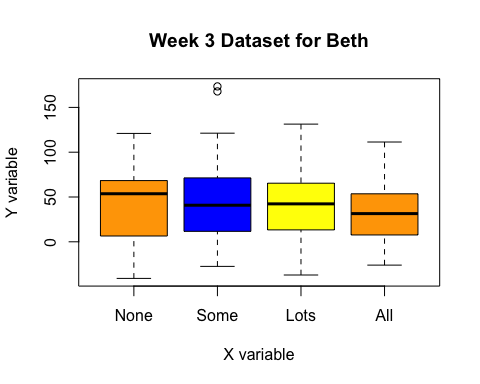
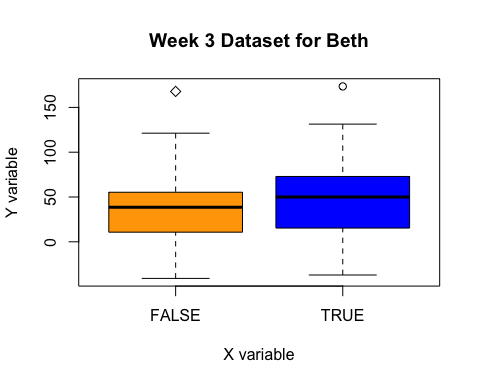
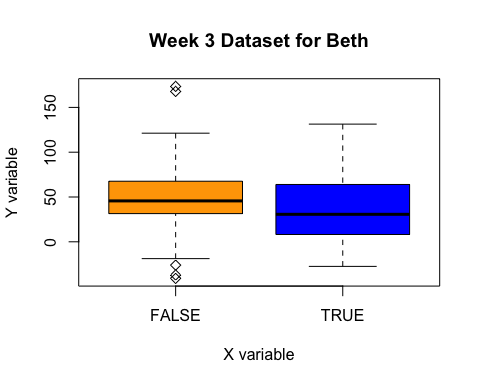
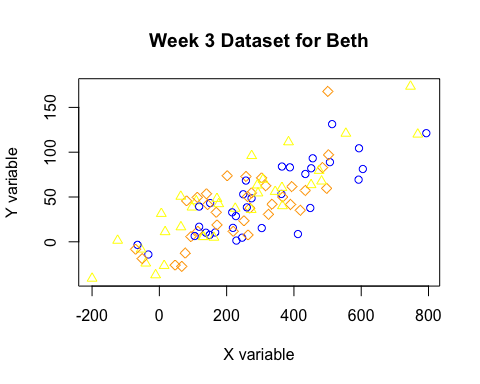
summary(bethdata3k.f)  
sd(bethdata3k.f)

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm = na.rm): Calling var(x) on a factor x is deprecated and will become an error.  
## Use something like 'all(duplicated(x)[-1L])' to test for a constant vector.

ggplot2::geom\_point(plot(x~y\*bethdata3j.f\*bethdata3i.f\*bethdata3k.f, data = week3\_dataset.beth, ylab="Y variable", xlab="X variable", main="Week 3 Dataset for Beth"))



ggplot2::geom\_point(plot(x~y\*bethdata3j.f\*bethdata3i.f\*bethdata3k.f, data = week3\_dataset.beth, col=(c("orange", "blue", "yellow")), pch=c(5,1,2), ylab="Y variable", xlab="X variable", main="Week 3 Dataset for Beth"))

 The big difference, is that now, the results show us how many are actually present (1 or True) versus not present (0 or False). It gives us more data than simply that this is a 0 or 1 nominal variable. Additionally, as can be seen, the graphs make much more sense than the previous graphs, because it can parse out specifics of T/F or 1-3 much better than all grouped together.