Bollinger Wk 2 HW

Beth Bollinger

1/3/2017

load("~/Downloads/week2\_dataset-beth.RData")  
week2.dataset

## [1] 28.693330 35.814335 49.611282 49.991994 59.647642 73.707844  
## [7] -8.364518 96.054222 61.597788 6.466243 48.186404 11.689363  
## [13] 5.658441 7.631364 11.683542 41.553474 54.573235 167.826028  
## [19] 82.805671 35.209281 -23.944501 1.499016 50.543368 71.306859  
## [25] 16.498631 32.894460 50.615489 63.192107 15.336463 -12.631475  
## [31] 7.666535 11.070058 16.777716 121.223774 67.598576 104.397060  
## [37] 56.047398 173.434226 62.503797 42.334679 31.459885 40.081420  
## [43] 48.568774 82.067424 88.904057 83.116294 81.226080 38.425886  
## [49] 53.057048 79.500427 83.995599 -27.439471 63.573402 41.663164  
## [55] -14.160687 45.640550 97.040157 -37.060086 60.475995 10.419978  
## [61] 30.677498 -26.613650 10.120725 4.593508 131.340253 5.547630  
## [67] 18.820737 1.370777 32.869205 41.916485 57.277956 72.915389  
## [73] 23.433062 37.723427 37.264568 69.569425 120.958388 39.319574  
## [79] 5.721526 53.560790 68.319101 53.398808 -40.852148 -26.024733  
## [85] -18.785546 -8.280129 38.556083 93.194209 69.362487 75.675282  
## [91] 37.602928 119.973523 54.302709 4.875573 45.536110 -3.428614  
## [97] 15.518033 111.370781 43.217629 8.642669

## PC3. Once you’ve found the variable, compute and present a series of statistics on it that you should already be familiar with. Use functions to compute the mean, median, variance, standard deviation, and interquartile range.

mean(week2.dataset)

## [1] 43.09519

median(week2.dataset)

## [1] 41.78982

var(week2.dataset)

## [1] 1736.336

sd(week2.dataset)

## [1] 41.66937

IQR(week2.dataset)

## [1] 56.87117

The mean is 43.09519. The median is 41.78982. The variance is 1736.336 The standard deviation is 41.66937 The IQR is 56.87117

## PC4. Although these basic functions all exist, many things you will want to do in the future won’t have functions. Write R code to compute these three statistics by hand: mean, median, and mode. It’s OK if getting the answer involves some eyeballing or counting this by hand. But do get the answer and be ready to walk us through how you did it.

sum(week2.dataset)/100 #mean

## [1] 43.09519

sortweek2data <- week2.dataset[order(week2.dataset)] #median  
table(week2.dataset)

## week2.dataset  
## -40.8521479536028 -37.0600861729767 -27.4394707301933 -26.6136497602233   
## 1 1 1 1   
## -26.0247329774009 -23.9445014953885 -18.7855464905273 -14.1606871877022   
## 1 1 1 1   
## -12.6314747958724 -8.36451775798208 -8.28012876527099 -3.42861381172755   
## 1 1 1 1   
## 1.3707773689551 1.49901558852403 4.59350826279258 4.87557297527698   
## 1 1 1 1   
## 5.54762996020345 5.65844114537842 5.72152565085104 6.46624258843072   
## 1 1 1 1   
## 7.63136435374992 7.6665353205413 8.64266896440952 10.1207250154375   
## 1 1 1 1   
## 10.4199778003196 11.0700577773358 11.6835420015188 11.6893630606738   
## 1 1 1 1   
## 15.3364627730003 15.5180333210577 16.4986305405722 16.7777160094759   
## 1 1 1 1   
## 18.8207371660071 23.4330615097242 28.6933296446239 30.6774975685442   
## 1 1 1 1   
## 31.4598853079133 32.8692047221264 32.8944598914031 35.209280549363   
## 1 1 1 1   
## 35.8143349945542 37.264567873172 37.6029282370497 37.7234274438144   
## 1 1 1 1   
## 38.4258862011859 38.5560833533736 39.3195742083734 40.0814200159895   
## 1 1 1 1   
## 41.5534743751608 41.6631636365883 41.9164849893844 42.334678712197   
## 1 1 1 1   
## 43.2176289187863 45.5361096844647 45.6405496134666 48.1864040055618   
## 1 1 1 1   
## 48.5687737780364 49.6112824985527 49.9919936657234 50.5433676890076   
## 1 1 1 1   
## 50.6154894589018 53.0570483207756 53.3988077408265 53.5607896751654   
## 1 1 1 1   
## 54.3027088373517 54.5732346903718 56.0473977911211 57.2779564604667   
## 1 1 1 1   
## 59.6476416938104 60.4759949947662 61.5977879611046 62.5037970633328   
## 1 1 1 1   
## 63.1921073601145 63.5734017695368 67.5985758420015 68.3191013228736   
## 1 1 1 1   
## 69.3624871531261 69.5694245290586 71.3068589744744 72.9153891887297   
## 1 1 1 1   
## 73.7078441592678 75.6752820865587 79.5004272070883 81.2260795432364   
## 1 1 1 1   
## 82.0674236022813 82.8056713897051 83.1162942350465 83.9955992771917   
## 1 1 1 1   
## 88.904056980776 93.1942094258462 96.0542215310119 97.0401571339381   
## 1 1 1 1   
## 104.3970600926 111.370780759869 119.973523240895 120.958387932131   
## 1 1 1 1   
## 121.223773615909 131.340252777407 167.826028178834 173.434226090422   
## 1 1 1 1

sort(table(week2.dataset)) #mode

## week2.dataset  
## -40.8521479536028 -37.0600861729767 -27.4394707301933 -26.6136497602233   
## 1 1 1 1   
## -26.0247329774009 -23.9445014953885 -18.7855464905273 -14.1606871877022   
## 1 1 1 1   
## -12.6314747958724 -8.36451775798208 -8.28012876527099 -3.42861381172755   
## 1 1 1 1   
## 1.3707773689551 1.49901558852403 4.59350826279258 4.87557297527698   
## 1 1 1 1   
## 5.54762996020345 5.65844114537842 5.72152565085104 6.46624258843072   
## 1 1 1 1   
## 7.63136435374992 7.6665353205413 8.64266896440952 10.1207250154375   
## 1 1 1 1   
## 10.4199778003196 11.0700577773358 11.6835420015188 11.6893630606738   
## 1 1 1 1   
## 15.3364627730003 15.5180333210577 16.4986305405722 16.7777160094759   
## 1 1 1 1   
## 18.8207371660071 23.4330615097242 28.6933296446239 30.6774975685442   
## 1 1 1 1   
## 31.4598853079133 32.8692047221264 32.8944598914031 35.209280549363   
## 1 1 1 1   
## 35.8143349945542 37.264567873172 37.6029282370497 37.7234274438144   
## 1 1 1 1   
## 38.4258862011859 38.5560833533736 39.3195742083734 40.0814200159895   
## 1 1 1 1   
## 41.5534743751608 41.6631636365883 41.9164849893844 42.334678712197   
## 1 1 1 1   
## 43.2176289187863 45.5361096844647 45.6405496134666 48.1864040055618   
## 1 1 1 1   
## 48.5687737780364 49.6112824985527 49.9919936657234 50.5433676890076   
## 1 1 1 1   
## 50.6154894589018 53.0570483207756 53.3988077408265 53.5607896751654   
## 1 1 1 1   
## 54.3027088373517 54.5732346903718 56.0473977911211 57.2779564604667   
## 1 1 1 1   
## 59.6476416938104 60.4759949947662 61.5977879611046 62.5037970633328   
## 1 1 1 1   
## 63.1921073601145 63.5734017695368 67.5985758420015 68.3191013228736   
## 1 1 1 1   
## 69.3624871531261 69.5694245290586 71.3068589744744 72.9153891887297   
## 1 1 1 1   
## 73.7078441592678 75.6752820865587 79.5004272070883 81.2260795432364   
## 1 1 1 1   
## 82.0674236022813 82.8056713897051 83.1162942350465 83.9955992771917   
## 1 1 1 1   
## 88.904056980776 93.1942094258462 96.0542215310119 97.0401571339381   
## 1 1 1 1   
## 104.3970600926 111.370780759869 119.973523240895 120.958387932131   
## 1 1 1 1   
## 121.223773615909 131.340252777407 167.826028178834 173.434226090422   
## 1 1 1 1

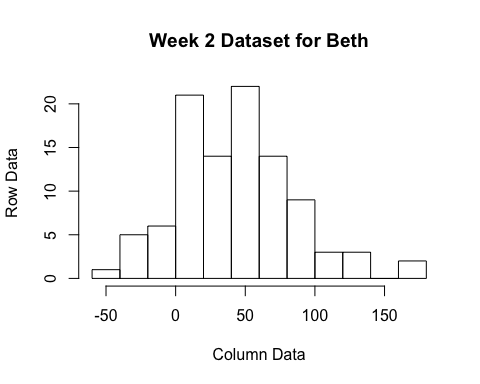
To find the mean, I added all of the values together and then divided them by the total number of values in the set (100). The mean is 43.09519.

To find the median, I then sorted the data into a new subset, which sorted the data in ascending order. Then, found the total number of variables (100) and divided it in half to find the median variable, which is variable 50. This makes the median 41.663164.

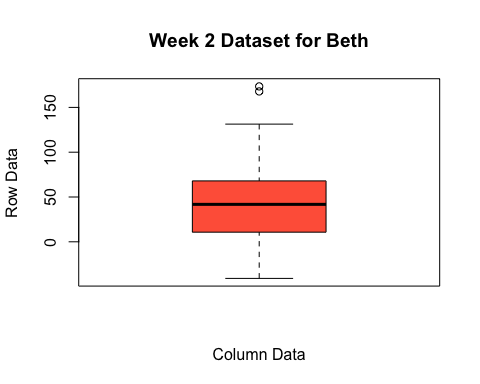
To find the mode, you view the table and sort it. As can be seen above, there is no variable with the highest frequency, therefore, they are all "the mode."

## PC5. Create a number of visualizations of your dataset: at the very least, create a boxplot and histogram

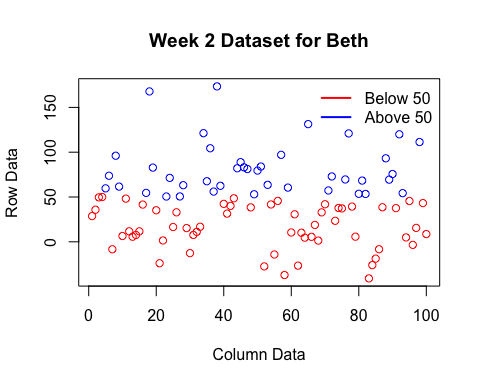
#historgram  
hist(week2.dataset, ylab="Row Data", xlab="Column Data", main="Week 2 Dataset for Beth")



#boxplot  
boxplot(week2.dataset, col=c("tomato"), vertical=TRUE, ylab="Row Data", xlab="Column Data", main="Week 2 Dataset for Beth")



#plot  
above50 <- ifelse(week2.dataset<50, 1,0)  
plot(week2.dataset, ylab="Row Data", xlab="Column Data", main="Week 2 Dataset for Beth", col=ifelse(above50==1, "red", "blue"))  
legend("topright", c("Below 50", "Above 50"), bty="n", lwd=2, col=c("red", "blue", pch=c(1)))

 Above, you can see the historgram, boxplot, and scatterplot results for the Week 2 Dataset.

## PC6. Some of you will have negative numbers. Whoops! Those were not suppose to be there. Recode all negative numbers as missing (i.e., NA) in your dataset. Now create and compute a new mean and standard deviation. How does it change? (Hint: the basic mean function will give you an error. You have to use a named argument (na.rm=TRUE) to work around this).

negnumbers <- week2.dataset  
negnumbers[negnumbers<0] <- NA  
mean(negnumbers, na.rm = TRUE)

## [1] 51.78528

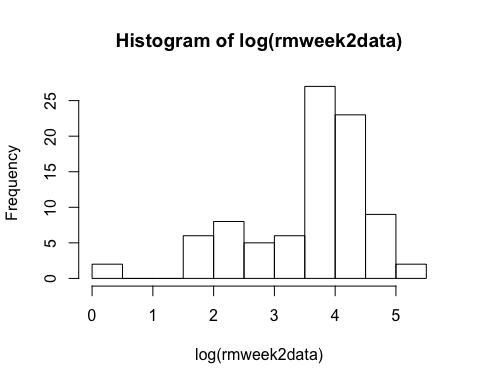
sd(negnumbers, na.rm = TRUE)

## [1] 36.35844

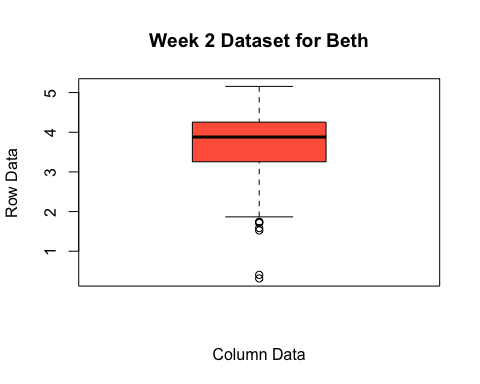
As can be shown above, once you remove the missing (negative) values, the mean goes up, and the standard deviation goes down. This means, that the mean of all the data is actually higher once the negative numbers are removed, as well as the standard deviation from the mean becomes smaller. This makes sense, as you are removing variables that bring the total number down (no more negative numbers) and increase distance from the mean

## PC7. Log transform your dataset. Create new histograms, boxplots, and means, median, and standard deviations.

rmweek2data <- na.omit(negnumbers)   
hist(log(rmweek2data)) #histogram



boxplot(log(rmweek2data), col=c("tomato"), vertical=TRUE, ylab="Row Data", xlab="Column Data", main="Week 2 Dataset for Beth") #boxplot



mean(log(rmweek2data)) #mean

## [1] 3.594182

median(log(rmweek2data)) #median

## [1] 3.879029

sd(log(rmweek2data)) #standard deviation

## [1] 1.006541

.