HW2

ME

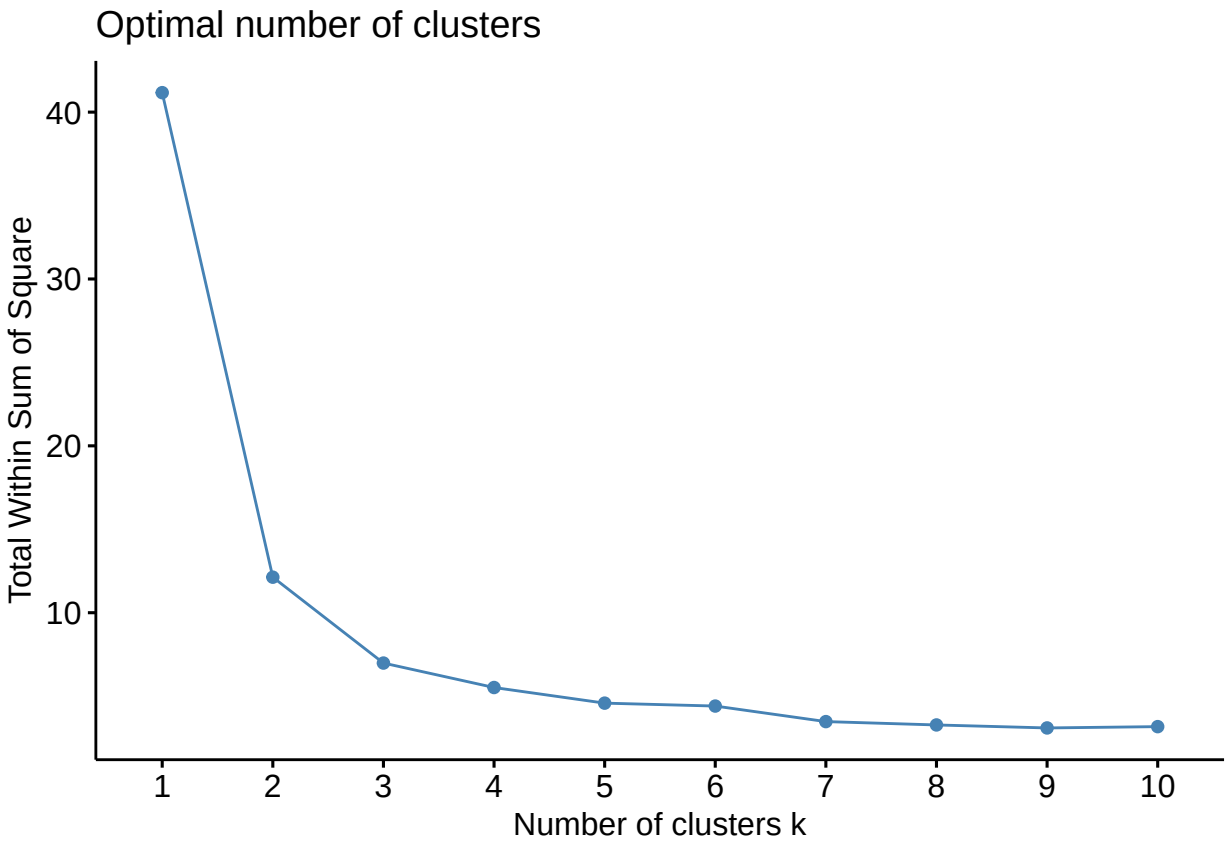
5/20/2020

Question 4.1

As elections ramp up clustering could be utilized to target likely voters of different candidates for ads. You could use income, location of residence, education level, age and a long list of other things to determine who would be the appropriate targets.

Question 4.2

```
library(datasets)
library(ggplot2)
library(NbClust)
library(factoextra) #this is for visualizing distance #https://uc-r.github.io/kmeans_clustering?
library(dplyr)
data(iris)
iris <- iris[order(iris$Species),]
#summary(iris)
#kiris <- scale(iris[-5])
kiris <- apply(iris[-5], MARGIN = 2, FUN = function(X) (X - min(X))/diff(range(X))) #https://stackoverf
#summary(kiris)

fviz_nbclust(kiris, kmeans, method = "wss") #wss = within sum of squares
```



Choosing number of clusters and assessing accuracy

We see here from the elbow chart that 2-3 clusters is a good number we will implement kmeans below with 3 as after that we don't see much benefit.

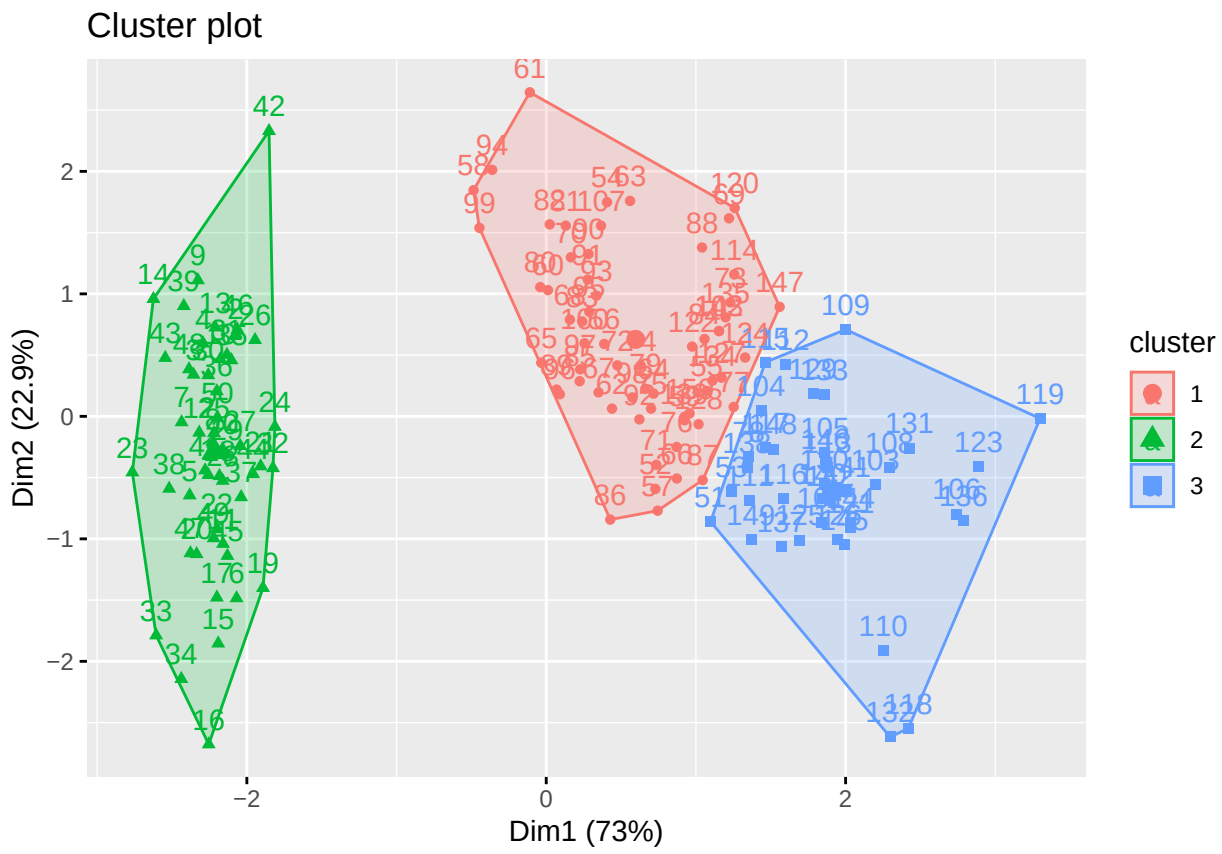
Originally I associated 1 with Setosa, 2, with veriscolor, but when plotting the original data next to the clustered it seemed apparent these two were mixed and I switched them to compute accuracy. There are two method of computing accuracy here. First i did so manually using the created 'accuracy' funciton. I later saw people using confusionMatrix so i wanted to look at that as well, so i added that in

```
library(ggplot2)
library(caret)
{
#summary(kiris)
set.seed(12)
cv <- kmeans(kiris, 3, nstart = 50)
#cv
#cv$size
clusa <-data.frame (cv$cluster)
clusa$ckey[clusa$cv.cluster==2]<- 'setosa'
clusa$ckey[clusa$cv.cluster==1]<- 'versicolor'
clusa$ckey[clusa$cv.cluster==3]<- 'virginica'
clusa$ckey <- as.factor(clusa$ckey)

#summary(clusa)
cm <- confusionMatrix(clusa$ckey, iris$Species)
print(paste('confusionMatrixAccuracy: ', cm$overall['Accuracy']))
print (cm$table)
```

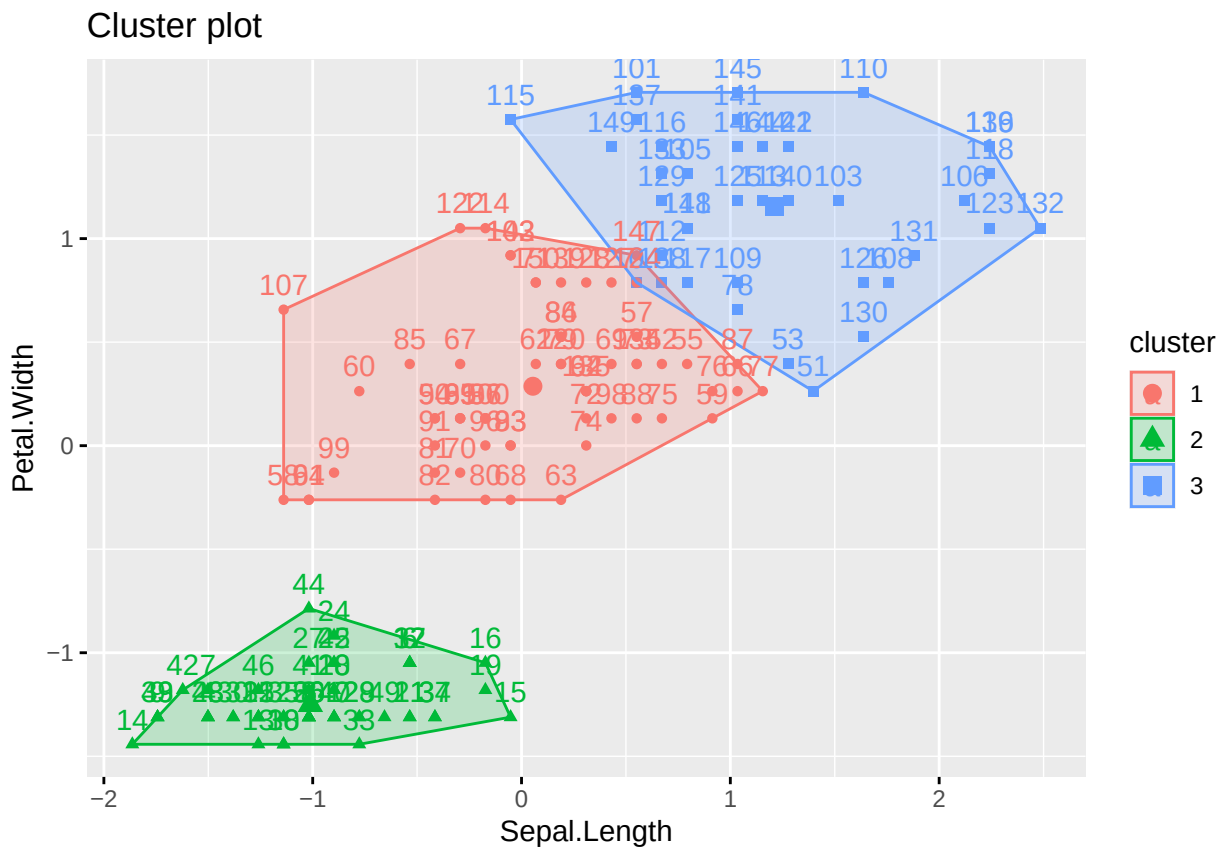
```r
#Function create a dataframe with the cluster vector and then work in the actual values to check accura
accuracy <- function(cv, odf) {

  clusa <-data.frame (cv$cluster)
  clusa$key <- odf$Species #creating new columns :https://stackoverflow.com/questions/31150764/how-to-c
  #clusa$ckey[clusa$key=='setosa']<- 1
  #clusa$ckey[clusa$key=='veriscolor']<- 2
  #clusa$ckey[clusa$key=='virginica']<- 3

  clusa$ckey[clusa$cv.cluster==2]<- 'setosa'
  clusa$ckey[clusa$cv.cluster==1]<- 'versicolor'
  clusa$ckey[clusa$cv.cluster==3]<- 'virginica'

  clusa$acc[clusa$ckey==clusa$key]<-1
  clusa$acc[clusa$ckey!=clusa$key]<-0
  #summary(clusa)
  #View(clusa)
  acc <- sum(clusa$acc)
  return (acc)
}
acc <- accuracy(cv, iris)
print(paste('Accuracy of : ', (sum(acc)/150)*100 ))
#print(clusa)
}
```

```
## [1] "confusionMatrixAccuracy:  0.886666666666667"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         47        14
##   virginica       0          3        36
## [1] "Accuracy of :  88.6666666666667"
```

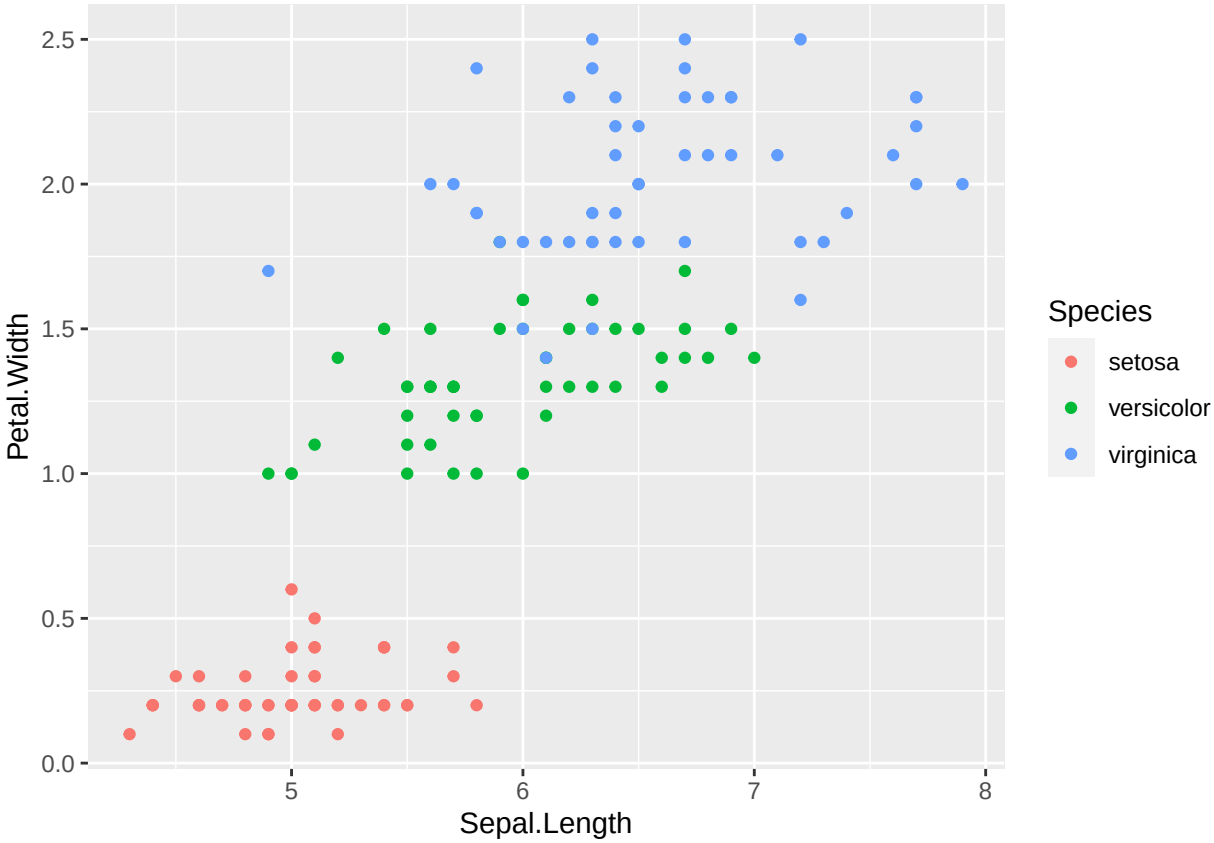
```r
fviz_cluster(cv, data = kiris)
```




```
fviz_cluster(cv, data = kiris, choose.vars = c("Sepal.Length", "Petal.Width"))
```




```
ggplot(iris, aes(x=Sepal.Length, y = Petal.Width, color = Species )) + geom_point()
```




```
## what is 'dim1' and 'dim2'? #https://stats.stackexchange.com/questions/263374/clusters-and-data-visua
###rtools not found remedy:https://stackoverflow.com/questions/33103203/rtools-is-not-being-detected-fr
```

next steps

Utilizing all the data points we see 88.67% accuracy. Below we will try to identify the best predictors to increase this accuracy. To do this, i created a dataframe of the possible combinations of 2/4 measures and then used that to iterate through kmeans and measure the success of each combination.

```
# create all the combinations of variables
#cols <- do.call(expand.grid, rep(list(c(F, T)), ncol(kiris)))[-1,]
#View(cols)
cols <-colnames(kiris)
coldf <-data.frame(t(combn(cols,2))) #creates DF of the possible combinations #https://stackoverflow.co
#print(cols)
#summary(kiris)

itkm <- function(df1, dfo){
  for (i in 1:nrow(df1)) {
    #lkm >- kmeans(
    xt = (df1[i,1])
    yt = (df1[i,2])
    #message(x)
    #message(y)
    #View(kiris[,xt])
```

```
    #tkiris <- data.frame(x1= integer(), x2 = integer())
    #tkiris$x1 <-NA
    tkiris <- data.frame(dfo[,1])
    tkiris$x1 <- dfo[,xt]
    tkiris$x2 <- dfo[,yt]
    tkiris <- tkiris[,-1]
    #View(tkiris)
    #summary(tkiris)
    #View(tkiris)
    set.seed(12)
    lkm <- kmeans(tkiris,3, nstart = 25)
    print(paste(xt,' ', yt))
    print(paste(' -- tot.withinss:', lkm$tot.withinss))
    print(paste(' -- betweenss/totss:', (lkm$betweenss/lkm$totss)*100))
    lkm
    clusa2 <-data.frame (lkm$cluster)
    clusa2$ckey[clusa2$lkm.cluster==1]<- 'setosa'
    clusa2$ckey[clusa2$lkm.cluster==2]<- 'versicolor'
    clusa2$ckey[clusa2$lkm.cluster==3]<- 'virginica'
    clusa2$ckey <- as.factor(clusa2$ckey)
    cm <- confusionMatrix(clusa2$ckey, iris$Species)
    print(paste('CF.overallAccuracy: ', cm$overall['Accuracy']))
    print(cm$table)
    print('_____________________________________________________________')
    cm

  }
  }

itkm(coldf, kiris)
```

```
## [1] "Sepal.Length   Sepal.Width"
## [1] " -- tot.withinss: 3.31264607889437"
## [1] " -- betweenss/totss: 81.85174557546"
## [1] "CF.overallAccuracy:  0.48"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0          8        36
##   virginica       0         42        14
## [1] "_____________________________________________________________"
## [1] "Sepal.Length   Petal.Length"
## [1] " -- tot.withinss: 4.11519420275407"
## [1] " -- betweenss/totss: 67.8444110330148"
## [1] "CF.overallAccuracy:  0.246666666666667"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa          0         13        34
##   versicolor      1         37        16
##   virginica      49          0         0
## [1] "_____________________________________________________________"
## [1] "Sepal.Length   Petal.Width"
## [1] " -- tot.withinss: 1.82628121729684"
```

```
## [1] " -- betweenss/totss: 81.4190755180772"
## [1] "CF.overallAccuracy:  0.293333333333333"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         18         28        34
##   versicolor      1         21        11
##   virginica      31          1         5
## [1] "------------------------------------------------------------"
## [1] "Sepal.Width   Petal.Length"
## [1] " -- tot.withinss: 2.72924511229799"
## [1] " -- betweenss/totss: 87.139675285248"
## [1] "CF.overallAccuracy:  0.0466666666666667"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa          0         40        16
##   versicolor      0          7        34
##   virginica      50          3         0
## [1] "------------------------------------------------------------"
## [1] "Sepal.Width   Petal.Width"
## [1] " -- tot.withinss: 3.31264607889437"
## [1] " -- betweenss/totss: 81.85174557546"
## [1] "CF.overallAccuracy:  0.48"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0          8        36
##   virginica       0         42        14
## [1] "------------------------------------------------------------"
## [1] "Petal.Length   Petal.Width"
## [1] " -- tot.withinss: 4.11519420275407"
## [1] " -- betweenss/totss: 67.8444110330148"
## [1] "CF.overallAccuracy:  0.246666666666667"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa          0         13        34
##   versicolor      1         37        16
##   virginica      49          0         0
## [1] "------------------------------------------------------------"
```

Assessment of predictors and accuracy

Viewing the results of the confusion matrix it seems that 'Sepal length' is the least effective classifier as the accuracy with that included was relatively low. Sepal Width seemed to be associated with the 'best' accuracies that I saw here. According to my confusion matrix, using only two classifiers i see an accuracy of 88.67 when using 'sepal width' and 'petal width', or 'sepal width' and 'sepal length'. Below we see an accuracy of 90% while excluding only Sepal Length.

Note, by setting nstart to a higher number it seems that the clusters remain in the 'correct' order, ie: 1=setosa, 2= versicolor etc. if you notice a lower accuracy than stated above, it is likely due to misassignment and can be corrected by switching it as stated in the code below.

```
 set.seed(12)
   # head(kiris)
{
```

```
    tkiris2 <- data.frame(kiris[,-2])
   # head(tkiris2)
    lkm <- kmeans(tkiris2,3, nstart=25)
    print(colnames(tkiris2))
    print(paste(' -- tot.withinss:', lkm$tot.withinss))
    print(paste(' -- betweenss/totss:', (lkm$betweenss/lkm$totss)*100))


    #lkm
    clusa2 <-data.frame (lkm$cluster)
    clusa2$ckey[clusa2$lkm.cluster==1]<- 'setosa'  #if low accuracy change the cluster to '2'
    clusa2$ckey[clusa2$lkm.cluster==2]<- 'versicolor' #if low accuracy change the cluster to '1'
    clusa2$ckey[clusa2$lkm.cluster==3]<- 'virginica'
    clusa2$ckey <- as.factor(clusa2$ckey)
    cm <- confusionMatrix(clusa2$ckey, iris$Species)
    print(paste('CF.overallAccuracy: ', cm$overall['Accuracy']))
    print(cm$table)
    #cm
}
```

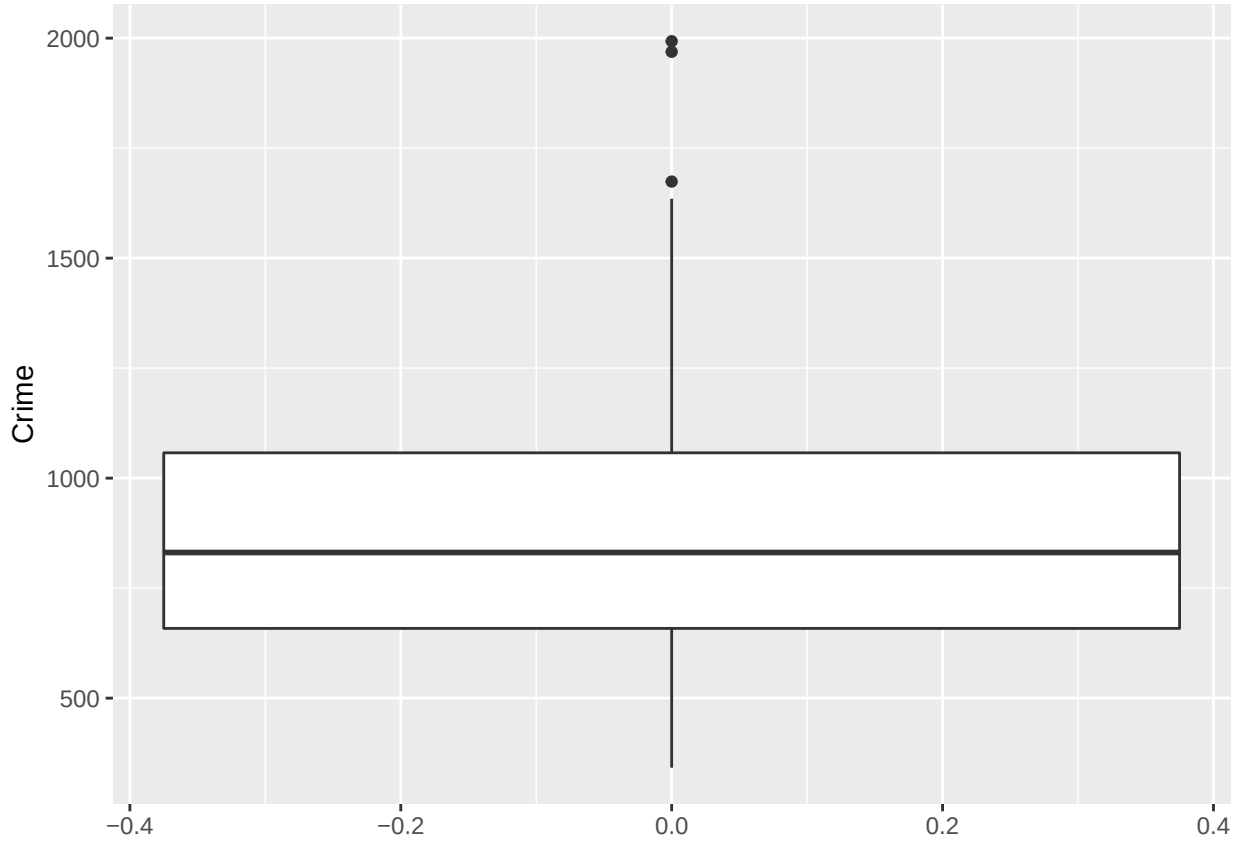
```
## [1] "Sepal.Length" "Petal.Length" "Petal.Width"
## [1] " -- tot.withinss: 4.30911821043673"
## [1] " -- betweenss/totss: 88.1133388184277"
## [1] "CF.overallAccuracy:  0.9"
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48        13
##   virginica       0          2        37
```

Question 5.1

Grubbs test requires normal data, I use Shapiro-Wilks Normality test and get a p-value of .001882, indicating the data is not normally distrubted. To identify what is cause the issue I can plot this, I used a box&whisker plot. I removed rows where crime was greater than 1900. I then ran another shapiro-wilks on the 'clean' data and it appears the data is normally distrubuted with a p-value >.05. i then ran a grubbs test and see a p-value of .1781, indicating that there are no remaining outliers. I think there could be arguments to include this data in certain circumstances, the differences in Prob and Time (or other factors) could account for the higher crime here (for instance the 2nd highest crime value also had one of the lowest Prob and an average Time).

```
library(ggplot2)
library(outliers)
library(dplyr)
{
crime <- read.csv(url("http://www.statsci.org/data/general/uscrime.txt"), sep = "\t")
#crime <-data.frame(crime)
#head(crime)
shapiro.test(crime$Crime)
#grubbs.test(crime$Crime, type =11 )
```

```
#ggplot(crime, aes(x=Crime)) +geom_histogram()
ggplot(crime, aes(y=Crime)) +geom_boxplot()
}
```




```
{
crimec <- filter(crime, Crime<1900)
shapiro.test(crimec$Crime)
}
```

```
##
##  Shapiro-Wilk normality test
##
## data:  crimec$Crime
## W = 0.95119, p-value = 0.05634
```

```
{
ggplot(crimec, aes(y=Crime)) +geom_boxplot()
grubbs.test(crimec$Crime, type =10 )
#View(crimec)
}
```

```
##
##  Grubbs test for one outlier
##
```

```
## data:  crimec$Crime
## G = 2.56457, U = 0.84712, p-value = 0.1781
## alternative hypothesis: highest value 1674 is an outlier
```

Question 6.1

I could use CUSUM to detect an increase in the number of COVID patients at work. This is important to know because it can have a big impact on staffing and on OR/room usage as well as demand for equipment. If the new covid patient count was 2 std dev above the ~10 day mean, I would say that is cause for concern.

Question 6.2

I pull in the data and use the first 80 days to create a mean and stdev for my cusum for each year (stored in 'sta'). I use the mean of the 80 days as 'u' and 1 standard deviation as the allowable tolerance (sl), I set T to 5*stdev. 'xt' is the measured temperature at time 't', 'u' is as defined above as is 'sl'. i run a nested for loop to iterate through each cell and and do (-xt+u-sl). Next i do a similar process to get a CUSUM, a few examples are plotted below for the year 1996,1999 and 2013 (you should be able to change the y to different columns if you'd like to see other years). 1996 looks like I would expect, 1999 almost had a 'false' positive and in 2013 we see an August 15 'end of summer' and then the temperature quickly goes back to normal. When looking at 1999, i noticed the spike and raised my T, I think the tolerance for being a couple days 'late' is high and that the tolerance of more false positives was low, thus raising 'T' made sense to me (visually, it looks like the end of summer is only put off by a few days at most with the higher T)

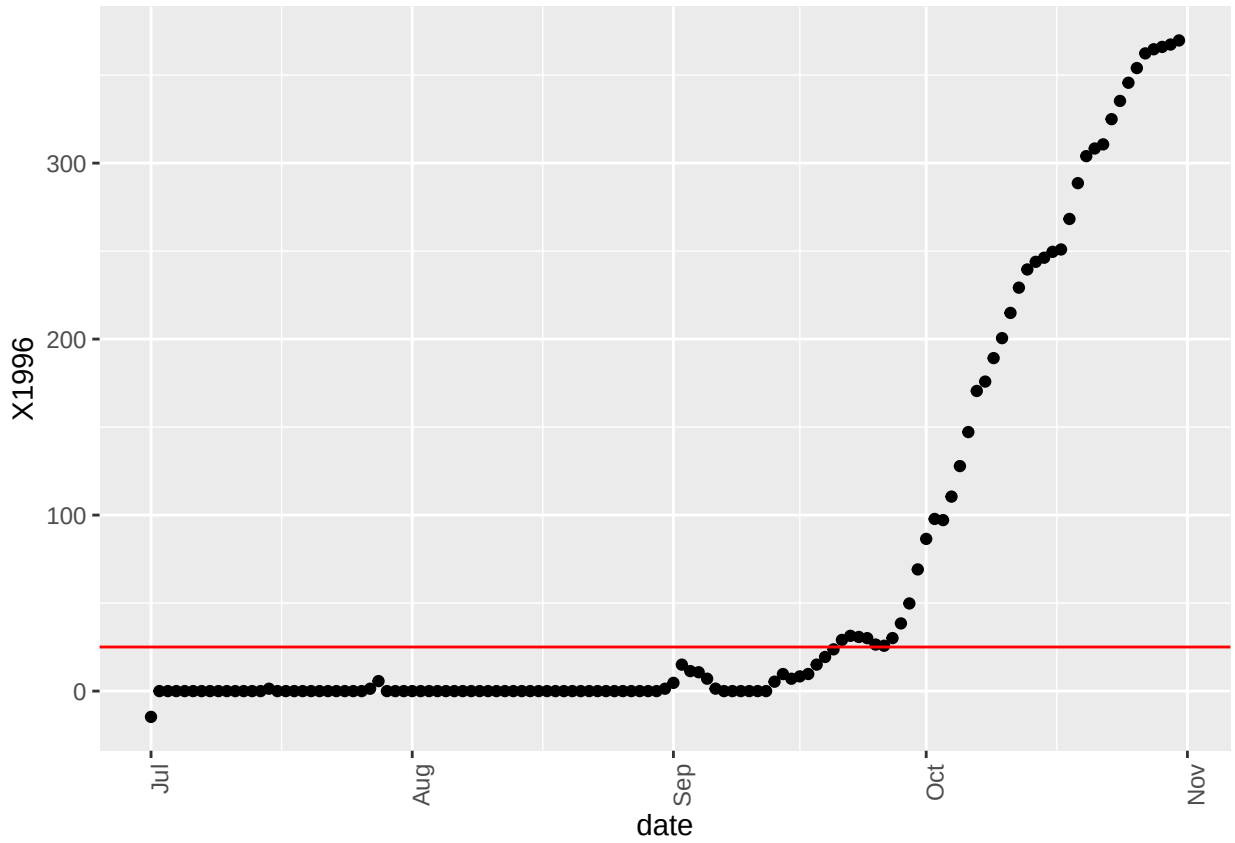
```
library(ggplot2)
library(outliers)
library(dplyr)
library(tidyr)
library(purrr)
library(lubridate)
{
mywd <- "C:\\Users\\Eddie\\Dropbox\\GT\\Classes\\ISYE6501\\Coding_And_data\\Week_2" #set your WD to tes
setwd(mywd)
tempdata <- read.delim("temps.txt") #this is the raw temperature data
#head(tempdata)
#summary(tempdata)

ts <- tempdata[1:80,] #I am using the 80 days to calculate a mean temp and standard deviation

sta <- data.frame(colMeans(ts[-1])) #mean
sta$sd <- apply(ts[-1],2,sd) #stdev

cusum <-tempdata #set up a df to store our measures

#this is getting the measure above/below our range of tolerance
for (j in 2:ncol(tempdata)){

  for (i in 1:nrow(tempdata[j])) {
    xt <- tempdata[i,j]
    u <- sta[j-1,1]
    sl <- sta[j-1,2]
    cusum[i,j] <- (-xt+u-sl)
```

```
  }
}


#this is going through the measures from last for loops and doing CUSUM
cusumc <- cusum
for (j in 2:ncol(cusum)){

  for (i in 1:nrow(cusum[j])) {
    xt <- cusum[i,j]
    if (i==1){
      St <- xt
    }
    else {
      st <- cusumc[i-1,j] +xt
  if (st<0){
    st <- 0
  }
  cusumc[i,j] <- st
    }
  }
}



#head(sta)
sta$endsummer <- ('zero')
#summary(cusumc)
for (i in 2:ncol(cusumc)){
 # cn <- colnames(cusumc[i])
  xi <- 0
  for (j in 1:nrow(cusumc[i])){

    if (sta$endsummer[i-1] == ('zero')) {
      xi <- xi+1
      cx <- cusumc[j,i]
      sx <- (sta$sd[i-1])*5 #you can change '5' here to other numbers if you want to see the effect

      if (cx > sx) {
        es <-(cusumc$DAY[j]) # if i print es i get a 'DAY' like 21-Sep
        print(paste(colnames(cusumc[i]), ' end of summer:  ', es, '  ','at index: ', xi))

        sta$endsummer[i-1] <- (es) # here when i look at 'sta' i have a seemingly random number (56)
        #print(es)
        #print(sta$endsummer[i-1])

        }
    }

  }
}
```

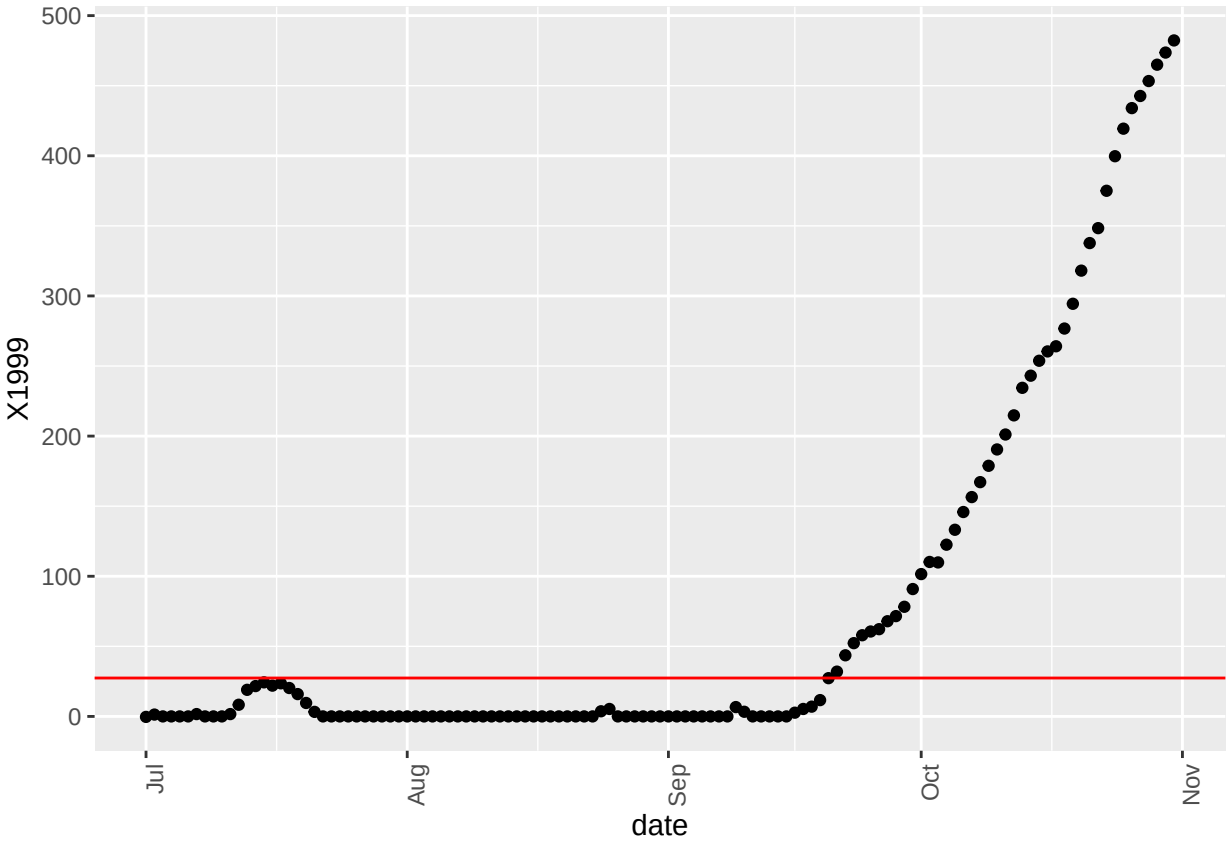
```
cusumc <-cusumc %>% separate(DAY, c("day", "month"), sep = '-', remove = FALSE )

#i use 2020 here to create a date as it wont be displayed and wont have a negative impact
cusumc$date <- paste(2020, cusumc$month, cusumc$day, sep = "-") %>% ymd() %>% as.Date()
#sta$summerend <-

#cusumc %>% keep(is.numeric) %>% gather() %>% ggplot(aes(value)) + facet_wrap(ncol(cusumc), scales = "f
#cusumc <- cusumc[,3]


}
```

```
## [1] "X1996  end of summer:   21-Sep    at index:  83"
## [1] "X1997  end of summer:   25-Sep    at index:  87"
## [1] "X1998  end of summer:   30-Sep    at index:  92"
## [1] "X1999  end of summer:   21-Sep    at index:  83"
## [1] "X2000  end of summer:   9-Sep    at index:  71"
## [1] "X2001  end of summer:   3-Sep    at index:  65"
## [1] "X2002  end of summer:   24-Sep    at index:  86"
## [1] "X2003  end of summer:   28-Sep    at index:  90"
## [1] "X2004  end of summer:   20-Sep    at index:  82"
## [1] "X2005  end of summer:   7-Oct    at index:  99"
## [1] "X2006  end of summer:   20-Sep    at index:  82"
## [1] "X2007  end of summer:   4-Oct    at index:  96"
## [1] "X2008  end of summer:   19-Sep    at index:  81"
## [1] "X2009  end of summer:   2-Oct    at index:  94"
## [1] "X2010  end of summer:   27-Sep    at index:  89"
## [1] "X2011  end of summer:   7-Sep    at index:  69"
## [1] "X2012  end of summer:   1-Oct    at index:  93"
## [1] "X2013  end of summer:   17-Aug    at index:  48"
## [1] "X2014  end of summer:   25-Sep    at index:  87"
## [1] "X2015  end of summer:   16-Sep    at index:  78"
```

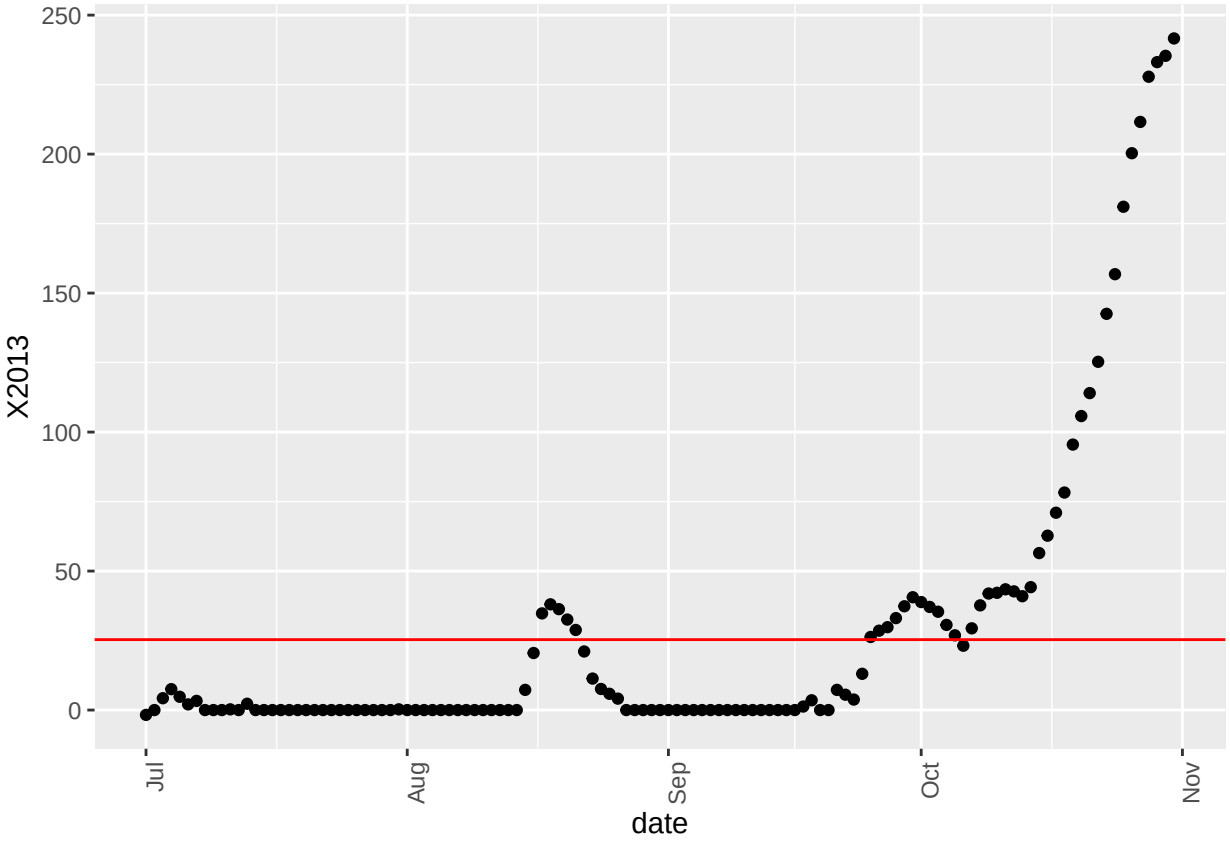
```
ggplot(cusumc, aes(x=date, y=X1996)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hjus
```




```
ggplot(cusumc, aes(x=date, y=X1999)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hjus
```




```
ggplot(cusumc, aes(x=date, y=X2013)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hjus
```



##6.2.2

I will use the first summer mean temperature as the target (u), the standard deviation (sl) as the tolerance. I believe climate is slower moving than seasonal weather so I will use 2 stdev as the threshold. The code below will utilize some of the variables produced above. Utilizing cusum here, we have 6 years that get flagged for change detection. While the argument could be made that the 'warmer' summers are occuring more frequently (50% of the years from 2007-2015 were flagged), overall only 6/18 (33%) of the years measured were flagged as warmer by this CUSUM method. If we utilize more data, we may be able to get a more conclusive answer.

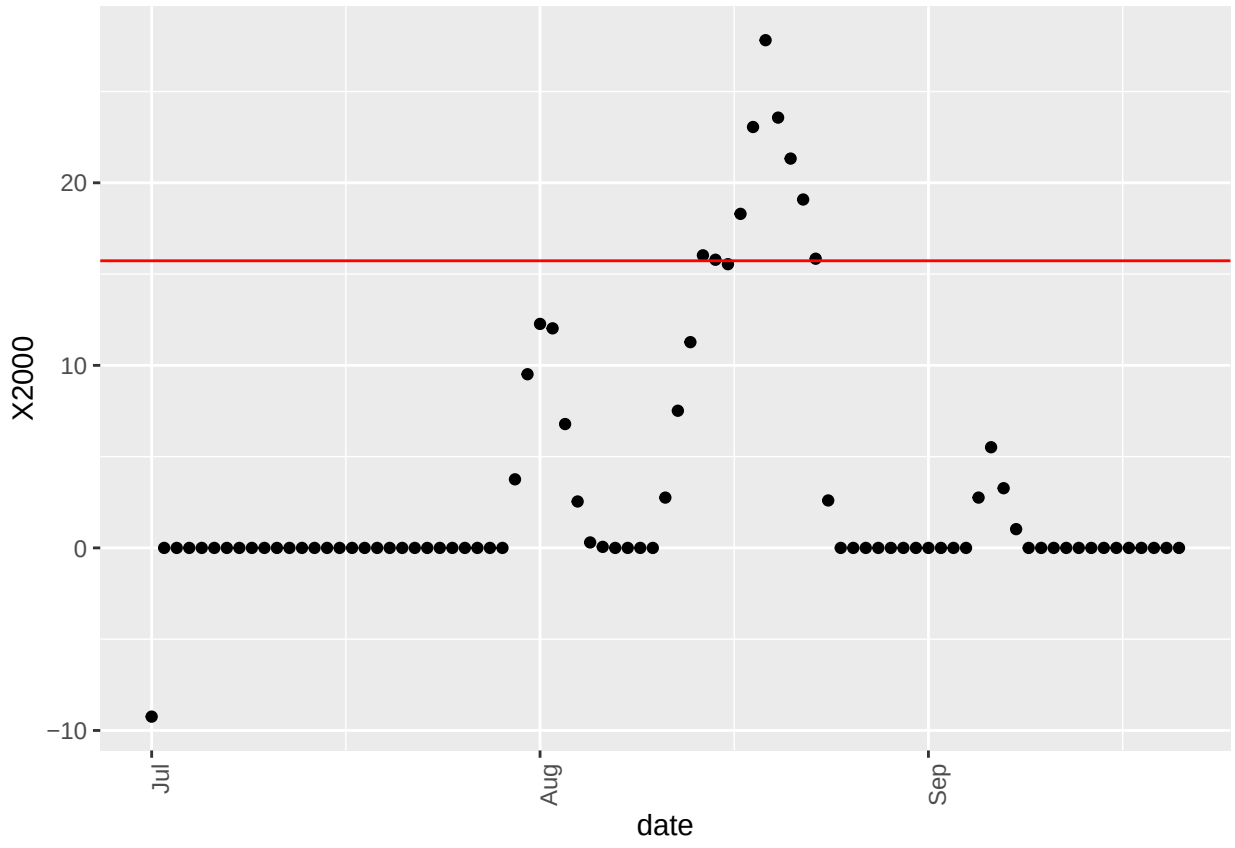
```
{
s1996 <- tempdata[1:83,2]
u1 <- mean(s1996)
sl1 <- sd(s1996) #apply(s1996,2,sd)
t1<- 2*sl1
#tail(tempdata)
cusum2 <- tempdata[-(84:123),]
#View(cusum2)
cusum2 <- cusum2[,-2]
#cusum2 <- cusum2[1:83,]
sumt <- tempdata[-(84:123),]

for (j in 2:ncol(sumt)){

  for (i in 1:nrow(sumt[j])) {
    xt <- sumt[i,j]
    u <- u1
```

```
    sl <- sl1
    cusum2[i,j] <- (xt-u-sl)
  }
}


#this is going through the measures from last for loops and doing CUSUM
cusumc2 <- cusum2
for (j in 2:ncol(cusum2)){

  for (i in 1:nrow(cusum2[j])) {
    xt <- cusum2[i,j]
    if (i==1){
      if (j>2) {
        st <- cusumc2[83,j-1]+xt}
      else {st <- xt }
    }
    else {
      st <- cusumc2[i-1,j] +xt
  if (st<0){
    st <- 0
  }
  cusumc2[i,j] <- st
    }
  }
}

sta$hotsummer <- ('zero')
#summary(cusumc2)
for (i in 2:ncol(cusumc2)){
 # cn <- colnames(cusumc[i])
  xi <- 0
  for (j in 1:nrow(cusumc2[i])){

    if (sta$hotsummer[i-1] == ('zero')) {
      xi <- xi+1
      cx <- cusumc2[j,i]
      sx <- sl1*3 #you can change '5' here to other numbers if you want to see the effect

      if (cx > sx) {
        es <-(cusumc2$DAY[j]) # if i print es i get a 'DAY' like 21-Sep
        print(paste(colnames(cusumc2[i]), ' hot of summer:  ', es, '  ','at index: ', xi))

        sta$hotsummer[i-1] <- (es) # here when i look at 'sta' i have a seemingly random number (56)
        #print(es)
        #print(sta$endsummer[i-1])

        }
    }

  }
}
cusumc2 <-cusumc2 %>% separate(DAY, c("day", "month"), sep = '-', remove = FALSE )
```

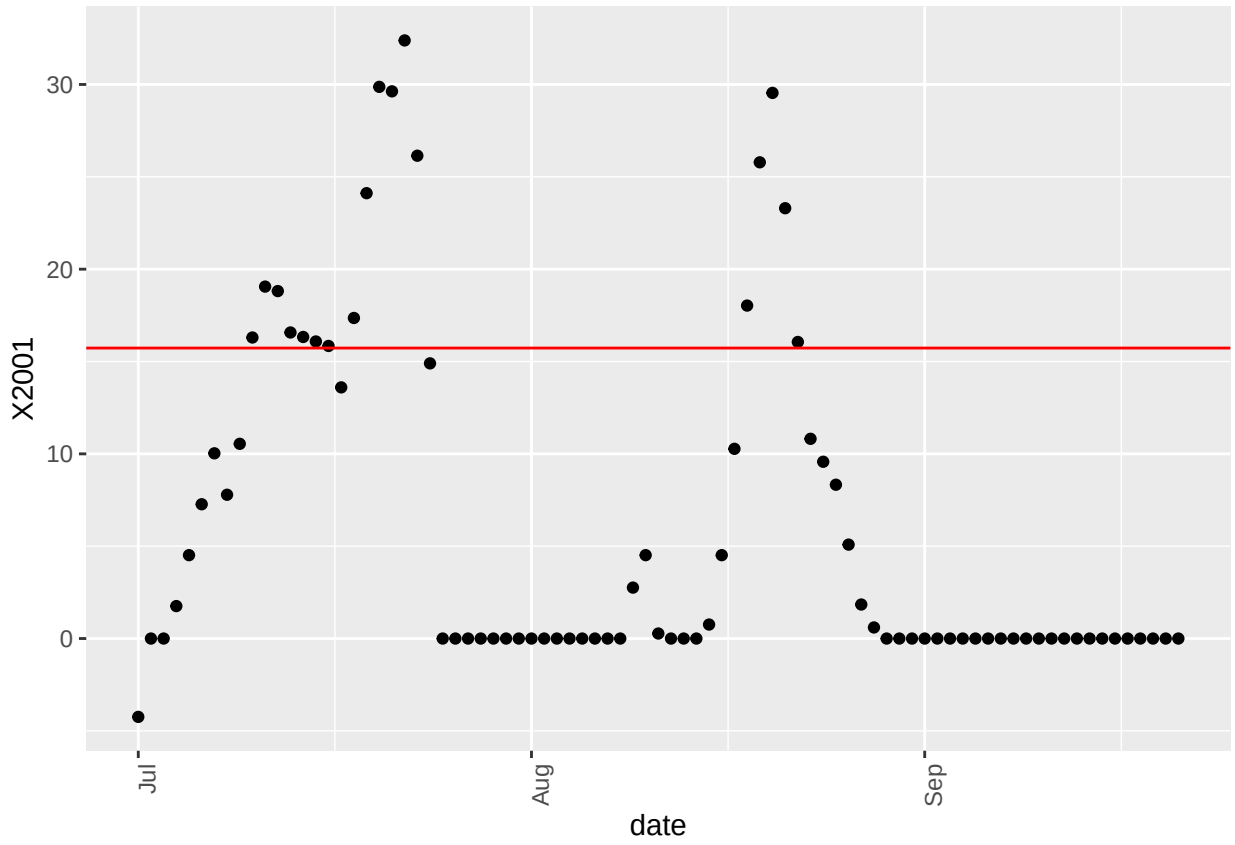
```
#i use 2020 here to create a date as it wont be displayed and wont have a negative impact
cusumc2$date <- paste(2020, cusumc2$month, cusumc2$day, sep = "-") %>% ymd() %>% as.Date()

}
```

```
## [1] "X2000  hot of summer:   14-Aug    at index:  45"
## [1] "X2001  hot of summer:   10-Jul    at index:  10"
## [1] "X2007  hot of summer:   8-Aug    at index:  39"
## [1] "X2008  hot of summer:   7-Aug    at index:  38"
## [1] "X2012  hot of summer:   27-Aug    at index:  58"
## [1] "X2013  hot of summer:   3-Jul    at index:  3"
```

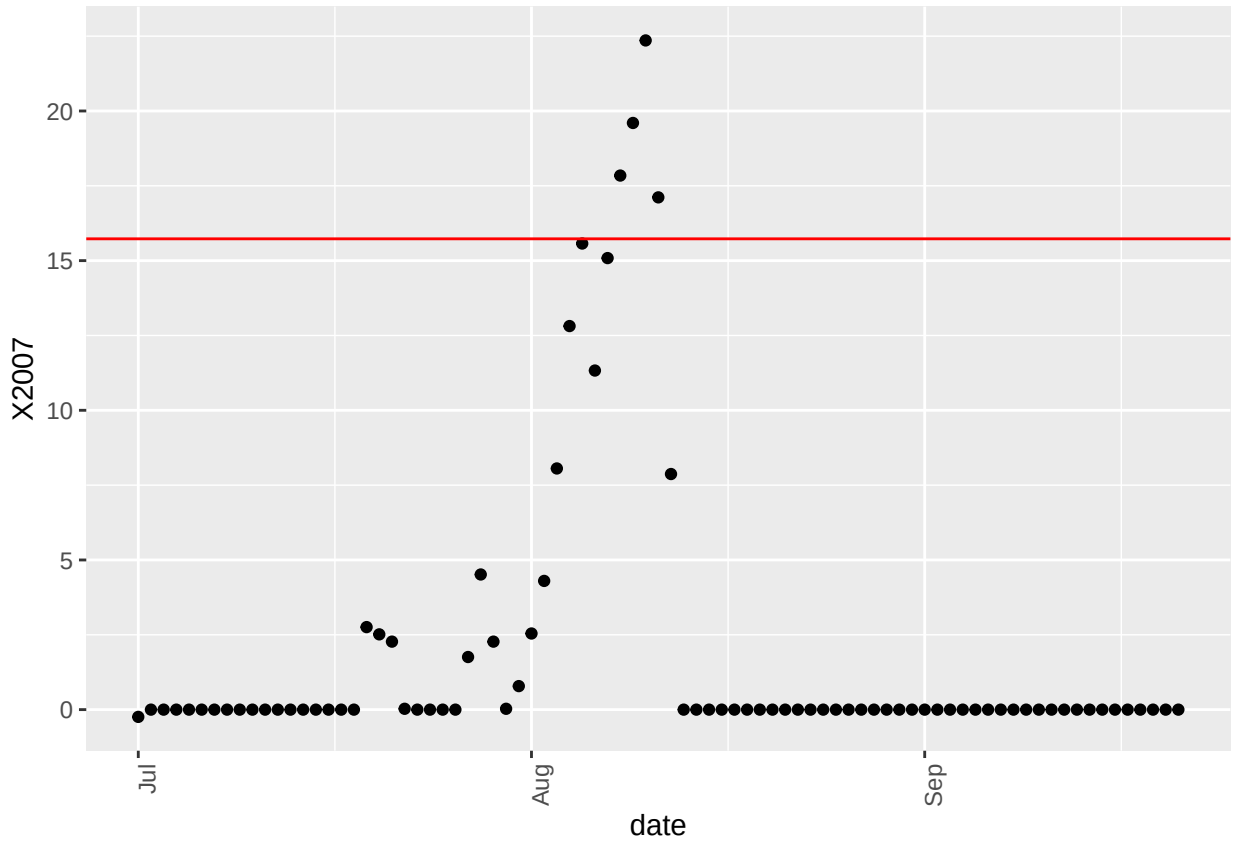
```
ggplot(cusumc2, aes(x=date, y=X2000)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hju
```



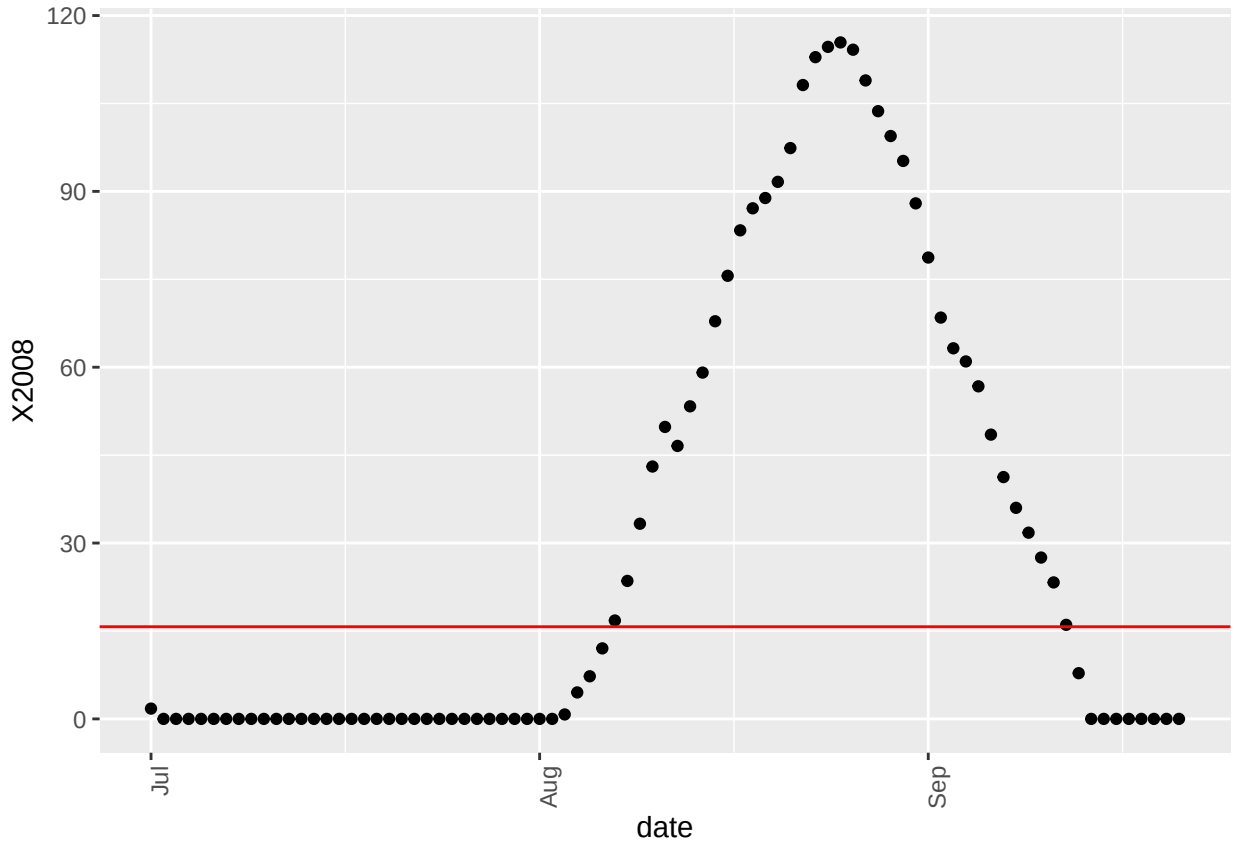

```
ggplot(cusumc2, aes(x=date, y=X2001)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hju
```



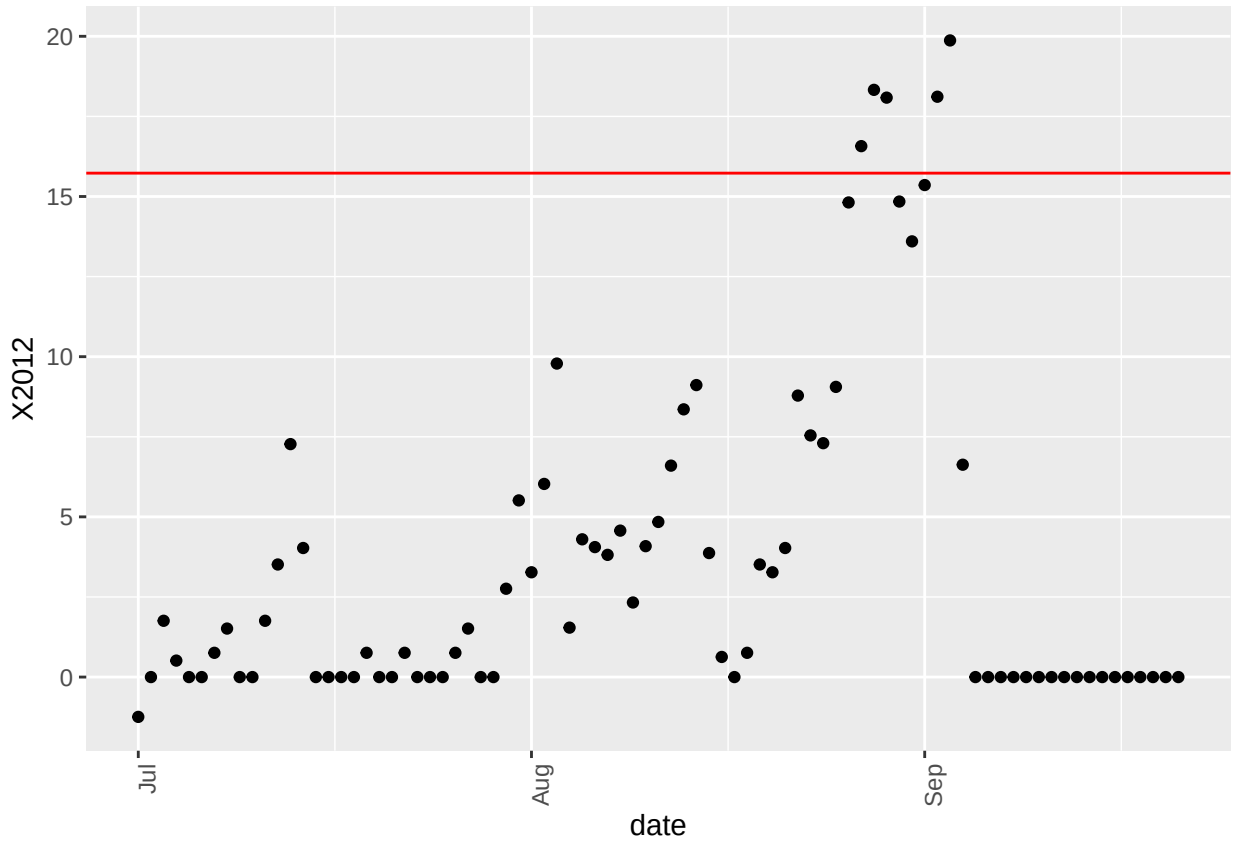

```
ggplot(cusumc2, aes(x=date, y=X2007)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hju
```



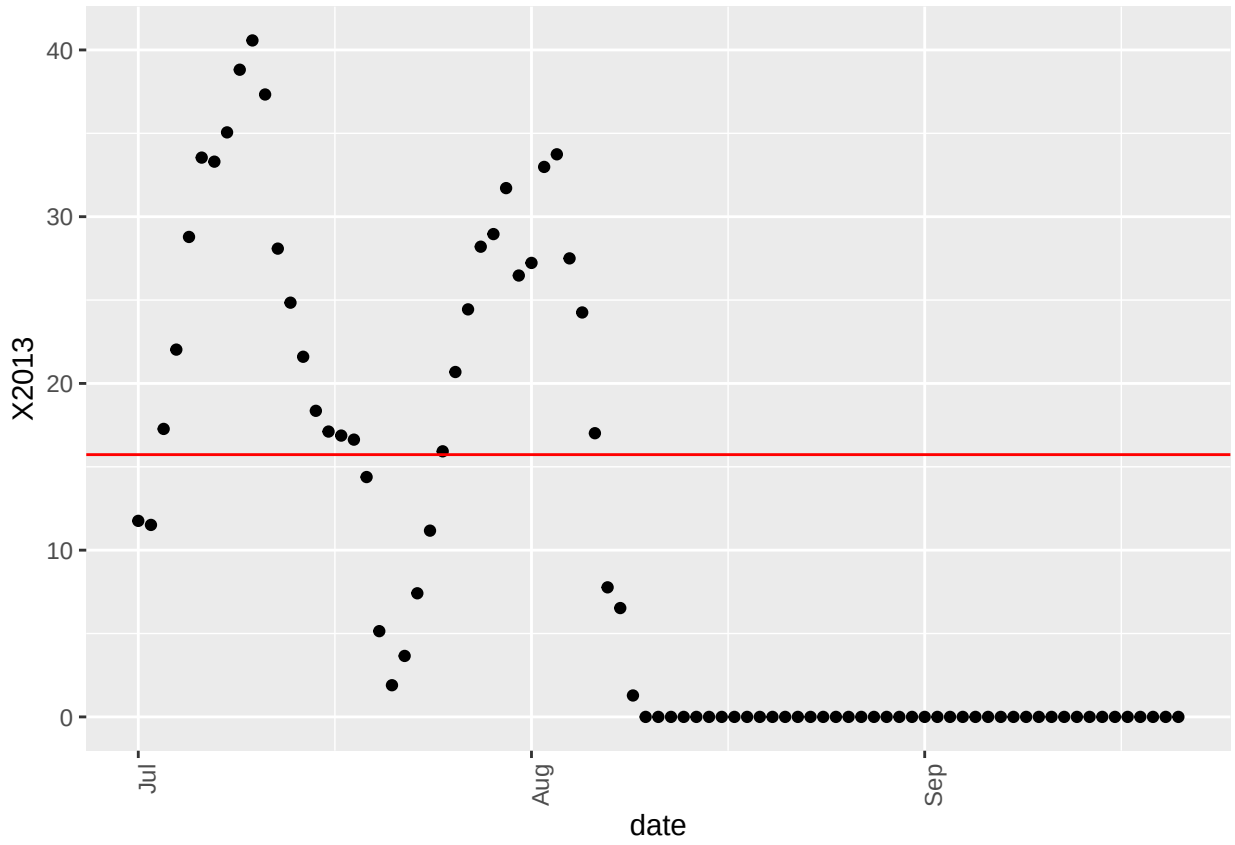

```
ggplot(cusumc2, aes(x=date, y=X2008)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hjus
```




```
ggplot(cusumc2, aes(x=date, y=X2012)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hjus
```




```
ggplot(cusumc2, aes(x=date, y=X2013)) + geom_point() + theme(axis.text.x = element_text(angle = 90, hju
```


40
30
20
10
0
X2013
Jul
Aug
Sep
date