HW2

CSM

5/19/2020

Question 4.1 Describe a situation or problem from your job, everyday life, current events, etc., for which a clustering model would be appropriate. List some (up to 5) predictors that you might use.

```
Clustering is just grouping of similar objects. There are many examples where I think clu
sturing analysis could be used. I think we can use clustreing model for any online Retail
shop. We can group the cumstoers who search and buy products from them. By grouping the c
ustomers they can market the specific products to certrain groups of people. We can use p
redictore's like custonmers'  age, items bought, price, items search for, visit to the si
te etc.
```

For example if one group of people only buying electornics items and other group of people are only buying cosmetics. It make more sence send new deals realted to electorics to the group of people who are buying or searching electronics items from there store.

Question 4.2 The iris data set iris.txt contains 150 data points, each with four predictor variables and one categorical response. The predictors are the width and length of the sepal and petal of flowers and the response is the type of flower. The data is available from the R library datasets and can be accessed with iris once the library is loaded. It is also available at the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Iris (https://archive.ics.uci.edu/ml/datasets/Iris)). The response values are only given to see how well a specific method performed and should not be used to build the model. Use the R function kmeans to cluster the points as well as possible. Report the best combination of predictors, your suggested value of k, and how well your best clustering predicts flower type.

Let’s start with previewing the data. There are 5 predictores but we do not need Species column, hence we will create new dataset without Species column.

```
library(datasets)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Creating new dataset without responce column.

```
my_iris <-iris[,c(1,2,3,4)]
head(my_iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

```
iris_response<- iris[,"Species"]
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.6.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

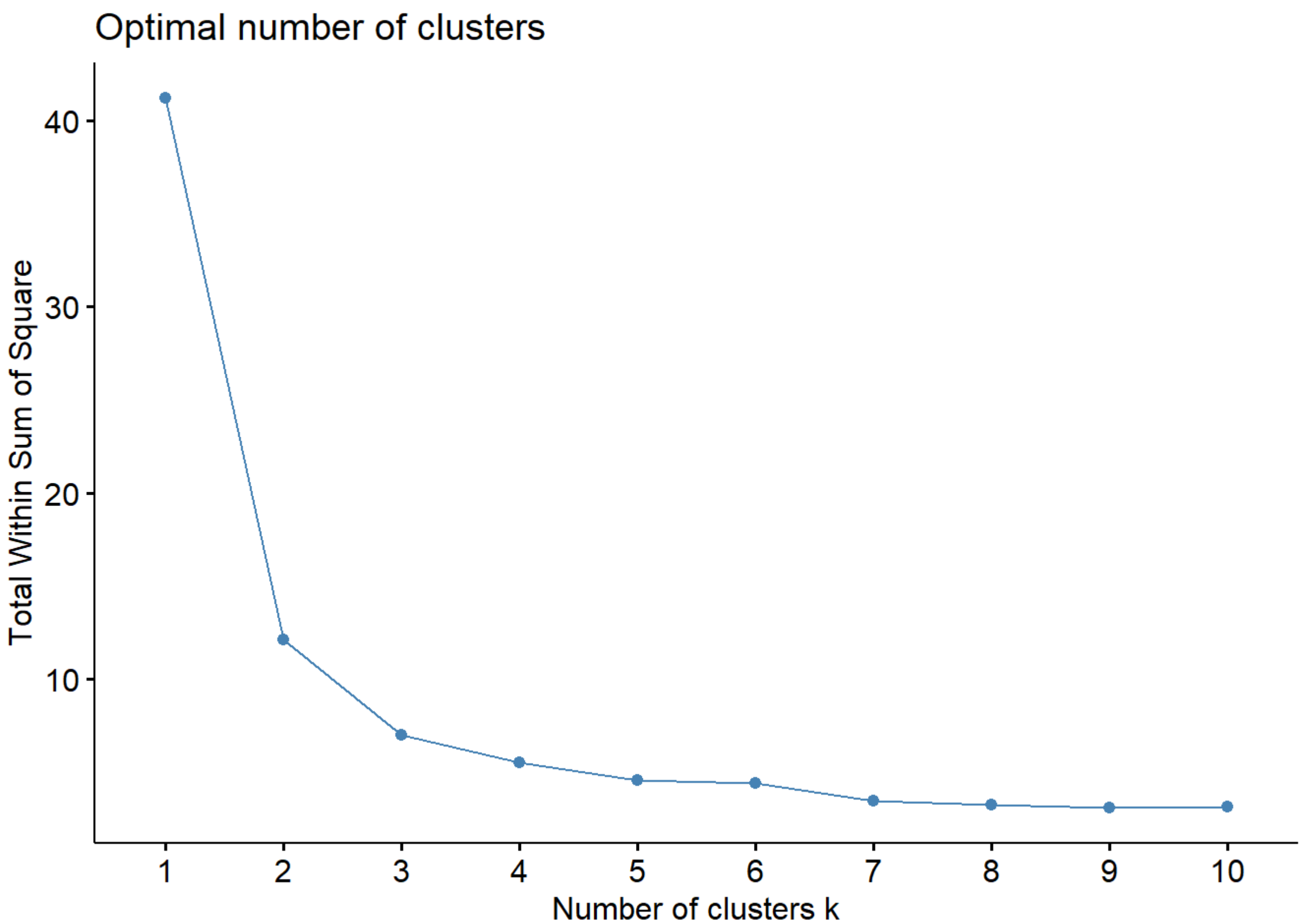
```
iris_scaled <- my_iris
for (i in 1:4) { iris_scaled[,i] <- (my_iris[,i]-min(my_iris[,i]))/(max(my_iris[,i])-min
  (my_iris[,i])) }
head(iris_scaled)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1   0.22222222   0.6250000   0.06779661  0.04166667
## 2   0.16666667   0.4166667   0.06779661  0.04166667
## 3   0.11111111   0.5000000   0.05084746  0.04166667
## 4   0.08333333   0.4583333   0.08474576  0.04166667
## 5   0.19444444   0.6666667   0.06779661  0.04166667
## 6   0.30555556   0.7916667   0.11864407  0.12500000
```

This is very small data set and we alredy know by looking at the data that are three types of flowers so our algorithm should group the data into 3 clusters. That mean's that k =3 should give us the best result. We will try to prove this by ploting the values of k from 1 to 10.

```
k_cluster = rep(0,10)
for(k in 1:10) {
  cluster <- kmeans(iris_scaled, centers=k, nstart=5)
  k_cluster[k] <- cluster$tot.withinss
}

fviz_nbclust(iris_scaled, kmeans, method = "wss")
```



From the above plot(elbow method) it’s clear that K=3 gives us the best result. We will use kmeans function for k=3 to check the accuracy.

```
my_cluster_3 <- kmeans(my_iris, 3, nstart = 25)
my_cluster_3
```

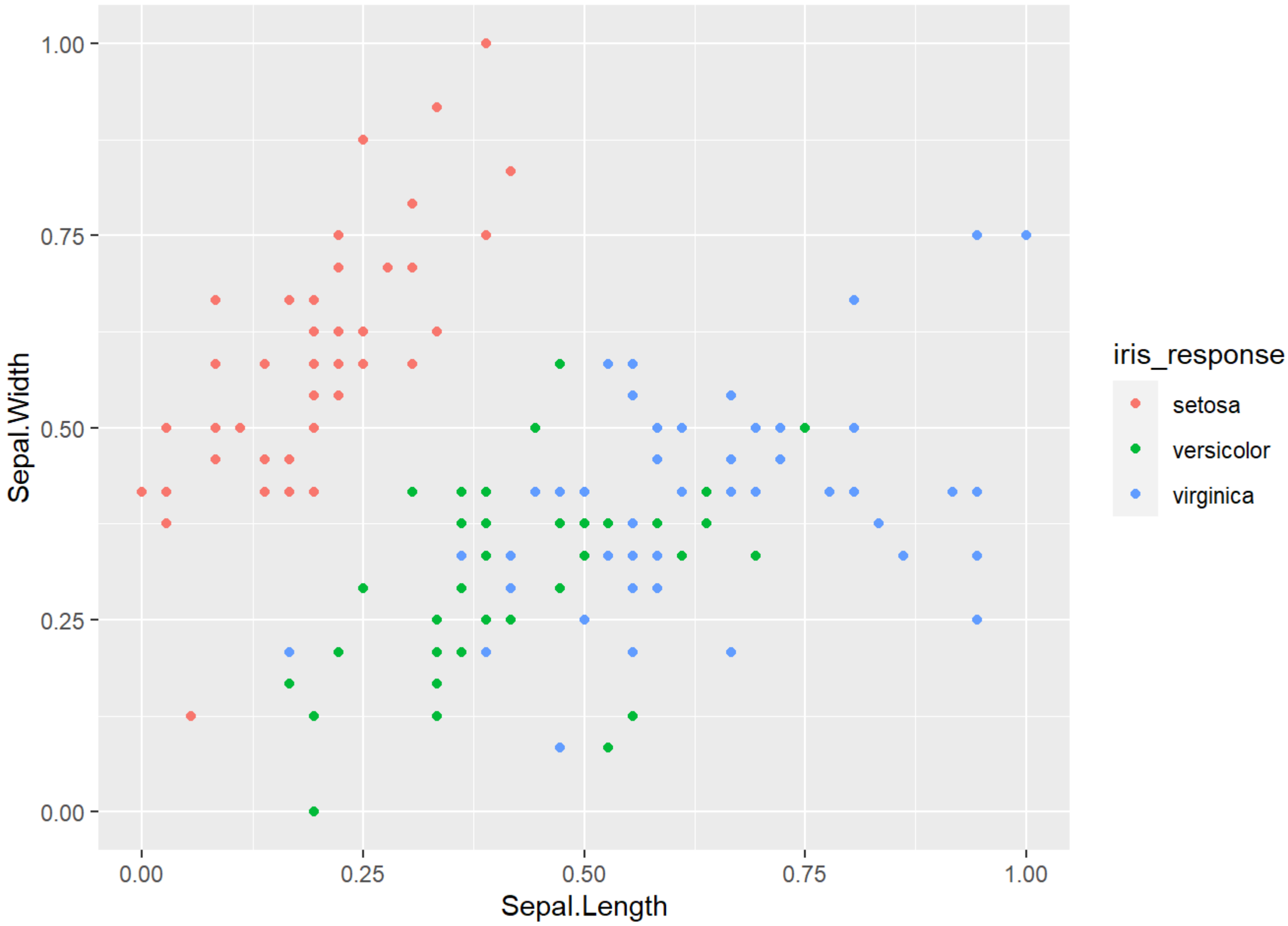
```
## K-means clustering with 3 clusters of sizes 50, 62, 38
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.006000    3.428000     1.462000    0.246000
## 2     5.901613    2.748387     4.393548    1.433871
## 3     6.850000    3.073684     5.742105    2.071053
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3
## [112] 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2 3
## [149] 3 2
##
## Within cluster sum of squares by cluster:
## [1] 15.15100 39.82097 23.87947
##  (between_SS / total_SS =  88.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
overall_accuracy <- my_cluster_3$betweenss / my_cluster_3$totss*100
overall_accuracy
```

```
## [1] 88.42753
```

For k=3 our model gives an accuracy of 88.42%. THis can be clearly seen in the below plots as well.That means that using all 4 predectiors our data can be clusterd into 3 groups with 88% accuracy. We need to check what happens if we use 2 predectiors instead. Will that be able to segergate our data more accurately? We can check what happens if we use two predictores instead of all four. From the below plot we can see that it is hard to group the data into three clusters using sepal length and width beacuse there is lot of ovelapping. This kind of overlapping affects the accuracy of our model. We can run kmeans algorithm on sepal width and length, keeping all other parameters constant. It should give us less accurate result than overall data result. But for the petal width and petal length it should give us more accurate results.

```
ggplot(iris_scaled, aes(Sepal.Length, Sepal.Width, color=iris_response)) +geom_point()
```



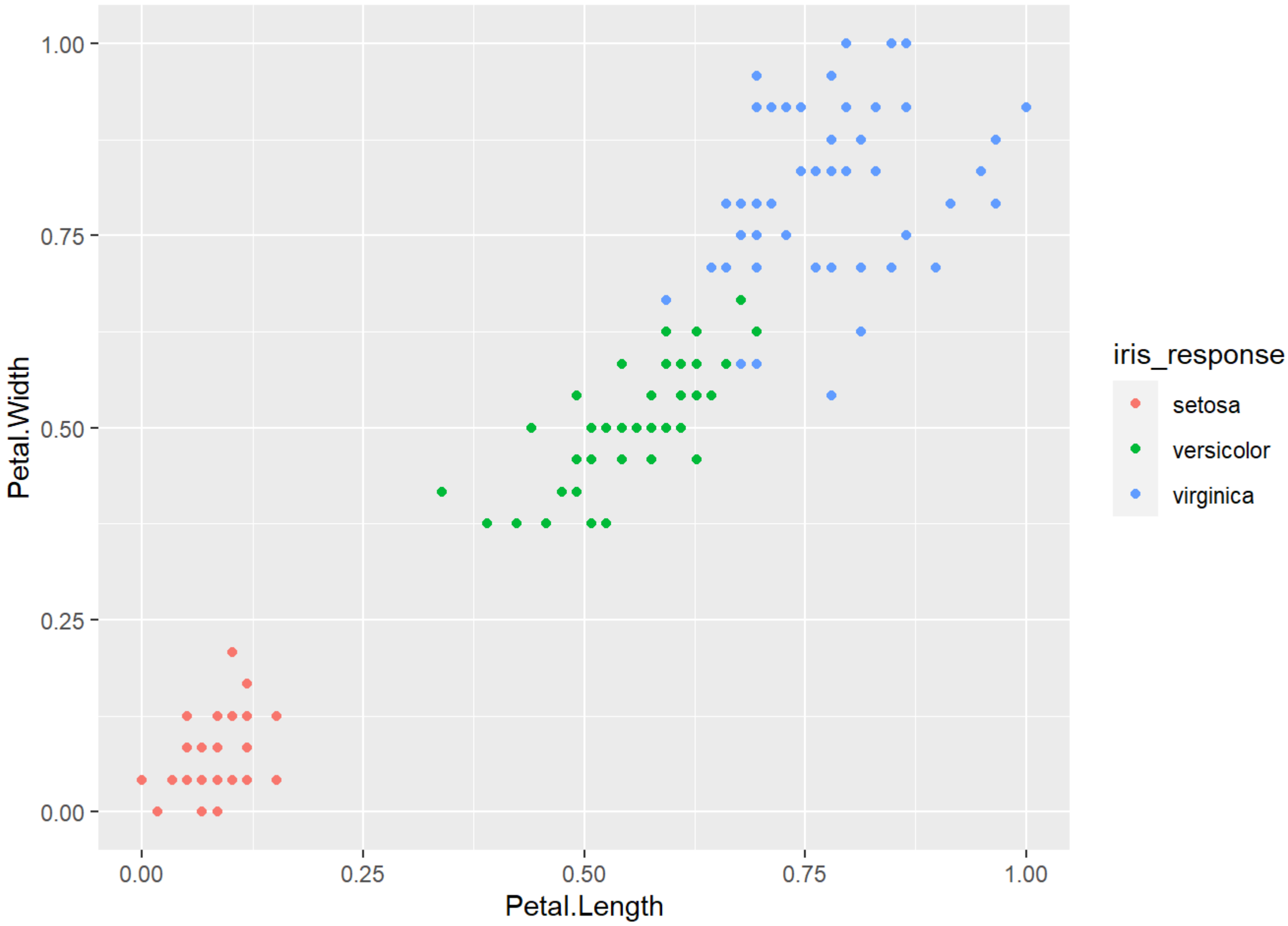

```
sepal_cluster<- kmeans(my_iris[,1:2], 3, nstart = 25)

 sepal_accuracy <- sepal_cluster$betweenss/sepal_cluster$totss*100
 sepal_accuracy
```

```
## [1] 71.60328
```

Here is the accuracy for using only sepal length and width and as said earlier its below overall data's accuracy. We can imporve this accuracy by dividing the into more cluters.

```
ggplot(iris_scaled, aes(Petal.Length, Petal.Width, color=iris_response)) +geom_point()
```




```
petal_cluster<- kmeans(my_iris[,3:4], 3, nstart = 25)

petal_accuracy <- petal_cluster$betweenss/petal_cluster$totss*100
petal_accuracy
```

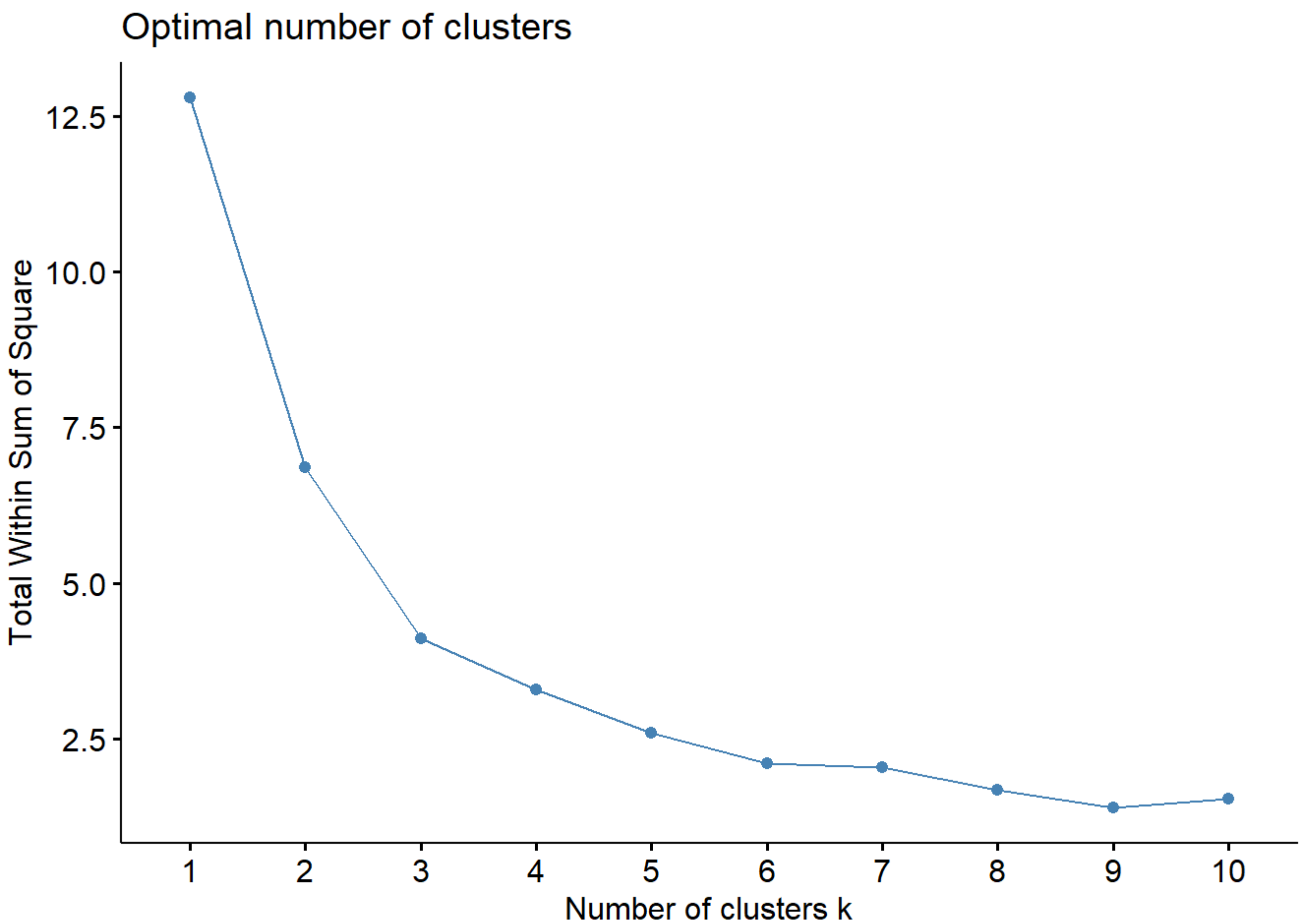
```
## [1] 94.30539
```

As, said earlier the data with only petal length and width give us more accurate result than our overall data. From abive analysis we can say that we can use just petal length and petal width to segragate our data more accuratly.

Just finding the good K vlaue for Sepal leght and sepal width data.

```
sepal_data <- iris_scaled[,1:2]
k_cluster = rep(0,15)
for(k in 1:15) {
  sepal_clusters <- kmeans(sepal_data, centers=k, nstart=5)
  k_cluster[k] <- sepal_cluster$tot.withinss
}

fviz_nbclust(sepal_data, kmeans, method = "wss")
```



For only sepal data K= 6 should give us good results.

```
newsepal_cluster<- kmeans(sepal_data, 6, nstart = 25)

 newsepal_accuracy <- newsepal_cluster$betweenss/newsepal_cluster$totss*100
 newsepal_accuracy
```

```
## [1] 83.6086
```

Question 5.1 Using crime data from the file uscrime.txt(http://www.statsci.org/data/general/uscrime.txt (http://www.statsci.org/data/general/uscrime.txt), description at http://www.statsci.org/data/general/uscrime.html (http://www.statsci.org/data/general/uscrime.html)), test to see whether there are any outliers in the last column (number of crimes per 100,000 people). Use the grubbs.testfunction in the outlierspackage in R.

let's load the datafirst and plot them to check any outlier visually.

```
us_crime_data <- read.table("/Users/chintan/Downloads/6501/crimedata.txt",stringsAsFactor
  s = FALSE, header = TRUE)
head(us_crime_data)
```

```
##      M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
## 6 20.9995   682
```

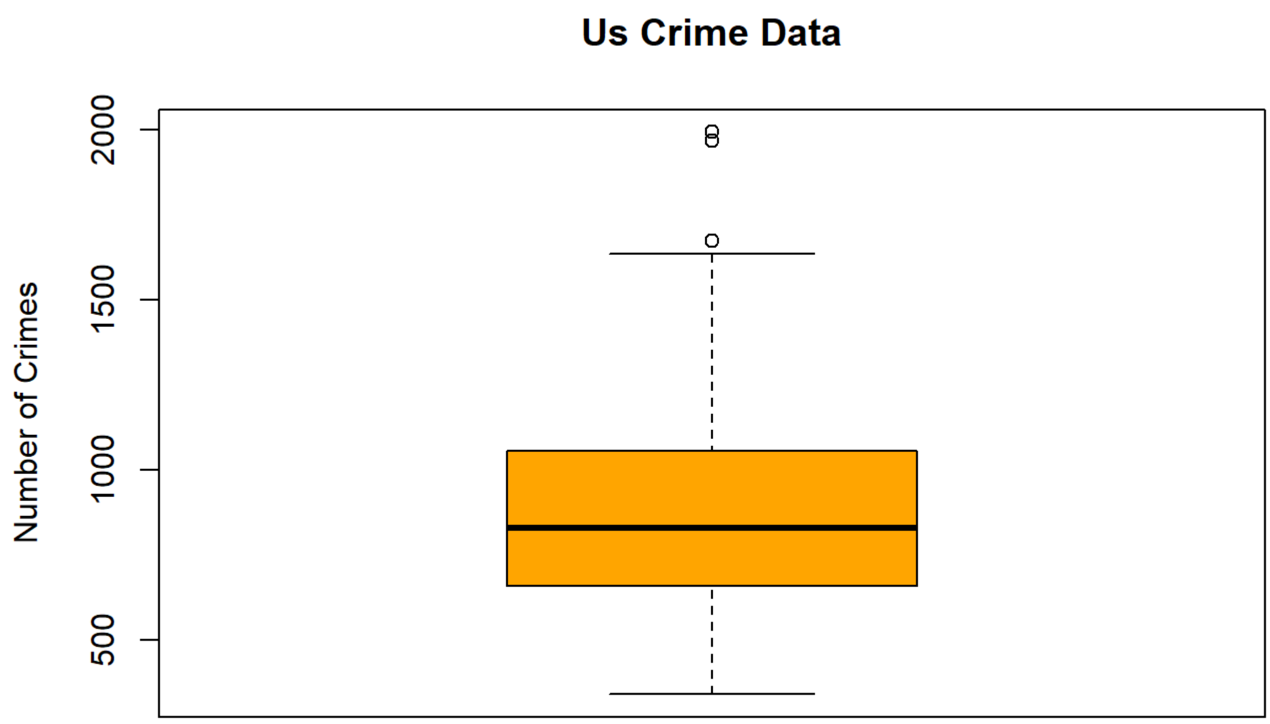
```r
# We only need the crime data.

crime_data <- us_crime_data[,16]
head(crime_data)
```

```
## [1]  791 1635  578 1969 1234  682
```

Let’s see this in box plot.

```r
boxplot(crime_data, main = "Us Crime Data", ylab = "Number of Crimes", col = "orange")
```



From the above boxplot we can see there are roughly three outliers on the max side of the data. We can use grubbs.test function to confrim this.

```
library(outliers)
```

```
outlier_test <- grubbs.test(crime_data, type = 11)
outlier_test
```

```
##
##  Grubbs test for two opposite outliers
##
## data:  crime_data
## G = 4.26877, U = 0.78103, p-value = 1
## alternative hypothesis: 342 and 1993 are outliers
```

Type =11 tests for two outliers on the opposite The p-value = 1 that means that atleast one of them is not an outlier.

```
outlier_test <- grubbs.test(crime_data, type = 10)
outlier_test
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data
## G = 2.81287, U = 0.82426, p-value = 0.07887
## alternative hypothesis: highest value 1993 is an outlier
```

Here, we can see that the value of p is very low that means higest value 1993 is an outlier. We can remove this point and check for next higest point if that is an outlier or not?

```
crime_data1 <- crime_data[-which.max(crime_data)]

outlier_test1 <- grubbs.test(crime_data1, type = 10)
outlier_test1
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data1
## G = 3.06343, U = 0.78682, p-value = 0.02848
## alternative hypothesis: highest value 1969 is an outlier
```

Based on this test we can see that value of p is very low and can also classify this as an outlier. Let's remove this poit from the data and check for 3rd higest point.

```
crime_data2<- crime_data1[-which.max(crime_data1)]

outlier_test2 <- grubbs.test(crime_data2, type = 10)
outlier_test2
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data2
## G = 2.56457, U = 0.84712, p-value = 0.1781
## alternative hypothesis: highest value 1674 is an outlier
```

```
crime_data3<- crime_data2[-which.max(crime_data2)]

outlier_test3 <- grubbs.test(crime_data3, type = 10)
outlier_test3
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data3
## G = 2.68561, U = 0.82837, p-value = 0.1139
## alternative hypothesis: highest value 1635 is an outlier
```

```
crime_data4<- crime_data3[-which.max(crime_data3)]

outlier_test4 <- grubbs.test(crime_data4, type = 10)
outlier_test4
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data4
## G = 2.69107, U = 0.82347, p-value = 0.1082
## alternative hypothesis: highest value 1555 is an outlier
```

```
crime_data5<- crime_data4[-which.max(crime_data4)]

outlier_test5 <- grubbs.test(crime_data5, type = 10)
outlier_test5
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data5
## G = 1.87133, U = 0.91251, p-value = 1
## alternative hypothesis: highest value 1272 is an outlier
```

```
crime_data6<- crime_data5[-which.max(crime_data5)]

outlier_test6 <- grubbs.test(crime_data6, type = 10)
outlier_test6
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data6
## G = 1.85223, U = 0.91209, p-value = 1
## alternative hypothesis: lowest value 342 is an outlier
```

One by one we have removed the max point from the data and check the P value. It looks like the p- value for 1993, 1969, 1674, 1635 & 1555 is below 1 and these five points posibly can be clssify as outlier. It is up to us to decide the threshold value of p.

Similarly we can aslo test for the lowest value of the data by setting opposite = True.

```
outlier_test_min <- grubbs.test(crime_data, type = 10, opposite = TRUE)
outlier_test_min
```

```
##
##  Grubbs test for one outlier
##
## data:  crime_data
## G = 1.45589, U = 0.95292, p-value = 1
## alternative hypothesis: lowest value 342 is an outlier
```

Here we get p=1 that means that the lower point is not an oulier.

Comparing the result of boxplot and grubbs test we can see that boxplot shows 3 outliers while grubbs test gives us 5 possible outliers.

Question 6.1 Describe a situation or problem from your job, everyday life, current events, etc., for which a Change Detection model would be appropriate. Applying the CUSUM technique, how would you choose the critical value and the threshold?

From my previous experience working with telecoms industries I can tell that this change dectiond modle is wildley used in this industry. From monitoring the frequency of signals to the number of users they have.

They constantly monitor the sigal stength recevied by the cellphones. The good signal sterngth is considered tobe some where around (- 80dbs). Based on the location of the device like inside the house, in the basement and distance from the tower will receive the lower firequency strength upto - 110 dbs. If the signal strength recorded below -80dbs, upto -110dbs will be considered the false change and it has no major implication but if the signal strength drops below -110dbs that’s when the users will start facing the drop calls issues. All this analysis is done through change detection model. Some compnies would like to keep the smaller c for sensitive detection. Even though there is a cost ivolve in examing these issues, some compnies want to give there uers best experience.

The other aspect is businees aspect. The telecome compnies compete aggressively to gain their peers’ market shares in their customer segments. They keep monitering the number of customers they have. Everyday there will some new custoemr and some will be leaving this will be the false change but any sudden drop they want to invesitage. For that purpose they like keep the smaller c for any sensitve change.

Question 6.2 1. Using July through October daily-high-temperature data for Atlanta for 1996 through 2015, use a CUSUM approach to identify when unofficial summer ends (i.e., when the weather starts cooling off) each year. You can get the data that you need from the file temps.txt or online, for example at http://www.iweathernet.com/atlanta-weather-records (http://www.iweathernet.com/atlanta-weather-records) or https://www.wunderground.com/history/airport/KFTY/2015/7/1/CustomHistory.html (https://www.wunderground.com/history/airport/KFTY/2015/7/1/CustomHistory.html) . You can use R if you'd like, but it's straightforward enough that an Excel spreadsheet can easily do the job too.

```
data <- read.table("/Users/chintan/Downloads/6501/temps.txt",stringsAsFactors = FALSE, he
  ader = TRUE)
head(data)
```

```
##     DAY X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006 X2007
## 1 1-Jul    98    86    91    84    89    84    90    73    82    91    93    95
## 2 2-Jul    97    90    88    82    91    87    90    81    81    89    93    85
## 3 3-Jul    97    93    91    87    93    87    87    87    86    86    93    82
## 4 4-Jul    90    91    91    88    95    84    89    86    88    86    91    86
## 5 5-Jul    89    84    91    90    96    86    93    80    90    89    90    88
## 6 6-Jul    93    84    89    91    96    87    93    84    90    82    81    87
##   X2008 X2009 X2010 X2011 X2012 X2013 X2014 X2015
## 1    85    95    87    92   105    82    90    85
## 2    87    90    84    94    93    85    93    87
## 3    91    89    83    95    99    76    87    79
## 4    90    91    85    92    98    77    84    85
## 5    88    80    88    90   100    83    86    84
## 6    82    87    89    90    98    83    87    84
```

For change detection we need to find the center value (base line). As dicussed on the monday call, we can take the averae tmp value of july 1996 as our mu or avg of July & Aug of 1996. Its upto us but we need to keep the mu value constant and compare it with each year to figure out when the summer end each year. For this exercise I am going to take avg of july and aug month of 1996 for center value.

```
library(qcc)
```

```
## Warning: package 'qcc' was built under R version 3.6.3
```

```
## Package 'qcc' version 2.7
```

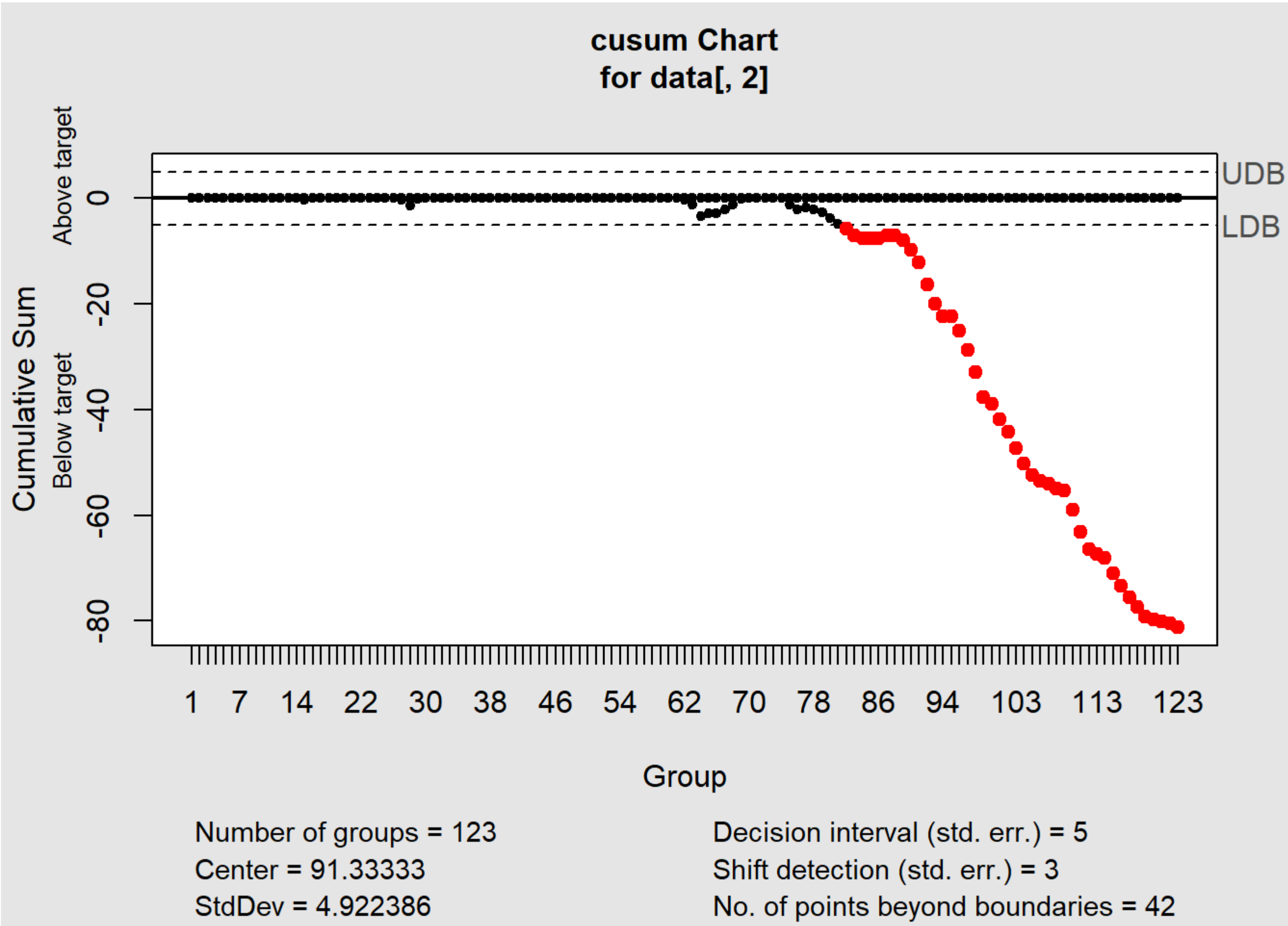
```
## Type 'citation("qcc")' for citing this R package in publications.
```

```
CV <- mean(data[1:30,2])
stdev <- sd(data[1:30,2])
CV
```

```
## [1] 91.33333
```

Now we have our base line temp of 91. We need keep this value constant and comapre it against all years to determine the end of summer.We will feed this value to cusum fucntion.

```
yr_1996 <-cusum(data[,2],center = CV, std.dev =stdev,se.shift =3, plot =TRUE)
```



The above graph indicates that at the first red point cusum function detects the first change(around row#82) and after that there is significant drop in the temperature.We can cosider the first red point as the summer end.

Below code captuers all the 42 records. We only care about the first red point, since that gives us the end of summerday.

```
yr_1996_v <- yr_1996$violations$lower

min(yr_1996_v)
```

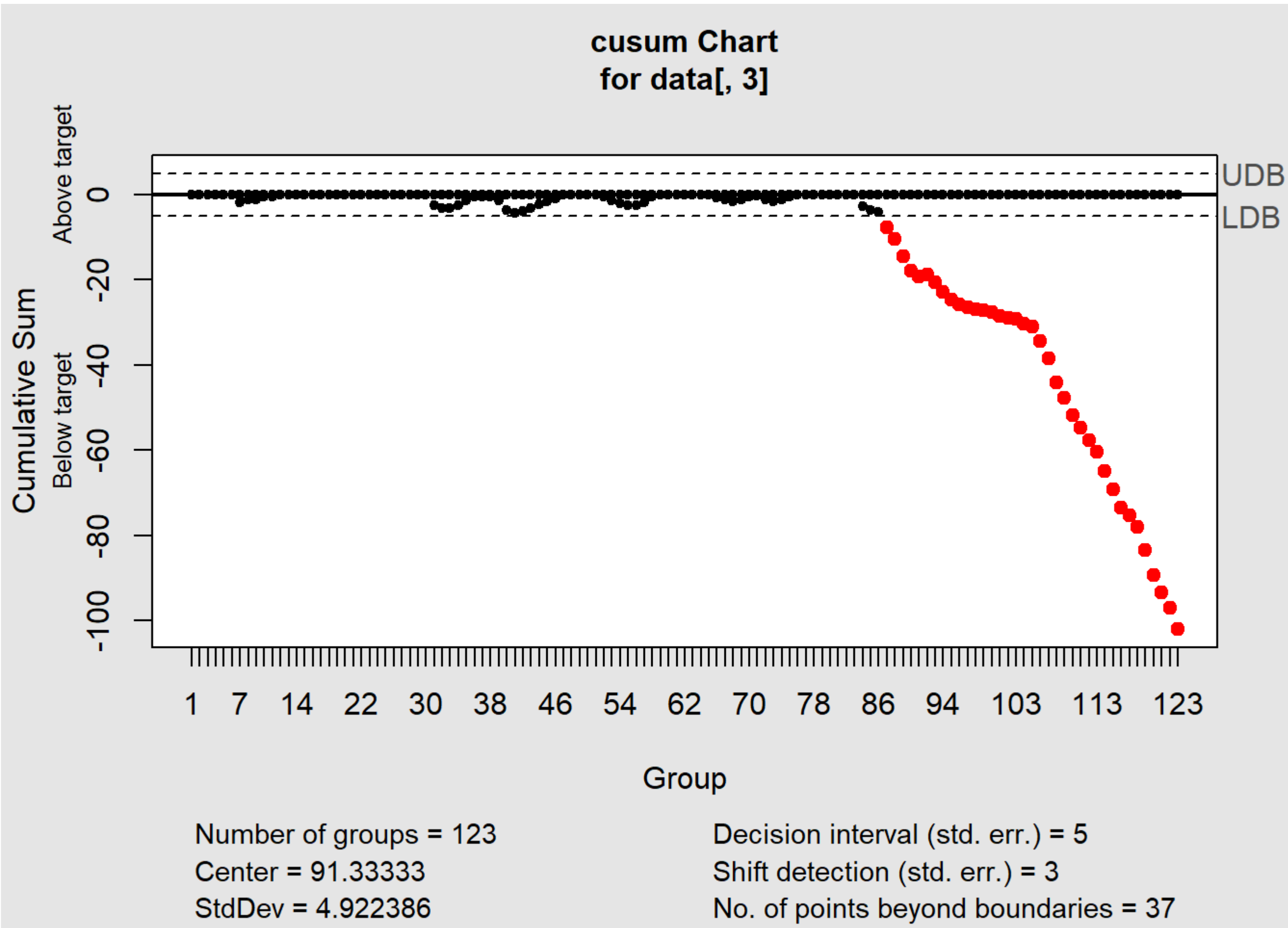
```
## [1] 82
```

```
data[82,1]
```

```
## [1] "20-Sep"
```

For year 1996 we can say that on summer ends on “Sep -20”. We can run the above code for each year and capture and find out the summer end date for all years.

```
yr_1997 <-cusum(data[,3],center = CV, std.dev =stdev,se.shift =3, plot =TRUE)
```

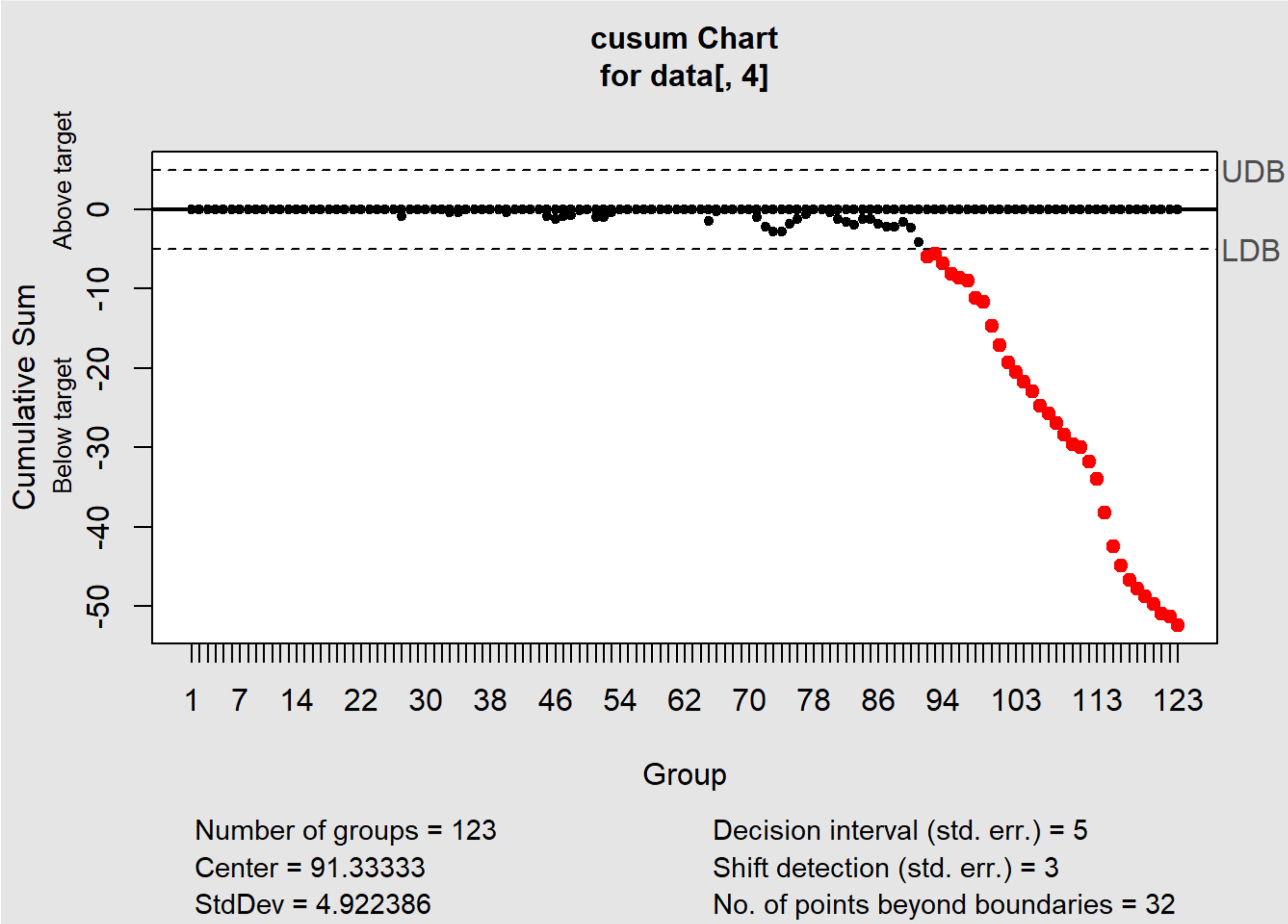



```
yr_1997_v <- yr_1997$violations$lower
```

```
date <- data[min(yr_1997_v),1]
date
```

```
## [1] "25-Sep"
```

```
yr_1998 <-cusum(data[,4],center = CV, std.dev =stdev,se.shift =3, plot =TRUE)
```

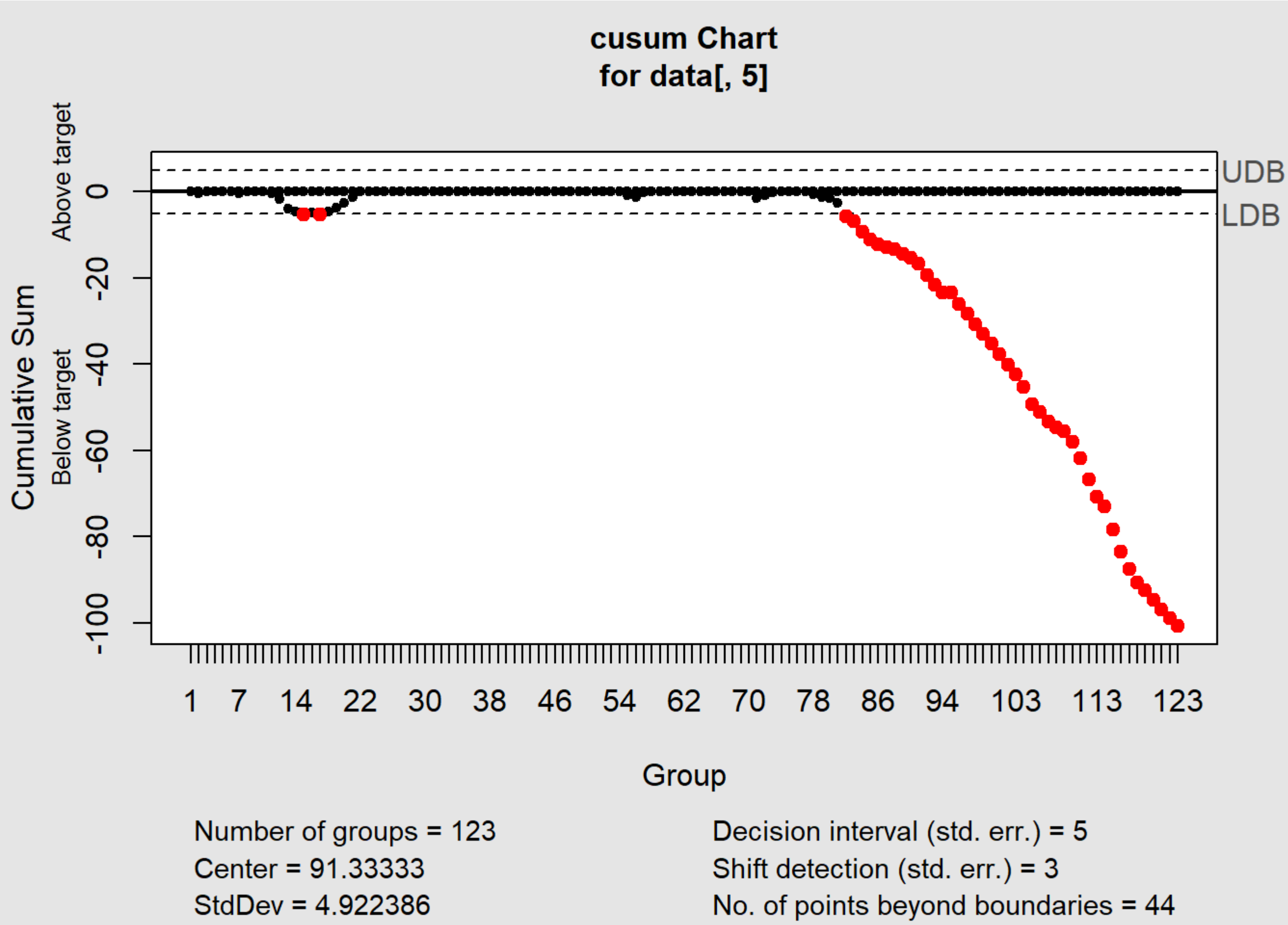



```
yr_1998_v <- yr_1998$violations$lower
```

```
date <- data[min(yr_1998_v),1]
date
```

```
## [1] "30-Sep"
```

```
yr_1999 <-cusum(data[,5],center = CV, std.dev =stdev,se.shift =3, plot =TRUE)
```



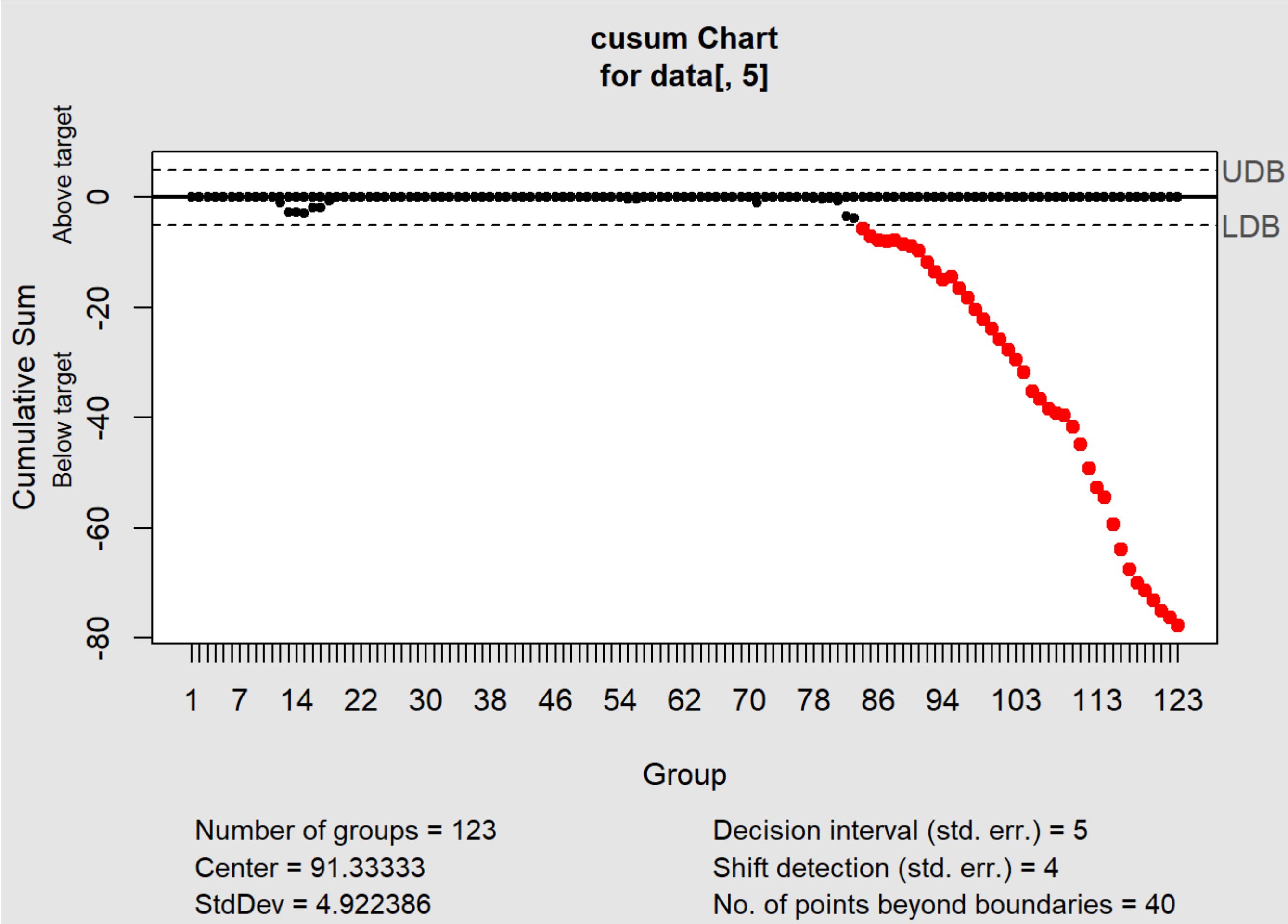

```
yr_1999_v <- yr_1999$violations$lower
```

```
dates <- data[(yr_1999_v),1]
dates
```

```
##  [1] "15-Jul" "17-Jul" "20-Sep" "21-Sep" "22-Sep" "23-Sep" "24-Sep" "25-Sep"
##  [9] "26-Sep" "27-Sep" "28-Sep" "29-Sep" "30-Sep" "1-Oct"  "2-Oct"  "3-Oct"
## [17] "4-Oct"  "5-Oct"  "6-Oct"  "7-Oct"  "8-Oct"  "9-Oct"  "10-Oct" "11-Oct"
## [25] "12-Oct" "13-Oct" "14-Oct" "15-Oct" "16-Oct" "17-Oct" "18-Oct" "19-Oct"
## [33] "20-Oct" "21-Oct" "22-Oct" "23-Oct" "24-Oct" "25-Oct" "26-Oct" "27-Oct"
## [41] "28-Oct" "29-Oct" "30-Oct" "31-Oct"
```

For year 1999 we got interesting result. On 15th july and 17th July temperature drops significant below the thresold that our model markes them in red. Its falsely detecting the change. We can increase our thresold vlue to avoid this kind of flase detection. I will run the cusum for the same year with SS = 4.

```
yr_1999 <-cusum(data[,5],center = CV, std.dev =stdev,se.shift =4, plot =TRUE)
```



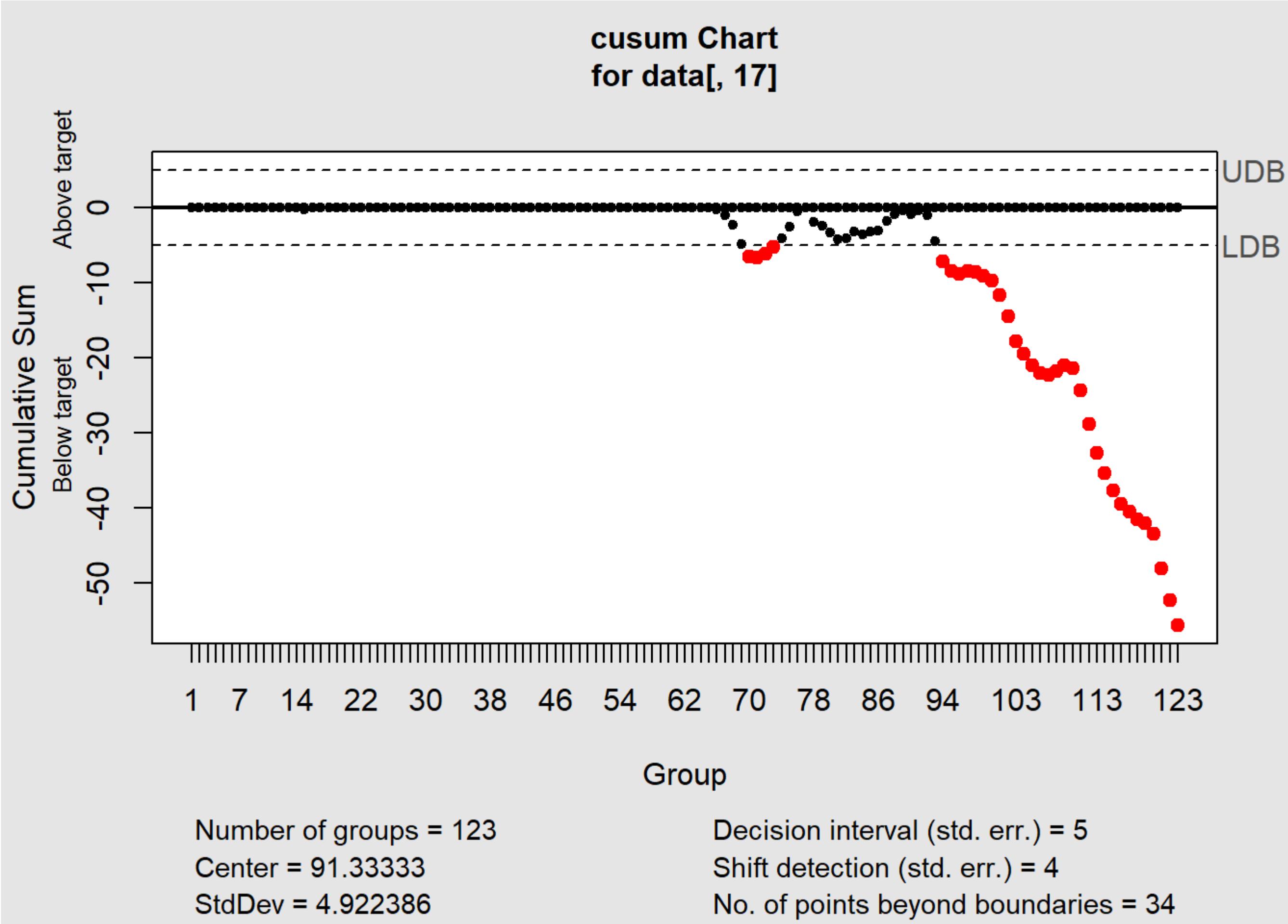

```
yr_1999_v <- yr_1999$violations$lowe
```

As we can see from the above graph that by increasing the shift value to 4 it avoids false change detection and gives us the Sep -22 as the summer end date.

```
dates <- data[min(yr_1999_v),1]
dates
```

```
## [1] "22-Sep"
```

```
yr_2011 <-cusum(data[,17],center = CV, std.dev =stdev,se.shift =4, plot =TRUE)
```




```
yr_2011_v <- yr_2011$violations$lower
```

```
dates <- data[(yr_2011_v),1]
dates
```

```
##  [1] "8-Sep"  "9-Sep"  "10-Sep" "11-Sep" "2-Oct"  "3-Oct"  "4-Oct"  "5-Oct"
##  [9] "6-Oct"  "7-Oct"  "8-Oct"  "9-Oct"  "10-Oct" "11-Oct" "12-Oct" "13-Oct"
## [17] "14-Oct" "15-Oct" "16-Oct" "17-Oct" "18-Oct" "19-Oct" "20-Oct" "21-Oct"
## [25] "22-Oct" "23-Oct" "24-Oct" "25-Oct" "26-Oct" "27-Oct" "28-Oct" "29-Oct"
## [33] "30-Oct" "31-Oct"
```

Here is also another year with very flucating temperateures. We can ignore the early drop in the temp and consider that as false chage detection. For year 2011 we can consider the “Oct 2nd” as the summer end date.After that data the temperature consistently drops.

I have run the cusum funcation for all the years and below are the dates of summer end for each year.

Year Date 1996 20-Sep 1997 25-Sep 1998 30-Sep 1999 22-Sep 2000 7-Sep 2001 25-Sep 2002 25-Sep 2003 29-Sep 2004 18-Sep 2005 8-Oct 2006 27-Sep 2007 12-Oct 2008 21-Sep 2009 17-Sep 2010 2-Oct 2011 2-Oct 2012 2-Oct 2013 25-Sep 2014 28-Sep 2015 25-Sep

It’s look like for most of the year summer end date is around the last week of september or early October.

Question 6.2.2 Use a CUSUM approach to make a judgment of whether Atlanta's summer climate has gotten warmer in that time (and if so, when).

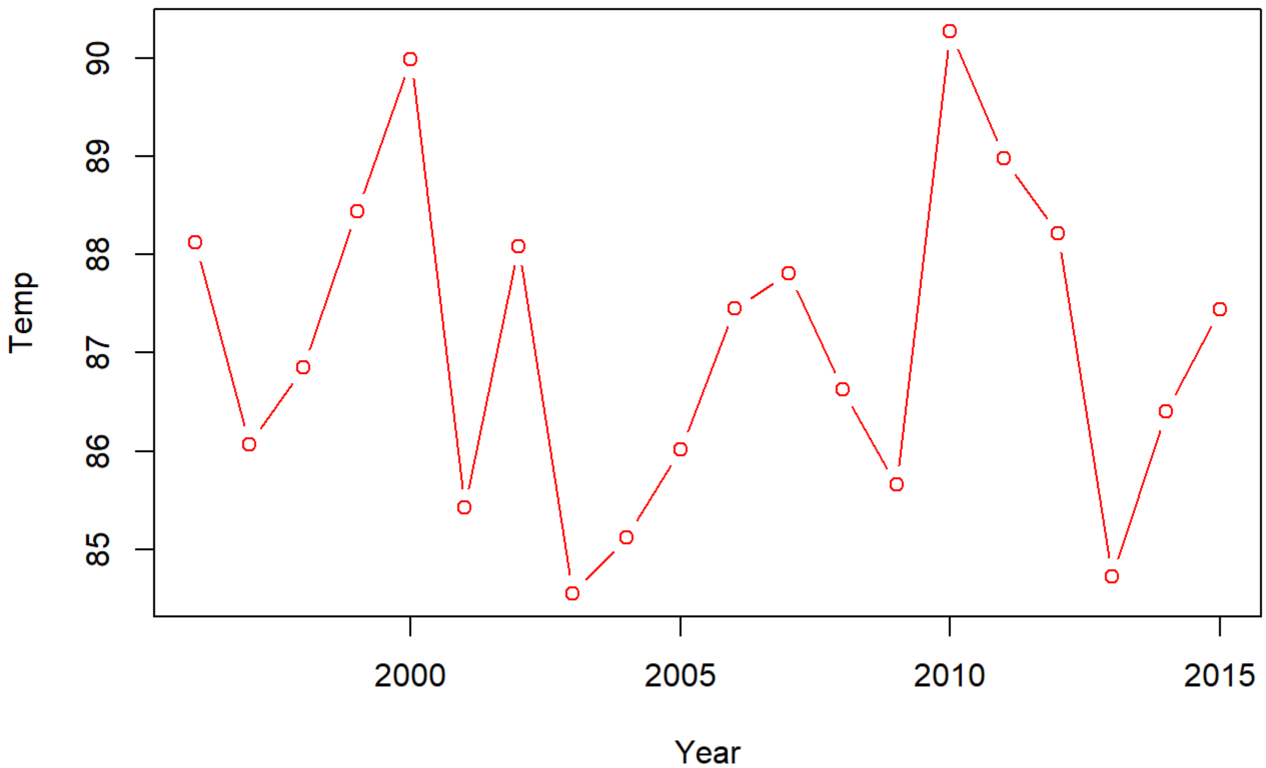
In question 1 we figure out the end of summer and now if we take the averege of all the summer days for each year we can observe the diffrence in the temp.

```
avg_1996 <- mean(data[1:which(data$DAY == "20-Sep"),2])
avg_1997 <- mean(data[1:which(data$DAY == "25-Sep"),3])
avg_1998 <- mean(data[1:which(data$DAY == "30-Sep"),4])
avg_1999 <- mean(data[1:which(data$DAY == "22-Sep"),5])
avg_2000 <- mean(data[1:which(data$DAY == "7-Sep"),6])
avg_2001 <- mean(data[1:which(data$DAY == "25-Sep"),7])
avg_2002 <- mean(data[1:which(data$DAY == "25-Sep"),8])
avg_2003 <- mean(data[1:which(data$DAY == "29-Sep"),9])
avg_2004 <- mean(data[1:which(data$DAY == "18-Sep"),10])
avg_2005 <- mean(data[1:which(data$DAY == "8-Oct"),11])
avg_2006 <- mean(data[1:which(data$DAY == "27-Sep"),12])
avg_2007 <- mean(data[1:which(data$DAY == "12-Oct"),13])
avg_2008 <- mean(data[1:which(data$DAY == "21-Sep"),14])
avg_2009 <- mean(data[1:which(data$DAY == "17-Sep"),15])
avg_2010 <- mean(data[1:which(data$DAY == "2-Oct"),16])
avg_2011 <- mean(data[1:which(data$DAY == "2-Oct"),17])
avg_2012 <- mean(data[1:which(data$DAY == "2-Oct"),18])
avg_2013 <- mean(data[1:which(data$DAY == "25-Sep"),19])
avg_2014 <- mean(data[1:which(data$DAY == "28-Sep"),20])
avg_2015 <- mean(data[1:which(data$DAY == "25-Sep"),21])
```

```
x<-c(1996:2015)
x
```

```
##  [1] 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
## [16] 2011 2012 2013 2014 2015
```

```
y <- c(avg_1996,avg_1997,avg_1998,avg_1999,avg_2000,avg_2001,avg_2002,avg_2003,avg_2004,a
  vg_2005,
       avg_2006,avg_2007,
       avg_2008,avg_2009,avg_2010,avg_2011,avg_2012,avg_2013,avg_2014,avg_2015)
plot(x,y, type = "b",col ="red",xlab= "Year", ylab = "Temp")
```



It is hard to tell wehter the summers are gettign warmer in Atlanta based on the above diagram. For year 2000 and 2010 the Avg temp of summer days is very high(close to 90) while for 2003 and 2013 the avg temp is close to 84.