ISYE 6501 - Week 2 Homework

Ujjawal Madan

23/05/2020

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.

A situation where a clustering model might be appropriate is trying to predict the neighbourhood of a resident based on a certain factors. Those factors could include: Price of home, Square footage of home, Age of Resident(s), Relationship and Familial Status, Household Family Income, Commute time to downtown, Local School Ratings, etc. By conducting a representative sample of the cities' residents and aggregating the data, I am inclined to think that clusters would be formed, with each cluster representing a distinct neighbourhood in the city.

As an example, a cluster of a particularly affluent neighbourhood could be formed based on the facts that:

- the median family household income is between $300,000 and $500,000
- the median home price is well above $1,000,000
- Median Square footage of home is 3500-4000 feet
- commute time to downtown (by car) might be 30 -40 minutes and would be almost the same for all residents in this neighbourhood
- The local schools have very good reviews and would be the same for all residents in the area
- 95% of households are married couples with kids. Therefore majority of households have more than 3 people.

It is likely in this affluent neighbourhood that the standard deviation for each of these feature variables is relatively low. For example, the prices of most of the houses will be close to the median since in this affluent neighbourhood, there are very few houses that ever really sell below $1,000,000 or go above $2,000,000. As such, it is likely with all of these variables, the data points would form clusters which would go on to represent the neighbourhoods.

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). 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.

Normalizing

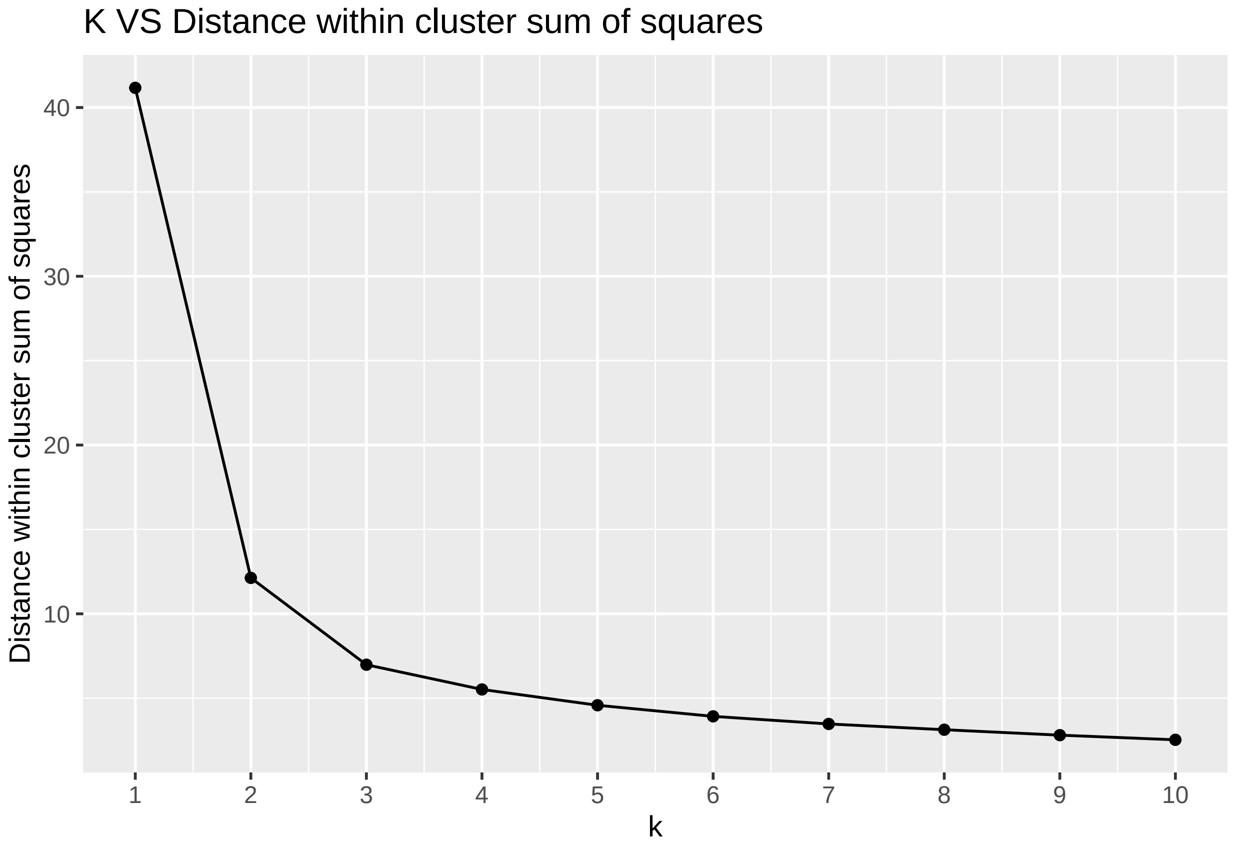
Although it is not specified, we should normalize the data. As we may be comparing different combinations of predictor variables, it is important for the data to be normalized so we can properly compare the results.

Number of clusters

Intuitively it would make sense that k =3 is the most appropriate. Given that in this example, we already know that the data falls into one of three classes, it would make sense that the data would also form three clusters and such a model would be of most use to us if we are hoping to predict the class of a data point.

However, just as in a real world scenario where we might not know the number of clusters we are looking for, let's rely on the elbow method and try plotting the total distances based on number of clusters.

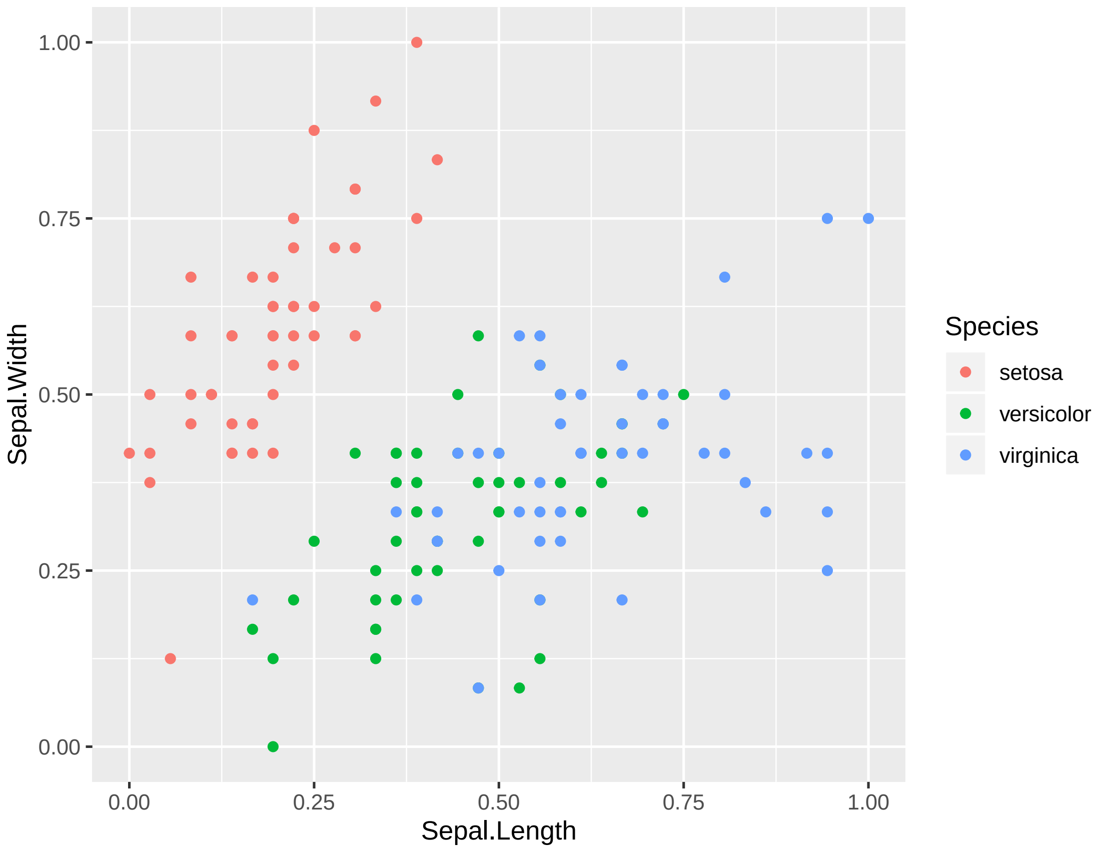
```
##  [1] 41.166110 12.127791  6.982216  5.516933  4.580323  3.923095  3.471598
##  [8]  3.130580  2.808462  2.532787
```

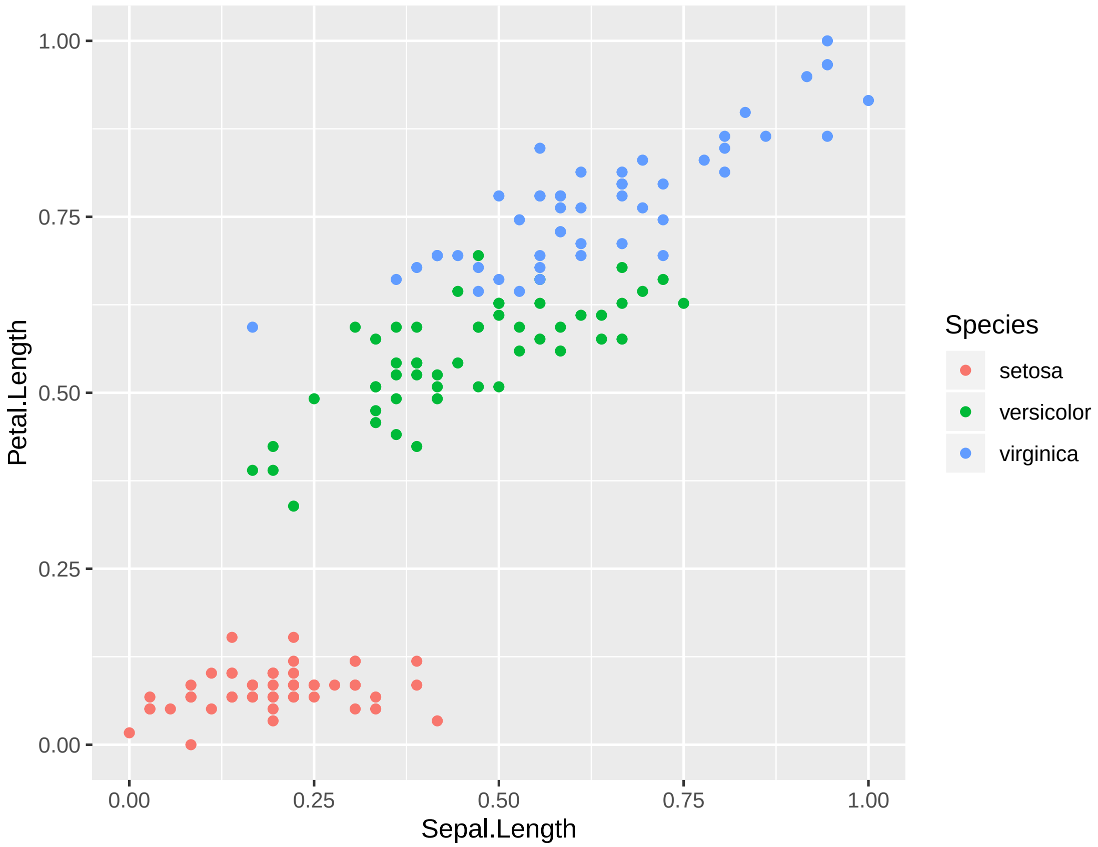


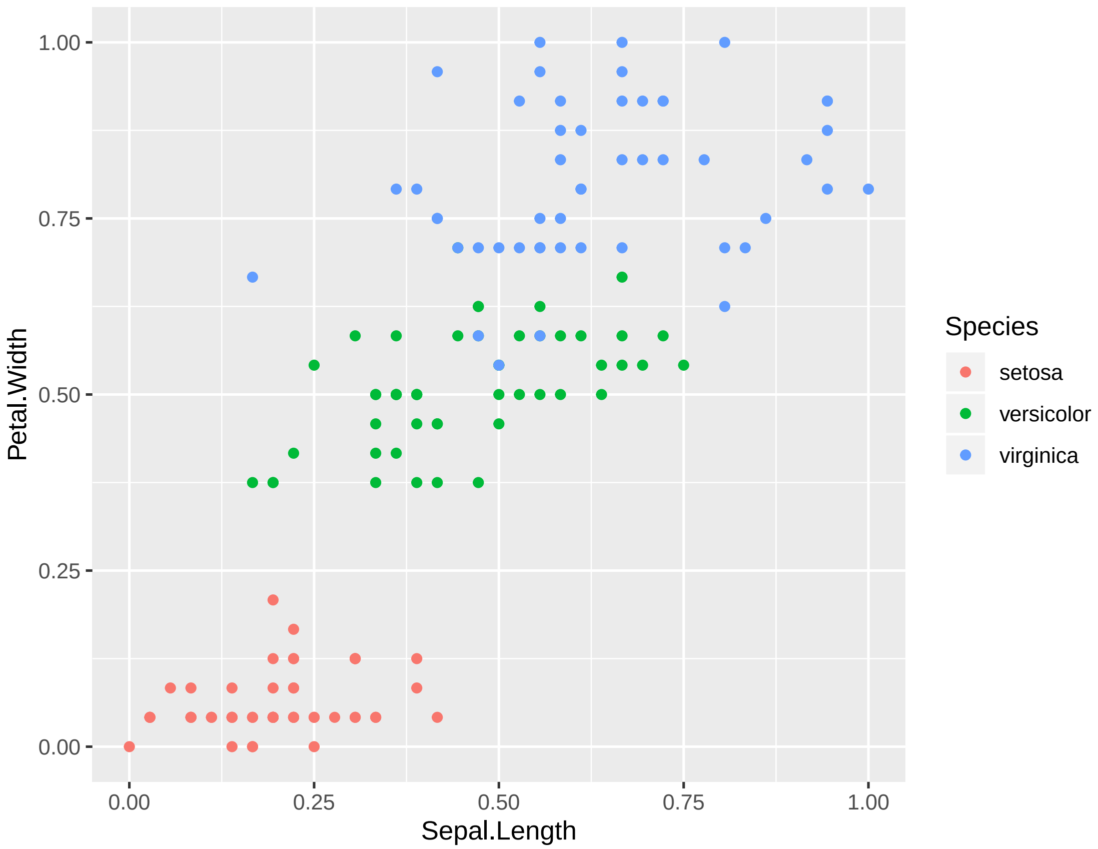
As we can see, when k is set to 3, it minimzes the distance most effectively which makes sense intuitively. Now that is decided, let's take a look at feature selection.

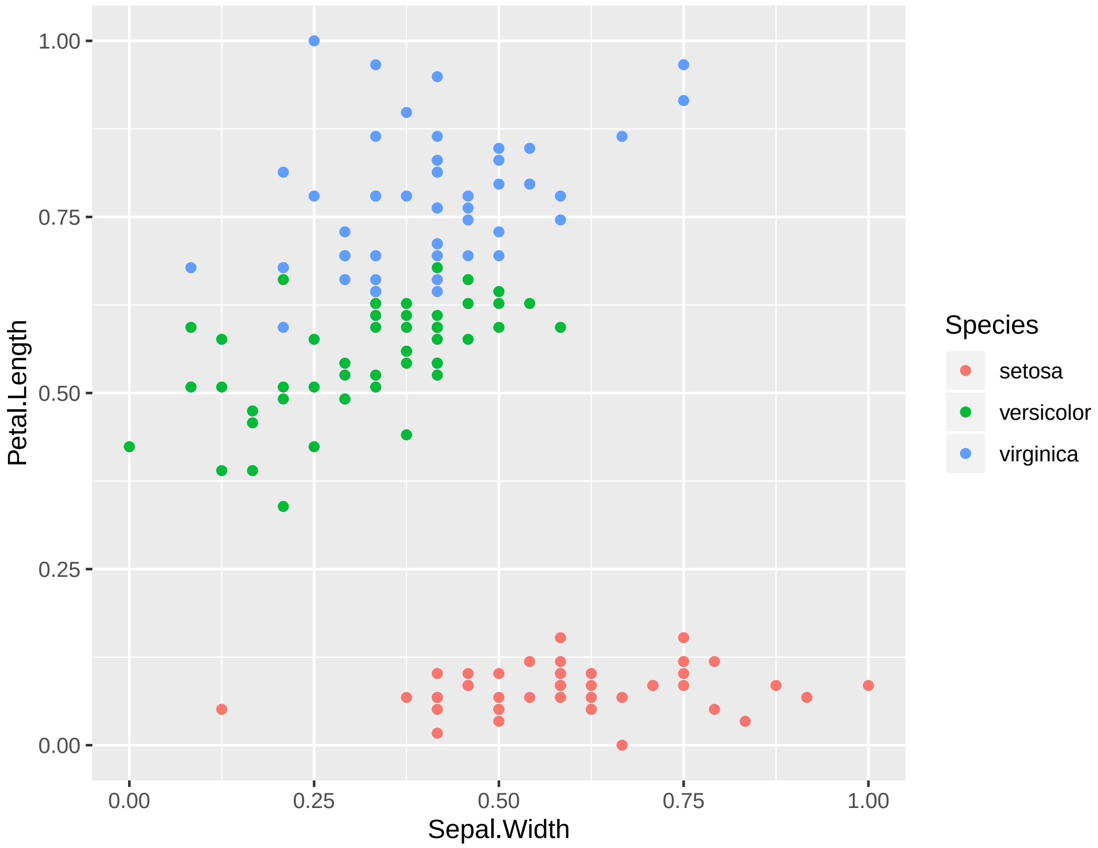
Feature Selection

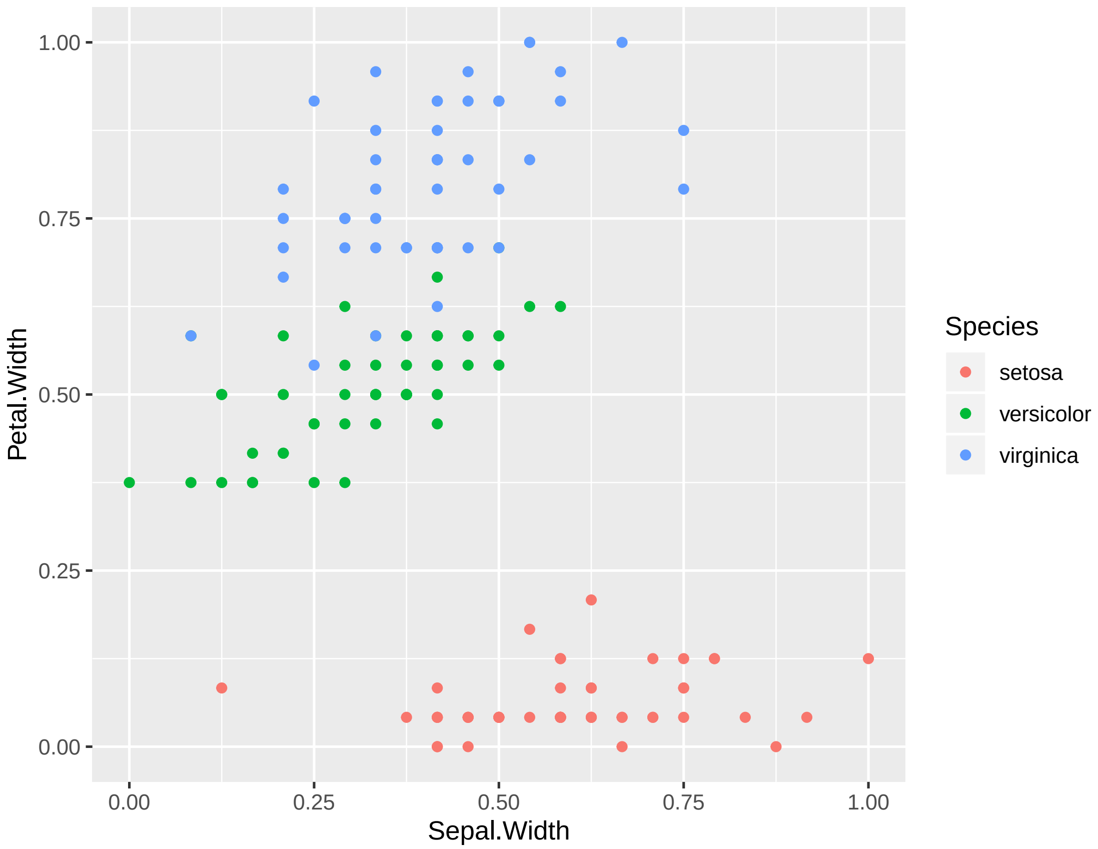
A simplistic way of exploring which features best help cluster our data points would be to plot different combinations of the four variables. In a real-world scenario, we would not have the species information, but as we do in this case let's take a look.

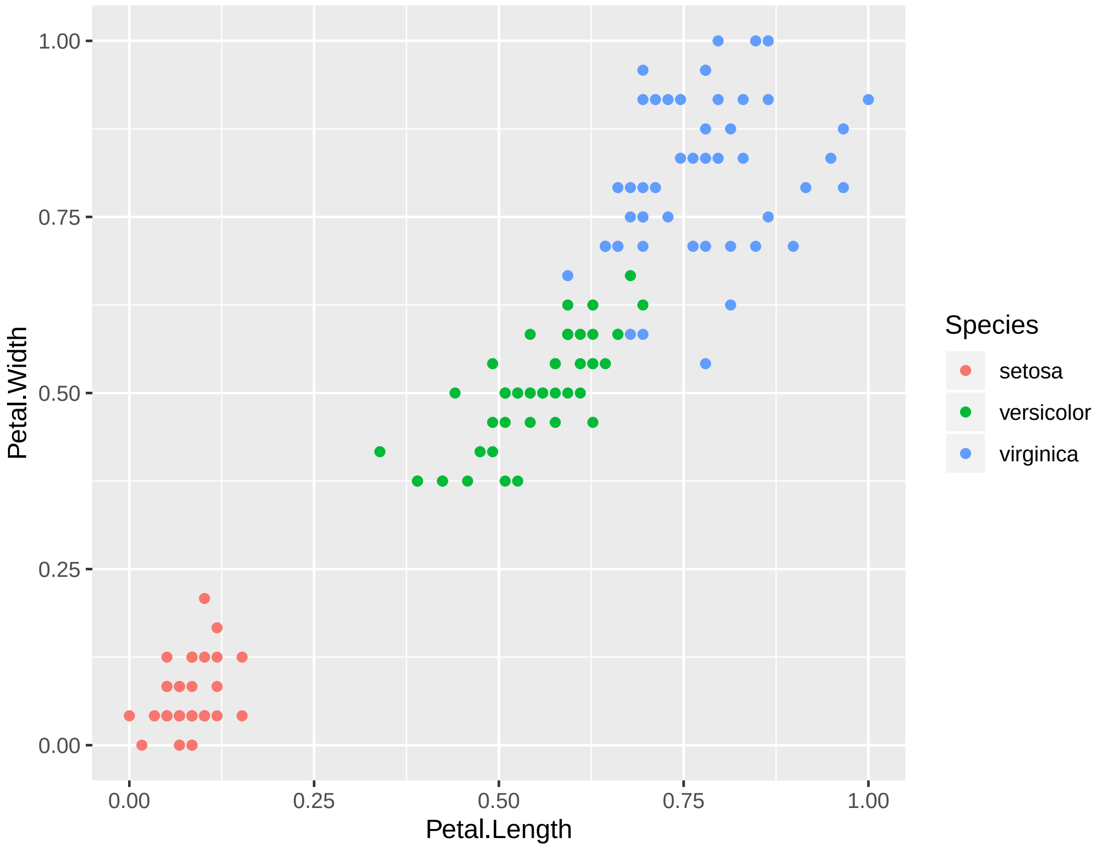

1.00
0.75
0.50
0.25
0.00
Sepal.Width
0.00
0.25
0.50
0.75
1.00
Sepal.Length
Species
setosa
versicolor
virginica


1.00
0.75
0.50
0.25
0.00
Petal.Length
0.00
0.25
0.50
0.75
1.00
Sepal.Length
Species
setosa
versicolor
virginica


1.00
0.75
0.50
0.25
0.00
Petal.Width
0.00
0.25
0.50
0.75
1.00
Sepal.Length
Species
● setosa
● versicolor
● virginica


1.00
0.75
0.50
0.25
0.00
Petal.Length
0.00
0.25
0.50
0.75
1.00
Sepal.Width
Species
● setosa
● versicolor
● virginica


1.00
0.75
0.50
0.25
0.00
Petal.Width
0.00
0.25
0.50
0.75
1.00
Sepal.Width
Species
● setosa
● versicolor
● virginica


1.00
0.75
0.50
0.25
0.00
Petal.Width
0.00
0.25
0.50
0.75
1.00
Petal.Length
Species
● setosa
● versicolor
● virginica

Intuitively, it seems that out of all the plots, the last one with Petal.Length and Petal.Width is the best. Of course, we are only looking at the pairs of feature variables right now since we cannot visualize in three or four dimensions.

After some experimentation, it seems that the best variables to use are Petal Length and Petal Width. The plots above would also suggest that those two variables would be best.

Of course, this visualization process was much too simplistic for more sophisticated feature selection. For more sophisticated feature selection we could employ dunn's index which is the ratio between: the smallest distance between observations not in the same cluster to the largest intra-cluster distance. In a real-world scenario, we could employ a forward step-wise regression where we would perform the clustering for each feature individually for some k and add the features to our selection set based on how well our set performs.

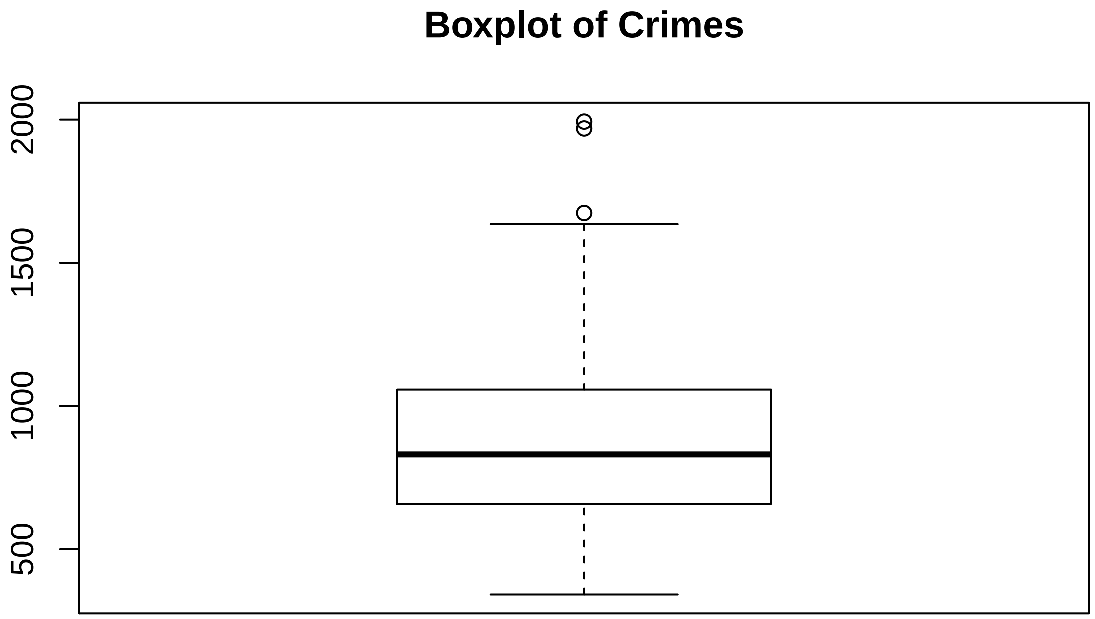
Results

setosa	versicolor	virginica
50	0	0
0	2	46
0	48	4

It seems that we achieved an accuracy of 144/150 or 96 percent, only misclassifying just 6 data points in total.

Question 5.1

Using crime data from the file uscrime.txt (http://www.statsci.org/data/general/uscrime.txt, description at 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.test function in the outliers package in R.



The grubbs test is used to detect a single outlier for data that follows an approximately normal distribution. The Tietjen-Moore Test or Generalized Extreme Studentized Deviate Test may be better if there may be more than one outlier present.

Above is a boxplot of our data. This gives us somewhat of an idea of how the data is distributed and it seems that the data is indeed a normal distribution which is right skewed. Therefore, we can run the grubbs test.

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

According to a grubb's test, the null hypothesis says that there are no outliers in the data. The p value for observing a Grubbs test statistic of 2.81287 or greater is 0.07887 and given that the threshold (significance level) for rejecting the null hypothesis is 0.05 (standard alpha level), we are unable to reject the null hypothesis. Therefore, according to the accepted threshold of 0.05 in our Grubbs' test, there are no outliers in the data.

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?

Currently, there is belief among many that less and less people of this generation (Gen Z and millenials) in North America are identifying as religious. See the economist article below. Change detection is key to help substantiating this argument. For example, if a person shows certain sample polls taken over the course of 50 years that support the idea of a decline in religion, we as data professionals need to make sure this person is attributing the results to a societal trend rather than random variability in the samples. Change detection, specifically CUSUM can be used to help define how much the results of the polls need to change for us to be confident in our assessment that there is a societal shift taking place.

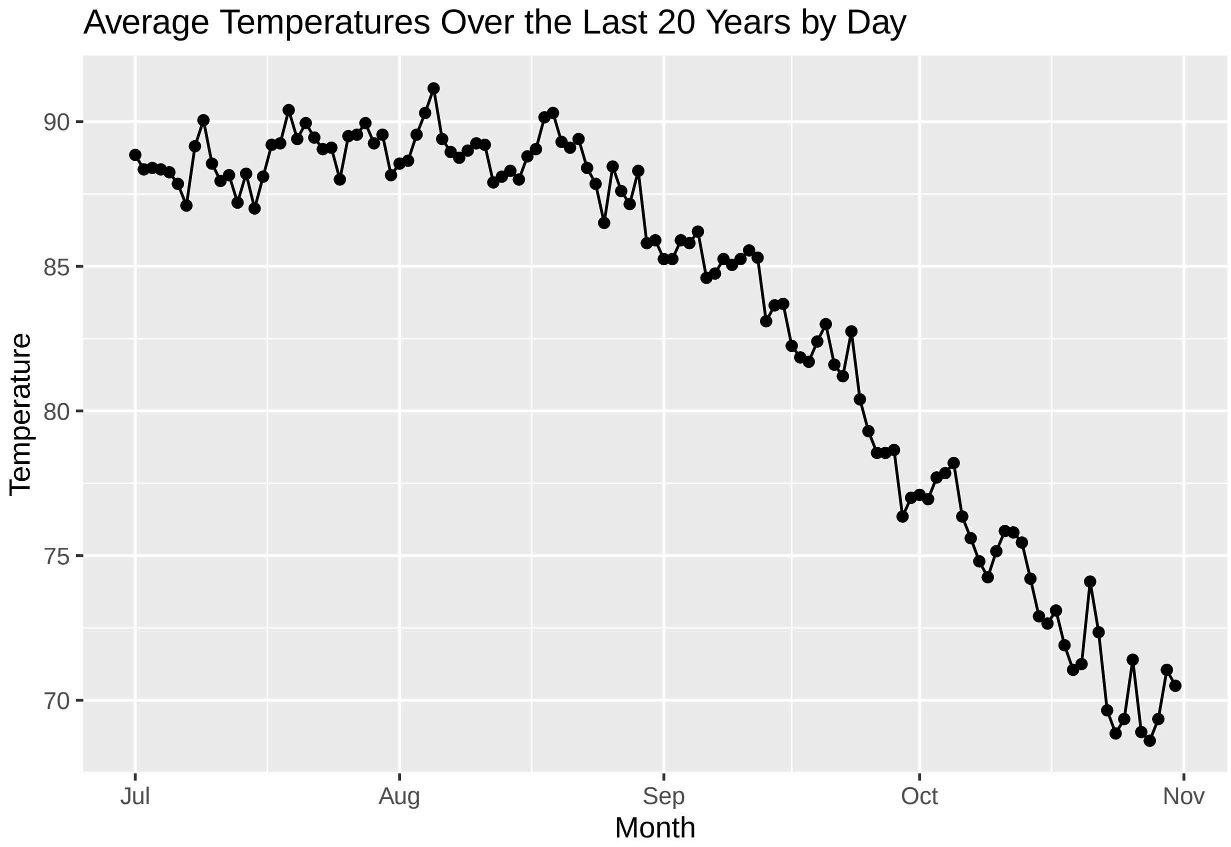
For example, using the CUSUM model, if we were to aggregate results from reliable polls and surveys (with a certain minimum sample size) on this topic taken over the past 30+ years and plot the results in a time-series distribution, we could monitor the changes and conclude whether more and more young people are opting to be non-religious. The critical value and threshold would be dependent on how certain one wants to be in their conclusion that a change is taking place. As this is not a scenario that is related to a life or death situation but is simply a socialogical assessment, I would choose a critical value and threshold that is balanced from which I could reach a reasonably confident verdict.

https://www.economist.com/united-states/2019/04/27/american-religion-is-starting-to-look-less-exceptional

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 or 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.

Let's start by importing the data and visualizing it:



So, if we were to assume that summer continues and there will be no decrease in temperatures, we can assume that August onwards the temperatures will remain in the approximate 87 to 92 degrees range.

```
## # A tibble: 1 x 1
##   average
##     <dbl>
## 1    88.8
```

Between July 1st and August 15th (the time we know it is summer for sure) the average temperature is 88.8. Our mu is therefore 88.8.

We now have to set a cost and threshold. As there are no safety issues dependent on us finding the change in time, nor is there a cost if we are a bit zealous in detecting a change a bit ahead of time, let's take a balanced approach.

There are two methods that I will use to implement the CUSUM. The first is manually coding the function that performs the CUSUM. The second is using a package called 'qcc' that performs it for us whilst also providing us with some additional information. Both approaches have their value so I will implement both.

Method 1 - Manual Function

```
#Extract temperature and the days as individual vectors
temp <- averages_day %>% pull(Temperature)
Day <- averages_day %>% pull(Day)
```

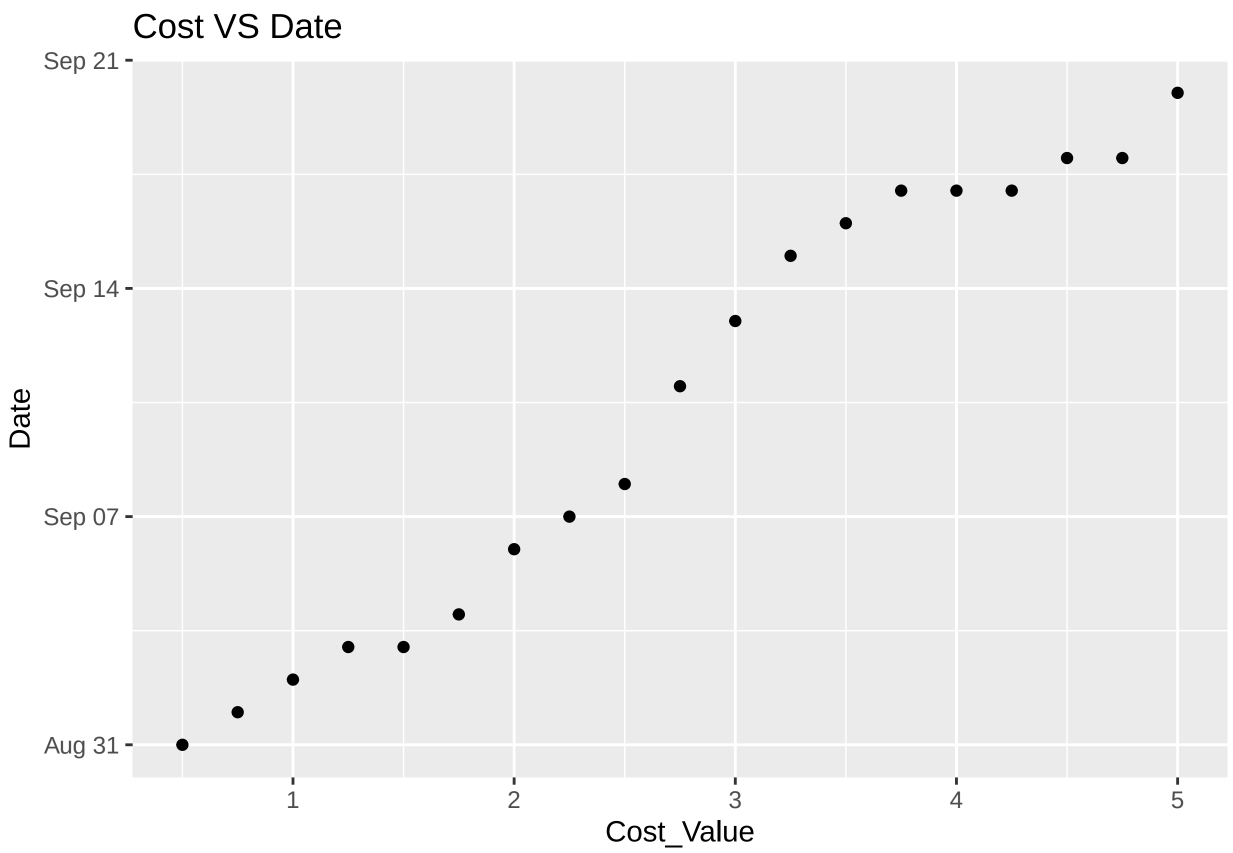
```r
#Function that performs the CUSUM
cusum <- function(average, cost, threshold){

  count <- 0
  for (i in 1:length(temp)){
    count <- min(temp[i] - average + cost + count, 0)
    if (abs(count) > abs(threshold)){
      Day <- Day[i]
      break
    }
  }

  if (i != length(temp)){
    return(c(cost, threshold, Day))
  } else{
    return (’No change detected’)
  }
}
#Example with mu set to 88.8, cost set to 1.5 degrees and threshold set to 8.
example <- cusum(88.8, 1.5, 8)[3]
print(as.Date(example, origin = "1970-01-01"))
```

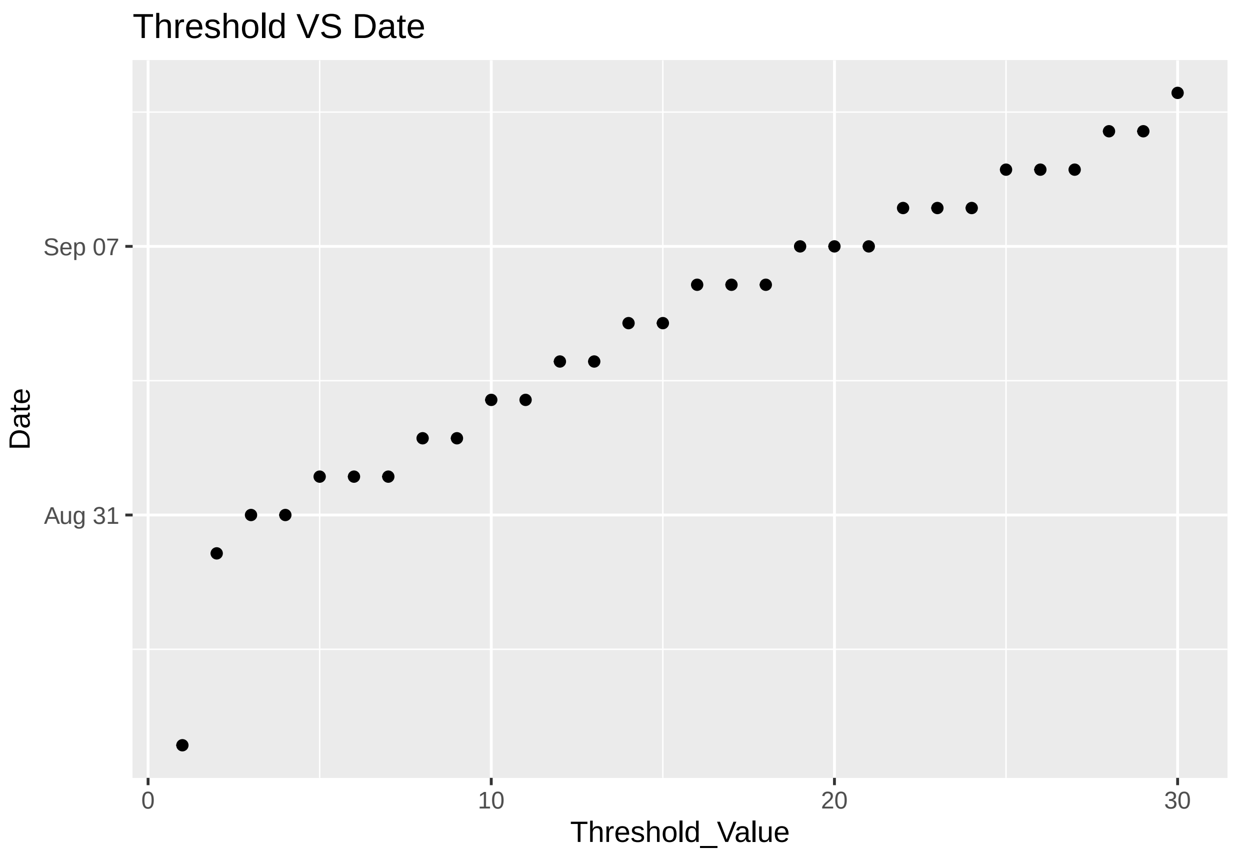
```
## [1] "2020-09-03"
```

The function above has returned us a date that tells us when a decline in temperatures has been detected and summer is officially transitioning into to autumn. This is tracking changes by the day as opposed to tracking changed by the week explored in method 2. The advantage of tracking by the week is that it ensures that the model is not detecting changes from random variation in temperatures. The mean temperature of the week is less likely to deviate from the expected value unless there is indeed change underfoot.

The mu was set to 88.8 degrees, cost was set to 1.5 degrees and threshold set to 8 degrees.

Let’s try changing up both the cost and the threshold (while keeping the other constant) and see how it might affect the official dates.

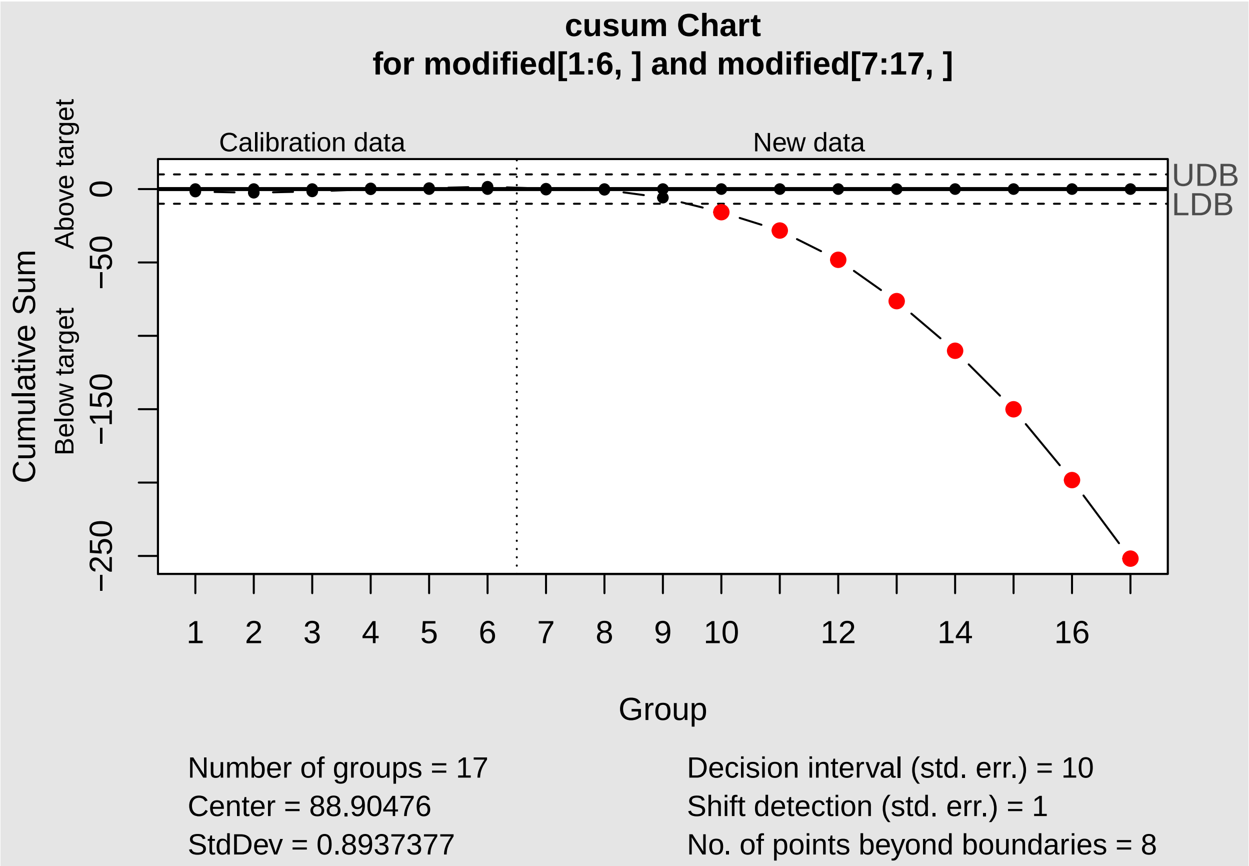

Cost VS Date
Sep 21
Sep 14
Sep 07
Aug 31
Date
1
2
3
4
5
Cost_Value



As we can see, in both plots the change seems to be the end of August/early September which makes sense intuitively. Let's explore the second method.

Method 2 - Using QCC package

```
#Using package

library(qcc)

#Data Manipulation so that is accepted in function
modified <- averages_day[1:119,]
attach(modified)
modified <- qcc.groups(Temperature, rep(1:17, each = 7))

qcc:: cusum(modified[1:6,], newdata=modified[7:17,], decision.interval = 10, se.shift = 1)
```



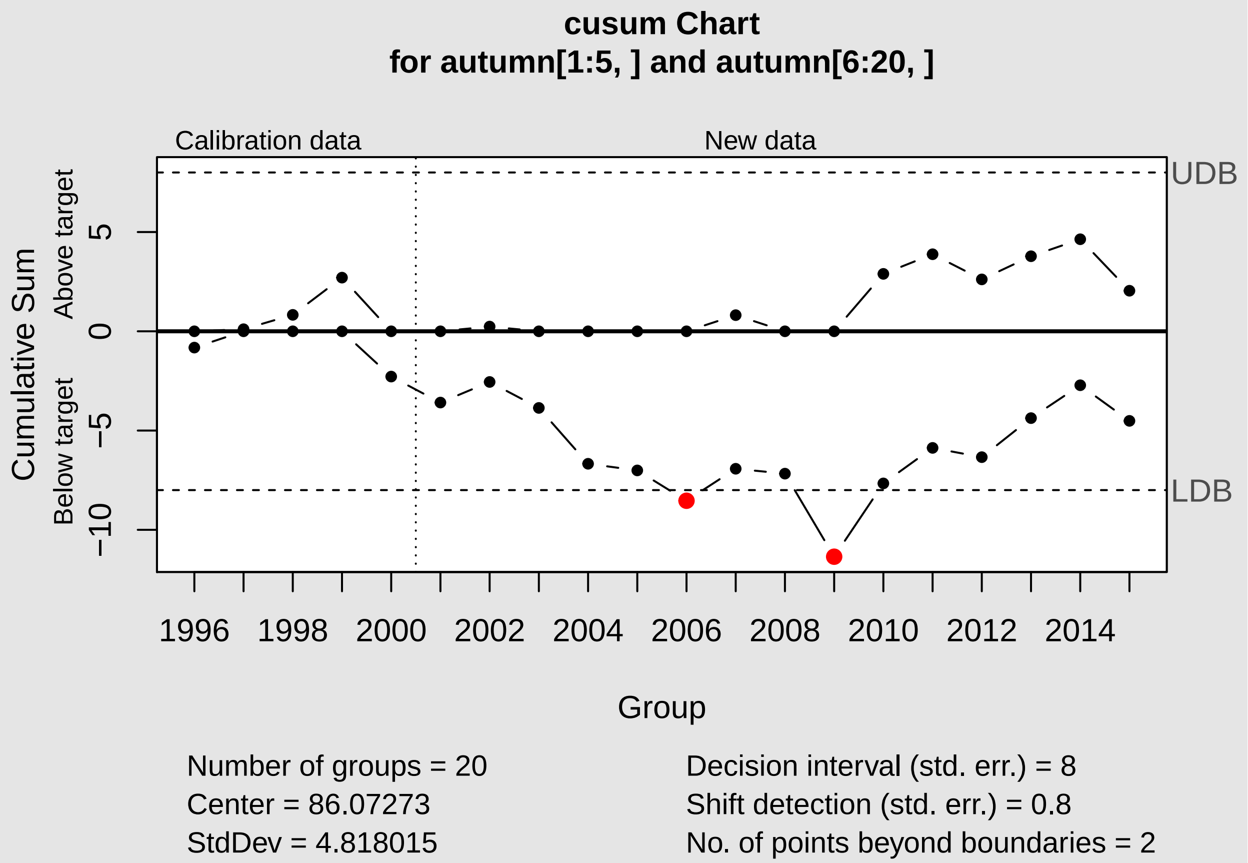
```
## List of 18
##  $ call              : language qcc::cusum(data = modified[1:6, ], decision.interval = 10, se.shift =
##  $ type              : chr "cusum"
##  $ data.name         : chr "modified[1:6, ]"
##  $ data              : num [1:6, 1:7] 88.8 89.2 87 89.5 89.2 ...
##   ..- attr(*, "dimnames")=List of 2
##  $ statistics        : Named num [1:6] 88.2 88.5 89 89.2 89.1 ...
##   ..- attr(*, "names")= chr [1:6] "1" "2" "3" "4" ...
##  $ sizes             : Named int [1:6] 7 7 7 7 7 7
##   ..- attr(*, "names")= chr [1:6] "1" "2" "3" "4" ...
##  $ center            : num 88.9
##  $ std.dev           : num 0.894
##  $ newstats          : Named num [1:11] 88.6 88.7 86.9 85.4 84.5 ...
##   ..- attr(*, "names")= chr [1:11] "7" "8" "9" "10" ...
##  $ newdata           : num [1:11, 1:7] 87.9 90.3 88.5 85.2 85 ...
##   ..- attr(*, "dimnames")=List of 2
##  $ newsizes          : Named int [1:11] 7 7 7 7 7 7 7 7 7 7 ...
##   ..- attr(*, "names")= chr [1:11] "7" "8" "9" "10" ...
##  $ newdata.name      : chr "modified[7:17, ]"
##  $ pos               : num [1:17] 0 0 0 0.459 0.663 ...
##  $ neg               : num [1:17] -1.692 -2.496 -1.587 -0.129 0 ...
##  $ head.start        : num 0
##  $ decision.interval: num 10
##  $ se.shift          : num 1
##  $ violations       :List of 2
##  - attr(*, "class")= chr "cusum.qcc"
```

The plot above was generated by manipulating our initial data and then feeding it into the cusum function in ‘qcc’ package. While the data could be fed so that changes were tracked daily like they were in method 1, I instead created samples of each week. The mean was calculated from the first six weeks (Beginning of July to mid August) and the algorithm was monitoring for changes in temperatures right after that point.

The threshold was set to 10 and the cost was set to 1. As expected, there is no increase in temperatures mid August onwards. However, there is definitely a decrease in temperatures as indicated by the line curving below. The only question is what parameters do we want to set to define when that change occurs. Luckily, it is very easy to alter the parameters in the cusum function if one desires. According to the parameters I set in the model above, the change occurs around week 10, which is approximately the beginning of September, and which matches the results from the the manual cusum function in method 1.

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)



```
## List of 18
##  $ call             : language qcc::cusum(data = autumn[1:5, ], decision.interval = 8, se.shift = 0.8
##  $ type             : chr "cusum"
##  $ data.name        : chr "autumn[1:5, ]"
##  $ data             : num [1:5, 1:22] 84 84 90 82 92 88 87 91 89 90 ...
##   ..- attr(*, "dimnames")=List of 2
##  $ statistics       : Named num [1:5] 84.8 86.6 87.2 88.4 83.3
##   ..- attr(*, "names")= chr [1:5] "1996" "1997" "1998" "1999" ...
##  $ sizes            : Named int [1:5] 22 22 22 22 22
##   ..- attr(*, "names")= chr [1:5] "1996" "1997" "1998" "1999" ...
##  $ center           : num 86.1
##  $ std.dev          : num 4.82
##  $ newstats         : Named num [1:15] 84.3 86.7 84.3 82.8 85.3 ...
##   ..- attr(*, "names")= chr [1:15] "2001" "2002" "2003" "2004" ...
##  $ newdata          : num [1:15, 1:22] 90 91 88 82 84 85 94 78 87 89 ...
##   ..- attr(*, "dimnames")=List of 2
##  $ newsizes         : Named int [1:15] 22 22 22 22 22 22 22 22 22 22 ...
##   ..- attr(*, "names")= chr [1:15] "2001" "2002" "2003" "2004" ...
##  $ newdata.name     : chr "autumn[6:20, ]"
##  $ pos              : num [1:20] 0 0.104 0.828 2.703 0 ...
##  $ neg              : num [1:20] -0.821 0 0 0 -2.282 ...
##  $ head.start       : num 0
##  $ decision.interval: num 8
##  $ se.shift         : num 0.8
##  $ violations       :List of 2
##  - attr(*, "class")= chr "cusum.qcc"
```

This model is very similar to the previous model except that the data was transformed so the samples are of temperatures across different years. Each sample mean was of temperatures between late-August to mid-September which is around the time the temperatures typically change. It seems that (according to the parameters set) there does not seem to be any strong evidence that temperatures have increased in Atlanta over the 20 years during late August/early September.

	x
1996	84.81818
1997	86.59091
1998	87.22727
1999	88.40909
2000	83.31818
2001	84.31818
2002	86.72727
2003	84.31818
2004	82.77273
2005	85.31818
2006	84.09091
2007	87.31818
2008	85.40909
2009	81.36364
2010	89.45455
2011	87.50000
2012	85.18182
2013	87.68182
2014	87.36364
2015	83.81818

Even from examining the means of the temperatures around that time during each year, we can see that there hasn't been some noteworthy increase in temperatures during such time.