ISYE 6501 Homework 1

Due: June 3, 2021

1 k-Means clustering

1.1 Question 4.1

One situation in everyday life in which a clustering model would be useful is clustering/classifying (e)books into unique categories. I came across this idea in a previous OMSA course, CSE 6040, which used spectral co clustering which is in the same spirit. That idea used Amazon user data such as previous books a customer reviewed, their rating & review of said book and more to co-cluster users and books. To provide a different example, if one wanted to cluster books of unknown genre, they could use predictors such as: book title, book author, book length, the book's top used words (not including stop words) and even other user data such which books were commonly purchased with the book etc. I believe strongly that these could very well cluster (e)books into unique genres such as: biographies, politics, sports, food, textbooks, mystery, sci-fi, children's books etc.

1.2 Question 4.2

Using the classic *iris* dataset, commonly used in machine learning, I examined how well the k-means clustering algorithm performs. By inspecting the data, one can see that there are four predictors, sepal and petal length & width each given in cm. Because the response variables are all given in the same unit, there is really no need to scale or preprocess the data before running the clustering. Another key fact that can be seen from peeking at the data is that it is split into three distinct classes (types of flowers) so even though we can explore clustering this data into $k \neq 3$ classes, the context of the actual data should be considered when further diving in. By using some built-in functions in R, such as *kmeans* and *fviz_cluster* [1], I clustered the data into $k = \{2, 3, 4, 5\}$ using all of the predictors and visualized the separation of clusters. Because the dimension of the data is greater than two, I visualized the first two principal components (predictors that explain the most variance). See Figure 1 for the clustering results. One could argue that the data naturally splits into a number of clusters that is not three, in fact, the total within cluster sum of squares (what is minimized during the k-means algorithm) is lowest for $k = 4$ clusters (57.26562). Note, that there are many distance metrics but this built in kmeans function uses the 2-norm distance. So although $k = 4$ is actually "best" for this data, I computed the actual accuracy for the $k = 3$ clustering because of what is knows about the actual data. After the clustering, I compared the predicted classes to the actual classes and found that the $k = 3$ clustering produced an accuracy of 89.33%. See Appendix A for code.

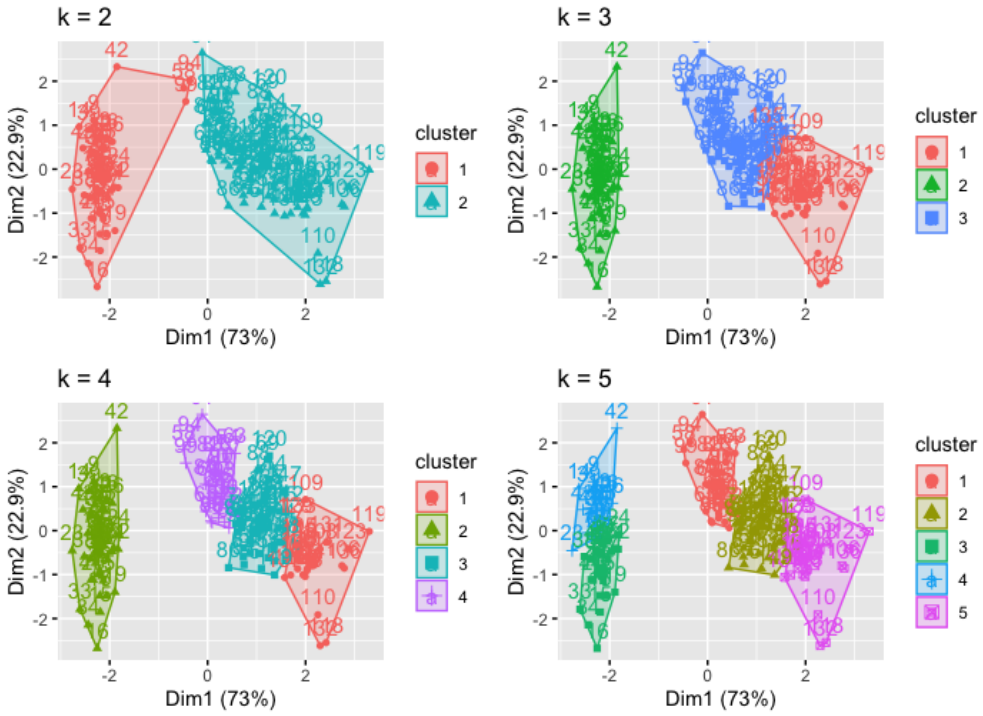

Figure 1: First two principal components after clustering the iris dataset into k clusters.

2 Outlier detection

2.1 Question 5.1

Using the *uscrime* dataset, I aimed to see if the last column (number of crimes per 100,000 people) has any outliers. The grubbs.test function contained in the *outliers* package in R allows for easy analysis of whether there are one or more outliers on either end of the data. Because it is a tedious and time intensive process to explore large datasets to determine if or where any potential outliers exist, the data can be plotted in various fashions to make a few conclusions. I first used a Box and whisker plot to explore the data and visualize how many and where outliers may exist. As seen in Figure 2, there are potentially three outliers in the upper tail of the data. I could then use the handy Grubbs' test function to check the three points. After, checking the highest most point (or worst potential outlier), it was found to be an outlier with p-value = 0.07887. Because this p-value is more than 0.05, I can conclude that although it appears to be an outlier and is indeed close, the null hypothesis cannot be rejected and therefore the point is not an outlier. In a similar light, the other two potential outliers will not be true outliers either because they are even closer to the bulk of the data. See Appendix B for code.

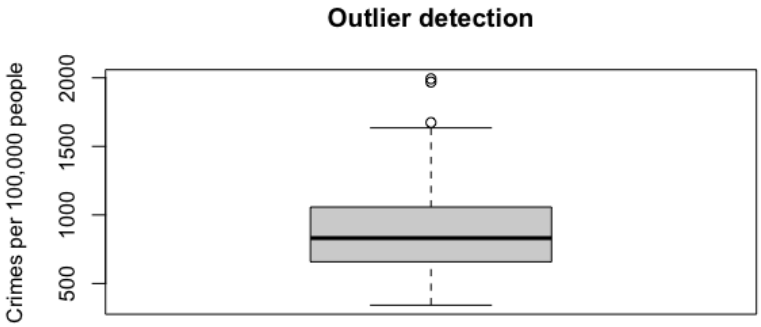

Figure 2: Box plot of crimes per 100,000 people dataset.

3 Change detection and CUSUM

3.1 Question 6.1

Change detection models are often employed when handling time-series data and one specific example that I have encountered in a past engineering course is temperature monitoring. Often, room air conditioning is control from a sensor that reads the rooms temperature and a Schmitt trigger circuit. The AC turns on once the room temp reads above a certain temperature and off when it dips below another (so that it isn't constantly turning off and on). But what if the contents in the room like expensive material/hardware are extremely sensitive to changes in temperature? In this case we may want to use a change detection model in our room such that if drastic changes start to occur, an alarm sounds. Applying the CUSUM technique, we can tune the change detection sensitivity by altering the critical value (C) and threshold (T). Recall that increasing C decreases the sensitivity of the change detection so if the alarm was ringing too often and was taking the time out of emergency workers' days, one could increase C from zero to five degrees. Alternatively, if there is really only a serious issue when the change is really large, one might choose to increase T. There is a fine trade-off between C and T based on the trade off between a false negative or a false positive for example.

3.2 Question 6.2

3.2.1 Question 1.

The mathematical definition of what is described above can be seen in Equation 1 below where s_t is the cumulative sum (CUSUM) at time t, μ is the mean of the observed data, and C & T are the previously described. A change is detected when $s_t \geq T$.

$$s_t = max\{0, s_{t-1} + (x_t - \mu - C)\} \tag{1}$$

I implemented this CUSUM technique in R on the July-October daily high-temperature data for Atlanta from 1996-2015 as seen in Appendix C. By using this CUSUM technique I could examine when the unofficial end of summer is i.e. when the weather changes to cold. Note that to detect decreases, the signs in front of s_t and μ are swapped in Equation 1. After an extensive parameter search for C and T, I found two reasonable values that trigger a detection for each year, C = 3 and T = 15. Table 1 summarizes the predicted unofficial change of summer based on this model.

Year	Detected date of summer's end
1996	29-Sep
1997	25-Sep
1998	3-Oct
1999	21-Sep
2000	6-Sep
2001	25-Sep
2002	25-Sep
2003	29-Sep
2004	21-Sep
2005	7-Oct
2006	29-Sep
2007	18-Sep
2008	11-Oct
2009	3-Oct
2010	28-Sep
2011	7-Sep
2012	1-Oct
2013	16-Aug
2014	26-Sep
2015	25-Sep

Table 1: Using CUSUM to detect the end of summer

3.2.2 Question 2.

Using the CUSUM approach described above, I predicted whether Atlanta's summer temperatures have been rising by using the CUSUM technique. The data I used was the average summer temperature per year. After a quick parameter search I discovered T = 3 and C = 1 give a reasonable detection that the average temperature starts to change (increase in this instance) 14 years after 1995 or 2009. I could then verify this prediction/detection by actually plotting the average summer temperature every year. See Figure 3 below. Turns out the CUSUM detection did an excellent good job, detecting the drastic spike in 2009. The detection also excels by detecting a change in 2010 but no year thereafter and what can be seen in Figure 3 is that the average temperature promptly rescinds back to the average. Therefore it can be concluded that there is some subtle increase in average temperature during this time period but mostly starting in 2009.

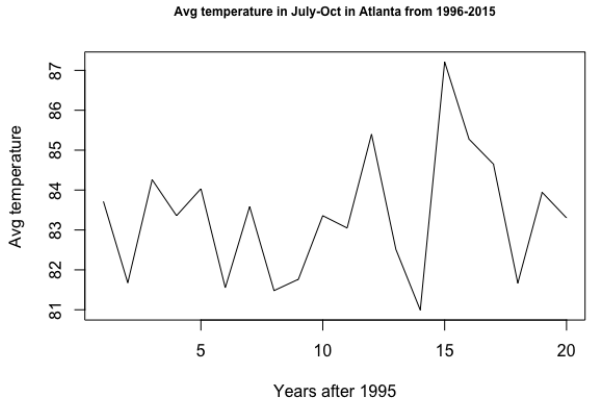

Figure 3: Inspecting when Atlanta's summer has gotten warmer

References

[1] https://uc-r.github.io/kmeans_clustering

Appendix A k-means clustering code

```
library(factoextra) # clustering algorithms & visualization
library(gridExtra)
data <- read.table("iris.txt", header=TRUE)

cluster1 <- kmeans(data[,1:4], centers=2, iter.max = 100)
cluster2 <- kmeans(data[,1:4], centers=3, iter.max = 100)
cluster3 <- kmeans(data[,1:4], centers=4, iter.max = 100)
cluster4 <- kmeans(data[,1:4], centers=5, iter.max = 100)

#print(cluster)
print(cluster2$cluster)
#print(sum(cluster1$withinss))
mapping <- c()
for (i in data[,5]) {
  if (i == "setosa") {mapping <- c(mapping, 2)}
  if (i == "versicolor") {mapping <- c(mapping, 3)}
  if (i == "virginica") {mapping <- c(mapping, 1)}
}
print(mapping)
acc <- sum(cluster2$cluster == mapping) / nrow(data)

plot1 <- fviz_cluster(cluster1, data = data[,1:4]) + ggtitle("k = 2")
plot2 <- fviz_cluster(cluster2, data = data[,1:4]) + ggtitle("k = 3")
plot3 <- fviz_cluster(cluster3, data = data[,1:4]) + ggtitle("k = 4")
plot4 <- fviz_cluster(cluster4, data = data[,1:4]) + ggtitle("k = 5")

grid.arrange(plot1, plot2, plot3, plot4, nrow = 2)
```

Appendix B Outlier code

```
library(outliers)
data <- read.table("uscrime.txt", header=TRUE)
crimes <- data[,16]
crimesdf <- data.frame(crimes)

outlier <- grubbs.test(crimes, type=10)
print(outlier)

boxplot(crimesdf, main="Outlier detection", ylab="Crimes per 100,000 people")
```

Appendix C Change detection code

```
data <- read.table("temps.txt", header=TRUE)
avg_array <- c()
for (year in 2:ncol(data)) {
  avg_array <- c(avg_array, sum((data[,year])/nrow(data)))
}

#180, 3
t = 15
c = 3
for (i in 2:ncol(data)) {
  cusum = 0
  for (j in 1:nrow(data)) {
    cusum <- max(0, cusum - data[j,i] + avg_array[i-1] - c)
    #print(cusum)
    if (cusum >= t) {print(data[j,1])}
  }
  print("--------------------------------")
}
plot(avg_array, type = "l", main='Avg temperature in July-Oct in Atlanta from 1996-2015',
     xlab='Years after 1995', ylab='Avg temperature', cex.main=0.75)

# CUSUM on avg temp data
t = 3
c = 0
cusum = 0
counter = 0
avg <- sum(avg_array)/length(avg_array)
for (k in avg_array) {
  cusum <- max(0, cusum + k - avg - c)
  #print(cusum)
  if (cusum >= t) {print(counter)}
  counter <- counter + 1
}
```