GTx: ISYE6501x - Homework 2

Muh Alif Ahsanul Islam

5/24/2019

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.

Answer:

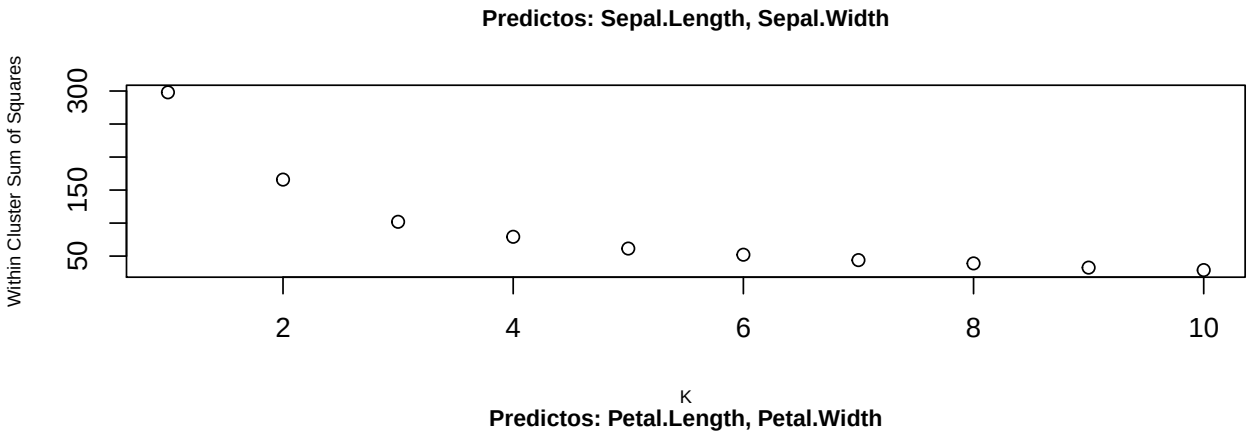
If you are a business owner and want to try a new pricing system, you need to know how your customers will react. Instead of analyzing each customers as a person, you can make groups of customers to represent all you customers. Customers in a group will have similar behavior. To decide which customer goes to which group you can use clustering analysis.
The predictors:
1. Monthly purchase
People with low income tends to be more sensitive to price changes than higher income. Because you don't want to lose many customers, you have to make the trade off between high margin and high volume.
2. Purchasing frequency
Purchase frequency indicates how often your customer buy for a given period. Loyal customer will have higher purchasing frequency.
3. Age
Different age group will have different need, lifestyle, finance, and behavior.
4. Customer satisfaction
Customer that is not satisfied with your service might be more incline to swith to your competitor if you further increase the price.

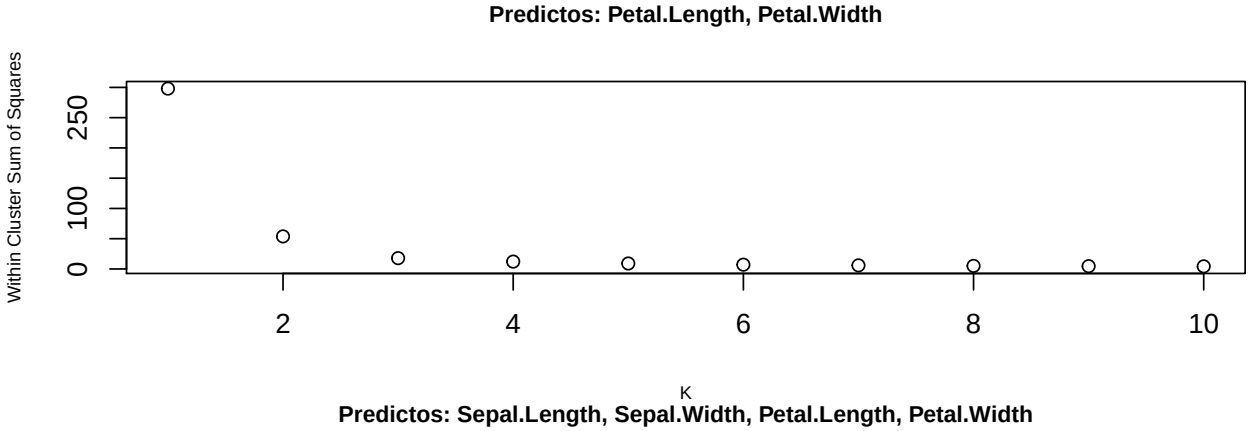
Question 4.2

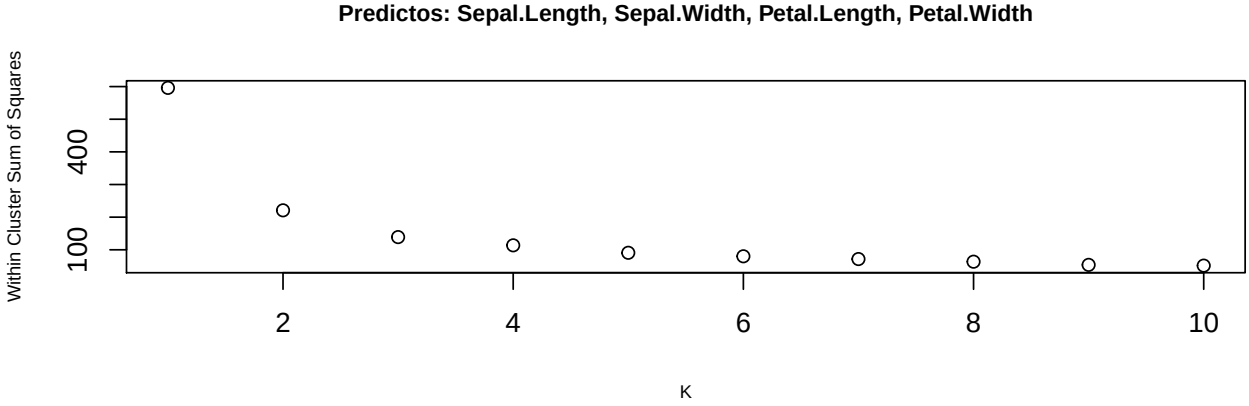
The iris data set iris.txt contains 150 data points, each with four predictor variables and one categorical response. The predictors are the width and length of the sepal and petal of flowers and the response is the type of flower. The response values are only given to see how well a specific method 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.

```
rm(list=ls())
iris = read.table('iris.txt', header=TRUE)
iris = na.omit(iris)
iris_scaled = iris
iris_scaled[,-5] = scale(iris[,-5])
k_vals = c(1:10)
predictor_cols = list(c(1,2), c(3,4), c(1,2,3,4))
k_res = NULL
for (i in 1:length(predictor_cols)){
  tot_with_pred = rep(NA, length(k_vals))
  for (k in k_vals){
    pred = predictor_cols[[i]]
    predictor_str = paste(names(iris)[pred], collapse=', ')
    kmean_model = kmeans(iris_scaled[,pred], centers=k, nstart=5)
    tot_with_pred[k] = kmean_model$tot.withinss
    conf_table = table(kmean_model[['cluster']], iris_scaled$Species)
    pred_error = 0
    for (j in 1:ncol(conf_table)){
      pred_error = pred_error + sum(conf_table[,j]) - max(conf_table[,j])
    }
    if (k!=nlevels(iris_scaled$Species)){
      pred_error = NA
    }
    else{
      pred_error = 100 * pred_error / nrow(iris_scaled)
    }
    k_res = rbind(k_res,
                  data.frame(i, k, kmean_model$tot.withinss,
                             pred_error))
  }

  plot(x=k_vals, y=tot_with_pred, ylab='Within Cluster Sum of Squares',
       cex.lab=0.7, xlab='K',
       main=sprintf('Predictos: %s', predictor_str), cex.main=0.8)
}
```








```
colnames(k_res) = c('predictor_set', 'k', 'tot_with', 'prediction_error')
```

The within-cluster sum of squares is a measure of the variability of the observations within each cluster. In general, a cluster that has a small sum of squares is more compact than a cluster that has a large sum of squares. Clusters that have higher values exhibit greater variability of the observations within the cluster.

```
k_res
```

```
##    predictor_set  k   tot_with prediction_error
## 1              1  1 298.000000               NA
## 2              1  2 165.838698               NA
## 3              1  3 101.931264         22.66667
## 4              1  4  79.220574               NA
## 5              1  5  61.421507               NA
## 6              1  6  52.091678               NA
## 7              1  7  43.891336               NA
## 8              1  8  38.921817               NA
```

```
## 9              1  9  32.545137              NA
## 10             1 10  28.811446              NA
## 11             2  1 298.000000              NA
## 12             2  2  53.807656              NA
## 13             2  3  17.906783         4.00000
## 14             2  4  12.201483              NA
## 15             2  5   9.091120              NA
## 16             2  6   7.123243              NA
## 17             2  7   5.954765              NA
## 18             2  8   5.089784              NA
## 19             2  9   4.635036              NA
## 20             2 10   4.347700              NA
## 21             3  1 596.000000              NA
## 22             3  2 220.879294              NA
## 23             3  3 138.888360        16.66667
## 24             3  4 113.649811              NA
## 25             3  5  90.840444              NA
## 26             3  6  80.103777              NA
## 27             3  7  71.611898              NA
## 28             3  8  63.575146              NA
## 29             3  9  53.804683              NA
## 30             3 10  51.529389              NA
```

Answer of Question 4.2

I proposed 3 kinds of predictors:
1. Sepal length and sepal width
2. Petal length and petal width
3. Sepal length, sepal width, petal length, and petal width

From 3 sets of predictors the set 2 has lowest prediction error and within clusters sum of squares.

Using elbow method, we can determine best K. In the graph above we can see that the marginal decrease in "Within Cluster Sum of Squares" diminishes when we move from K=3 to K=4 and keep decrease when K is increased. This indicates that "Within Cluster Sum of Squares" using K=4 is not that different with using K=3 (adding more clusters doesn't give us better model).

Best K = 3
Best Predictors: petal length and petal width
Prediction accuracy: 96%

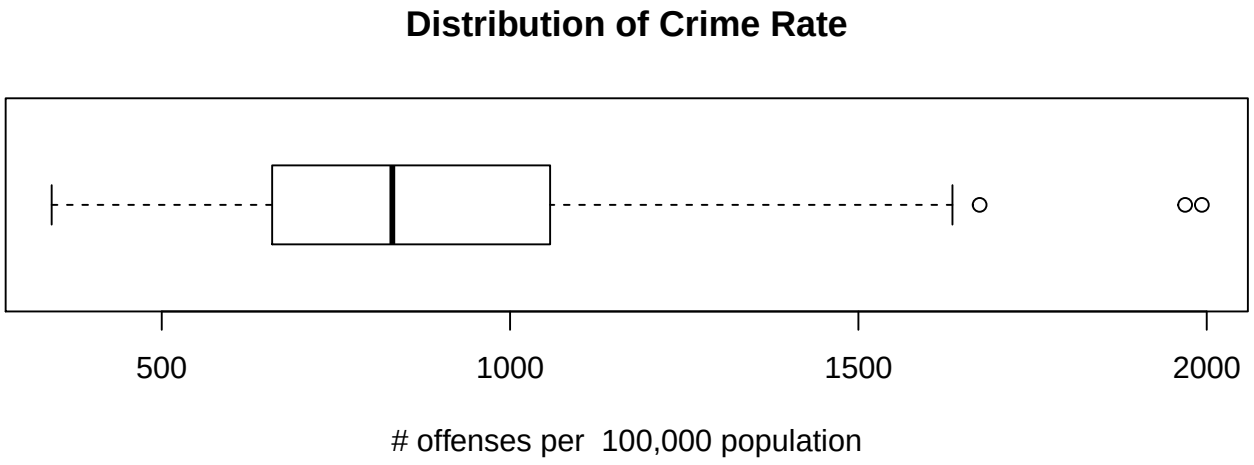
Question 5.1

Using crime data from the file uscrime.txt, 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.

```
rm(list=ls())
crime = read.table('uscrime.txt', header=TRUE)
```

Check the distribution of variable.

```
boxplot(crime$Crime, main='Distribution of Crime Rate',
        xlab='# offenses per  100,000 population', horizontal=TRUE)
```



Try Grubbs test for an outlier

```
grubb = grubbs.test(crime$Crime, type=10)
grubb
```

```
##
##  Grubbs test for one outlier
##
## data:  crime$Crime
## G = 2.81290, U = 0.82426, p-value = 0.07887
## alternative hypothesis: highest value 1993 is an outlier
```

Answer of Question 5.1

From distribution above we can see some points lie far outside Q3 + 1.5 IQR. We suspect these are outliers. To test it, we can use Grubbs Test. Grubbs test assumes normal distribution.

Null hypothesis: there are no outliers

First we check whether the highest point is an outlier or not. We set the significance level to be 0.05. From grubbs test we got the P value is 0.08 which is bigger than our standard 0.05, this indicates that we can not reject null hypothesis. Because the highest point is not an outlier, the points below the highest also not an outlier (we only do univariate analysis).

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?

Answer

I am working at analytics company. One of our client is a power plant company. We helped our client to make a system that send signal to plant engineers when machine needs to be repaired. One of parameter that we monitor is temperature, when temperature is constantly high the machine needs to stop and repaired. We don't want the machine to stop when there is only momentary upward spike in temperature. CUSUM can be implemented in this case for detecting an increase in temperature.
From historical data, we can do CUSUM analysis and tune threshold and critical value so that it will minimize false positive and false negative.
Choosing threshold: bigger threshold will increase chance of false negative, smaller threshold increase chance of false positive.
Choosing critical value: bigger value will make the model slow in response, smaller value will make model fast in response.

Question 6.2.1

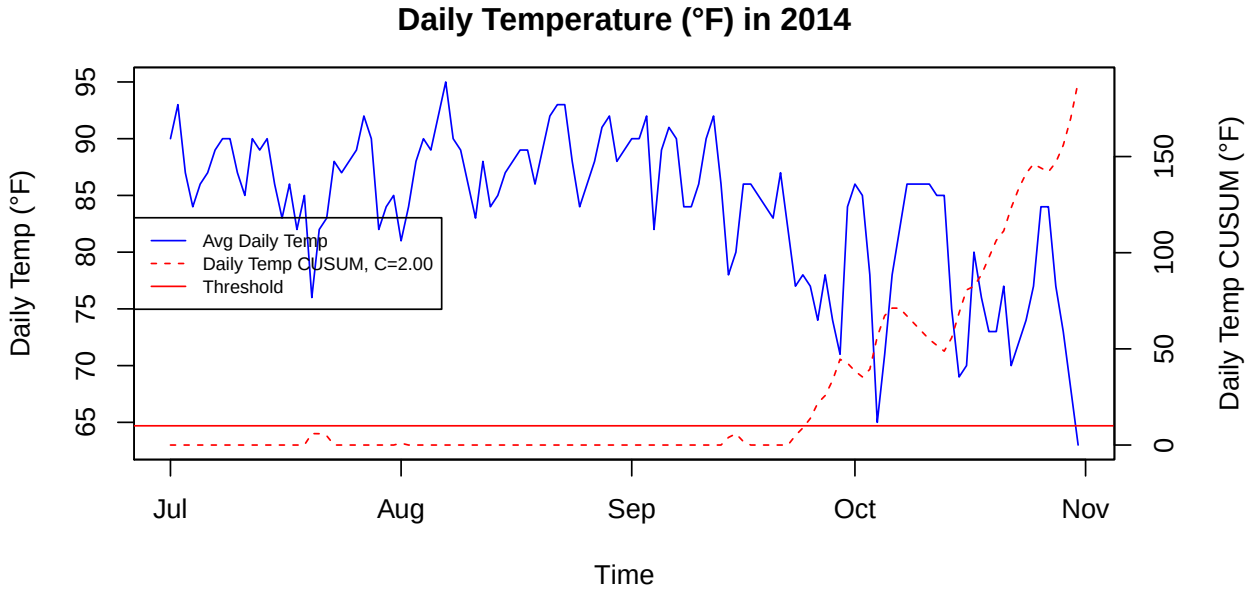
Using July through October daily-high-temperature data for Atlanta for 1996 through 2015, use a CUSUM approach to identify when unofficial summer ends (i.e., when the weather starts cooling off) each year.

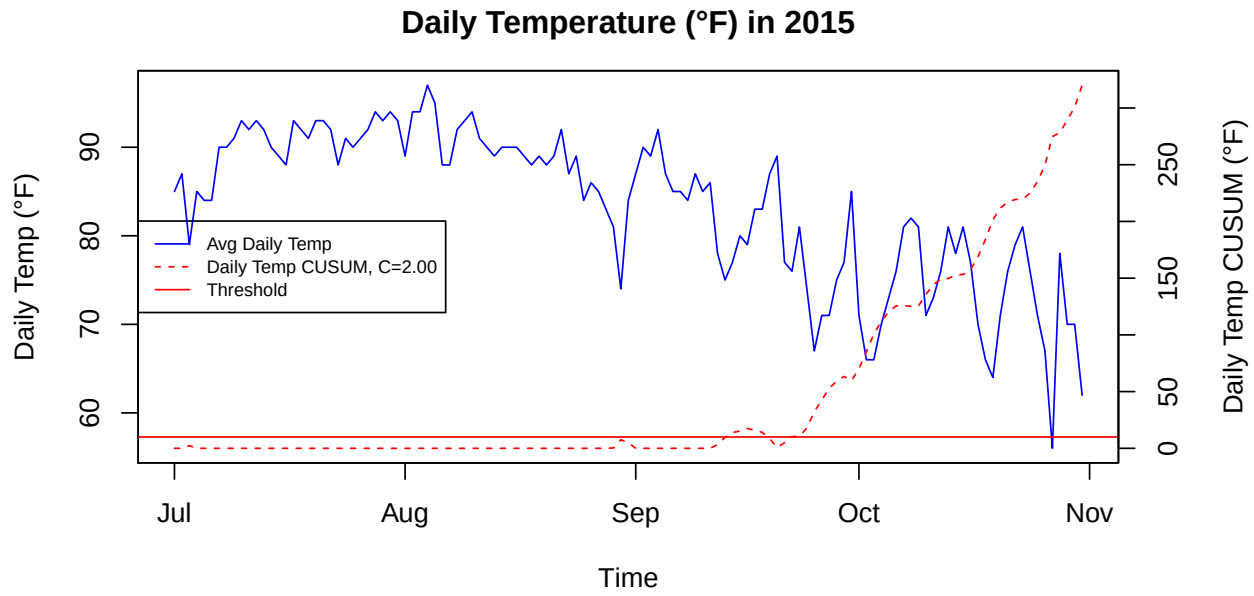
Because the problem is to find when the weather start cooling off, it is not detecting an increase, but detecting a decrease.

Plots for some of the years:

```
rm(list=ls())
temps = read.table('temps.txt', header=TRUE)
temps$DAY = as.Date(temps$DAY, format='%d-%b')
crit_val = 2
threshold = 10
years_col = names(temps)[c(2:length(names(temps)))]
years_int = rep(NA, length(years_col))
summer_end_idx = rep(NA, length(years_col))
summer_end_date = rep(NA, length(years_col))
summer_end_date_same_year = rep(NA, length(years_col))
temp_summer_end = rep(NA, length(years_col))
class(summer_end_date) = 'Date'
class(summer_end_date_same_year) = 'Date'
for (k in 1:length(years_col)){
  year_temp = as.matrix(temps[years_col[k]])
  avg_year_temp = mean(year_temp)
  S = rep(0, length(year_temp))
  i_summer_end = 0
  i_flag = FALSE
  for (i in 2:length(S)){
    S[i] = max(0, S[i-1] - year_temp[i] + avg_year_temp - crit_val)
    if (S[i] > threshold & i_flag==FALSE){
      i_summer_end = i
      i_flag = TRUE
    }
```

```
  }
  summer_end_idx[k] = i_summer_end
  year_str = substr(years_col[k], 2, nchar(years_col[k]))
  years_int[k] = as.integer(year_str)
  temp_summer_end[k] = year_temp[i_summer_end]
  month_day_str = format(temps$DAY[i_summer_end], '%m-%d')
  summer_end_date[k] = as.Date(sprintf('%s-%s', year_str, month_day_str),
                               format='%Y-%m-%d')
  summer_end_date_same_year[k] = as.Date(sprintf('2019-%s', month_day_str),
                               format='%Y-%m-%d')
  if (any(c('2014', '2015') == year_str)){
    par(mar = c(5, 5, 3, 5))
    plot(x=temps$DAY, y=year_temp, type ="l", ylab = "Daily Temp (°F)",
         main = sprintf("Daily Temperature (°F) in %s", year_str),
         xlab = "Time", col = "blue")
    par(new = TRUE)
    plot(x=temps$DAY, y=S, type = "l", xaxt = "n", yaxt = "n",
         ylab = "", xlab = "", col = "red", lty = 2)
    abline(h=threshold, col='red')
    axis(side = 4)
    mtext("Daily Temp CUSUM (°F)", side = 4, line = 3)
    legend("left", c("Avg Daily Temp",
                     sprintf("Daily Temp CUSUM, C=%.2f", crit_val),
                   'Threshold'),
           col = c("blue", "red", 'red'), lty = c(1, 2), cex=0.7)
  }
}
```





The unofficial summer end dates:

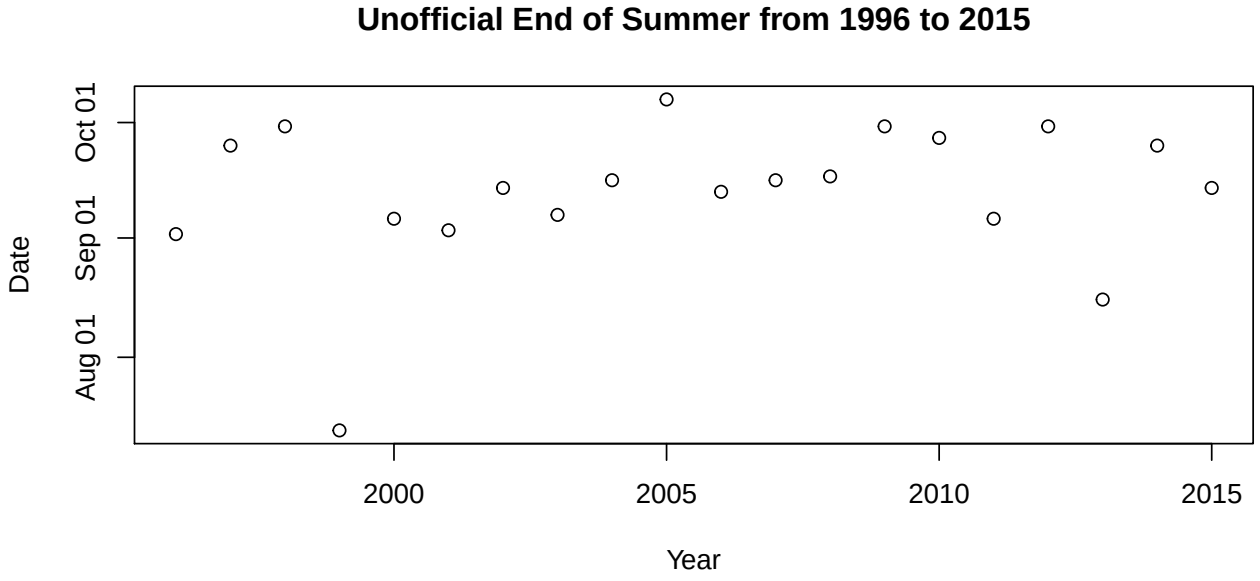
```
summer_end_date
```

```
##  [1] "1996-09-02" "1997-09-25" "1998-09-30" "1999-07-13" "2000-09-06"
##  [6] "2001-09-03" "2002-09-14" "2003-09-07" "2004-09-16" "2005-10-07"
## [11] "2006-09-13" "2007-09-16" "2008-09-17" "2009-09-30" "2010-09-27"
## [16] "2011-09-06" "2012-09-30" "2013-08-16" "2014-09-25" "2015-09-14"
```

Answer of Question 6.2.1

The graph below shows when summer ends for each year. End of summer occur relatively at the same time during 1996 to 2015.

```
plot(x=years_int, y=summer_end_date_same_year,
     main='Unofficial End of Summer from 1996 to 2015',
     xlab='Year', ylab='Date')
```



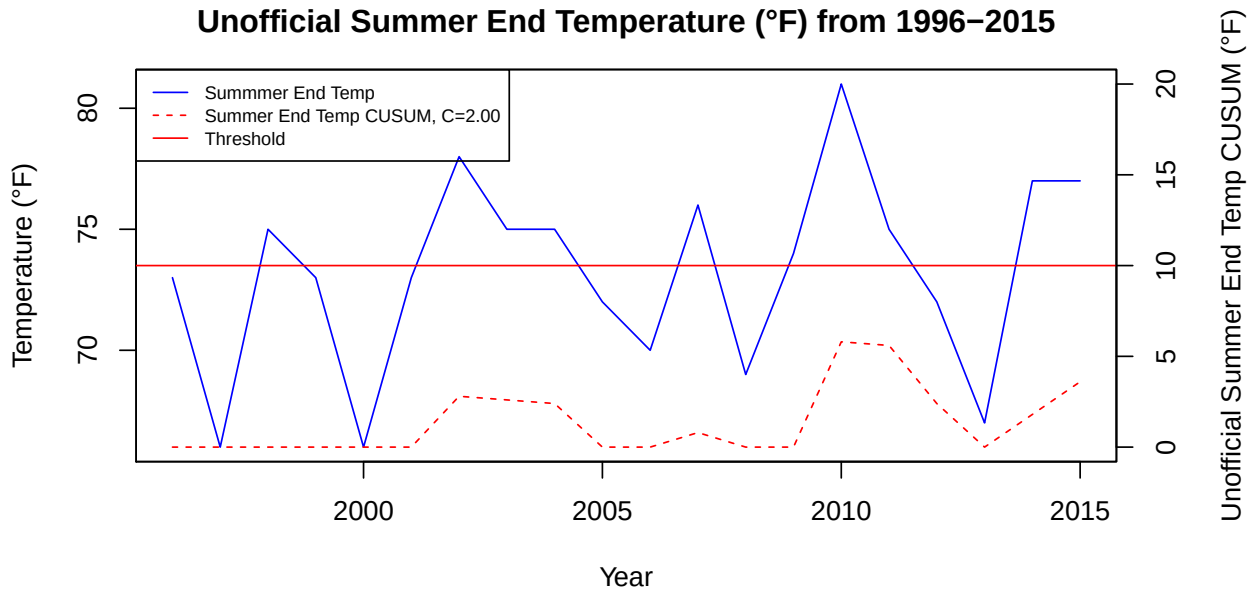
Question 6.2.2

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

To check whether the summer has become warmer at unofficial summer end date, we have to detect change in temperature of each corresponding unofficial summer end date each year from 1996 to 2015.

```
crit_val_temp = crit_val
threshold_temp = threshold
avg_temp_summer_end = mean(temp_summer_end)
S = rep(0, length(years_int))
for (i in 2:length(S)){
  S[i] = max(0, S[i-1] + temp_summer_end[i] - avg_temp_summer_end - crit_val_temp)
}

par(mar = c(5, 5, 3, 5))
plot(x=years_int, y=temp_summer_end, type ="l",
     ylab = "Temperature (°F)",
     main = "Unofficial Summer End Temperature (°F) from 1996-2015",
     xlab = "Year", col = "blue")
par(new = TRUE)
plot(x=years_int, y=S, type = "l", xaxt = "n", yaxt = "n", ylab = "",
     xlab = "", col = "red", lty = 2,ylim=c(0,threshold_temp+10))
abline(h=threshold_temp, col='red')
axis(side = 4)
mtext("Unofficial Summer End Temp CUSUM (°F)", side = 4, line = 3)
legend("topleft", c("Summmer End Temp",
                    sprintf("Summer End Temp CUSUM, C=%.2f", crit_val_temp),
                   'Threshold'),
       col = c("blue", "red", 'red'), lty = c(1, 2), cex=0.7)
```



Answer to Question 6.2.2

From graph above, we see that CUSUM value never cross the threshold. Also we don't see any consistent upward trend of CUSUM value anywhere during 1996-2015.
The result of this analysis suggests that temperature at unofficial summer end in Atlanta is not getting warmer.
