ISYE 6501 Homework 2

Brendan Schell

May 30, 2018

Question 4.1

One situation in my life where a clustering model would be appropriate would be in performing a customer profile for my consulting business. This would allow me to understand the different types of customers that I have and their characteristics.

Five predictors that could potentially be useful for this clustering model would be as follows:

1) Age of customer / company
2) Location of the customer (City / Country)
3) Average contract length
4) Pay rate ($/hr)
5) Email frequency (# Emails / Week)

Question 4.2

Read iris data

```r
df_iris = read.table('4.2irisSummer2018.txt',header=TRUE)
head(df_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
```

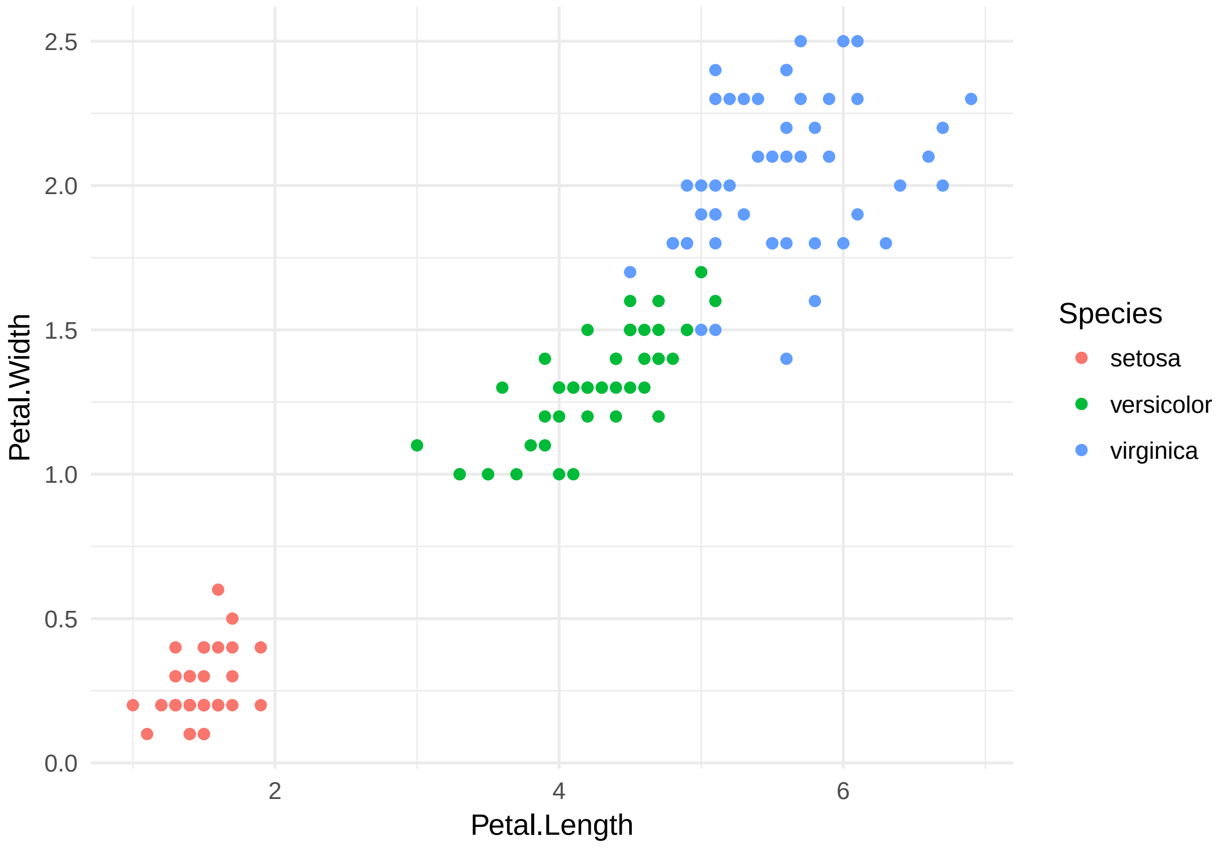
Look at a summary of the dataset to get an overview of the features.

```r
summary(df_iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

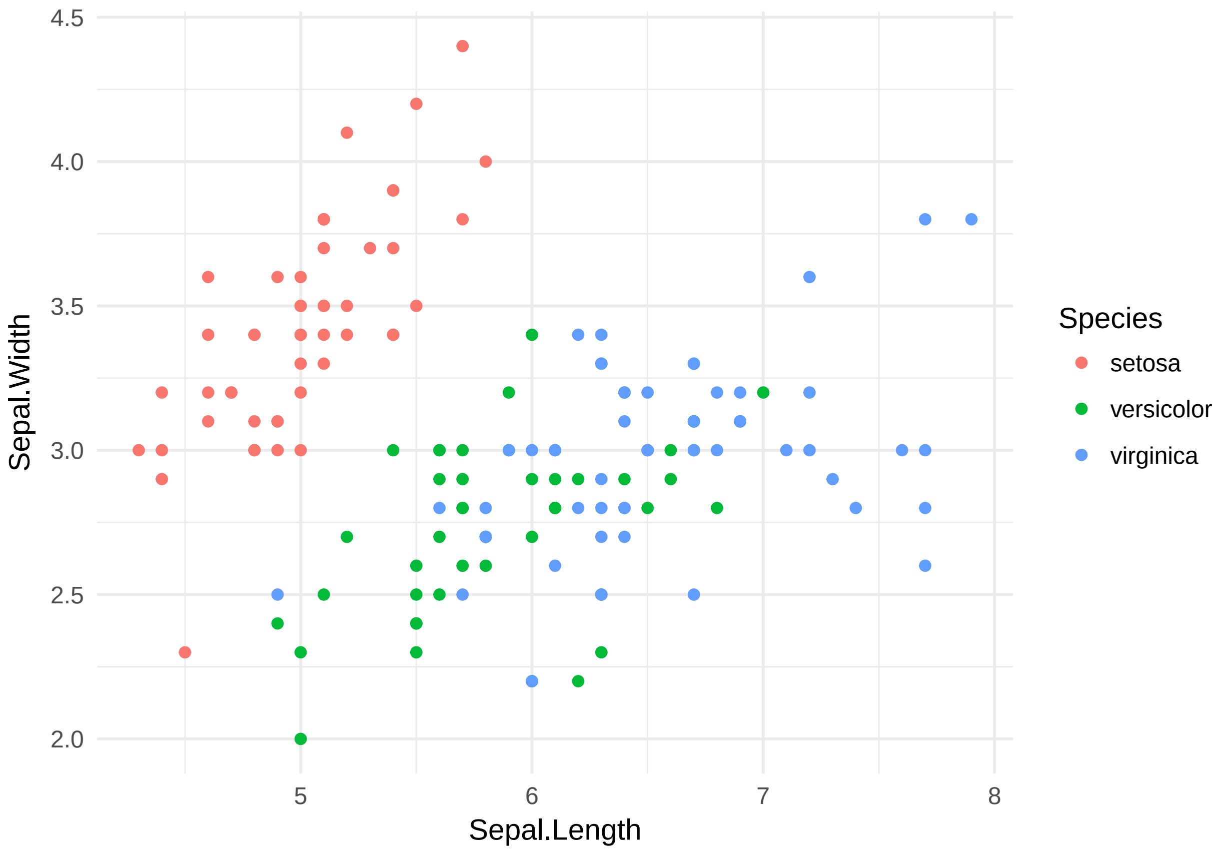
Looking at petal length vs. petal width seems to separate the classes fairly well.

```r
library(ggplot2)

ggplot(df_iris, aes(Petal.Length, Petal.Width, color=Species)) + geom_point() + theme_minimal()
```



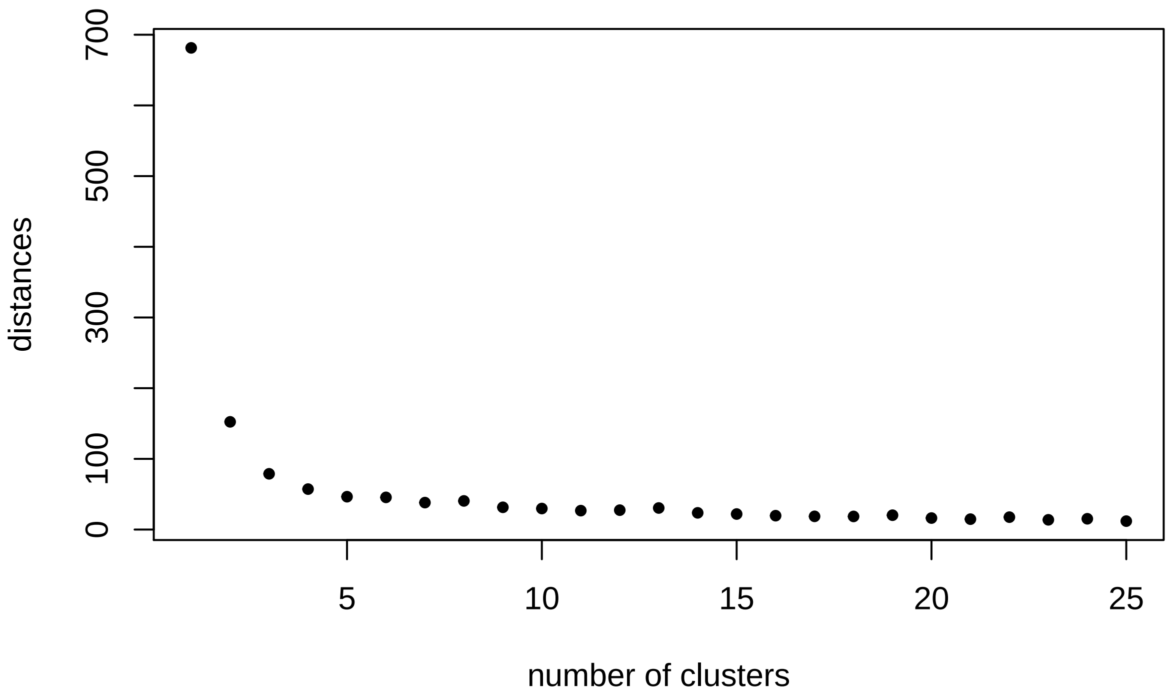
The sepal length and width do not seem to separate the versicolor and virginica classes very well, but do a good job of isolating the setosa class.

```r
ggplot(df_iris, aes(Sepal.Length, Sepal.Width, color=Species)) + geom_point() + theme_minimal()
```



Testing out the k parameter for the number of clusters from 1 to 25 and plotting, we can see that there is an elbow point at around k=3. After k=3, there are diminishing returns in the total sum of squares by increasing the k value past this point.

```r
# empty distance vector
distances <- rep(0,25)

# try k values from 1 to 25 and fit a kmeans model to each
for (k_clusters in 1:25){
  clusters <- kmeans(df_iris[,1:4], k_clusters)
  distances[k_clusters] <- clusters$tot.withinss
}

plot(distances, xlab='number of clusters',pch=20)
```



Choosing the point of diminishing returns (k=3) from the elbow plot and comparing against the actual class labels, we can see that it currently does a perfect job of isolating setosas into cluster 3, a good job at identiying versicolors as cluster 2 (48/50), but does not do very well with the virginica class (36/50) for cluster 1. A total sum of squares of 78.85 is achieved with this model.

```
# fit kmeans with k = 3
iris_clusters <- kmeans(df_iris[,1:4], 3)

# table of cluster  vs species
table(iris_clusters$cluster, df_iris$Species)
```

```
##
##     setosa versicolor virginica
##   1      0          2        36
##   2      0         48        14
##   3     50          0         0
```

```
print(c('total sum of squares:',iris_clusters$tot.withinss))
```

```
## [1] "total sum of squares:" "78.8514414261 46"
```

Normalizing the data to a normal distribution with mean 0 and standard deviation 1 is done so that the data is on the same scale, though there was no noticeable benefit in doing so on this dataset. The best model was obtained for k = 3 using only the Petal.Length and Petal.Width features. This makes sense intuitively since there was a very clear separation boundary between the classes when plotted above.

In this k means model, only 6 classes are mislabelled (2 versicolors identified to the virginica cluster and 4 virginicas to the setosa class). The total sum of squares is also reduced from 78.85 to 17.91.

```
# transform data to mean 0, std 1 normal distribution
df_iris_scaled <- df_iris
for (i in 1:4){
  df_iris_scaled[,i] = (df_iris_scaled[,i] - mean(df_iris_scaled[,i])) / sd(df_iris_scaled[,i])
}
```

```
# fit kmeans with k = 3
iris_clusters <- kmeans(df_iris_scaled[,3:4], 3)
iris_clusters_k4 <- kmeans(df_iris_scaled[,3:4], 4)
# table of cluster  vs species
table(iris_clusters$cluster, df_iris_scaled$Species)
```

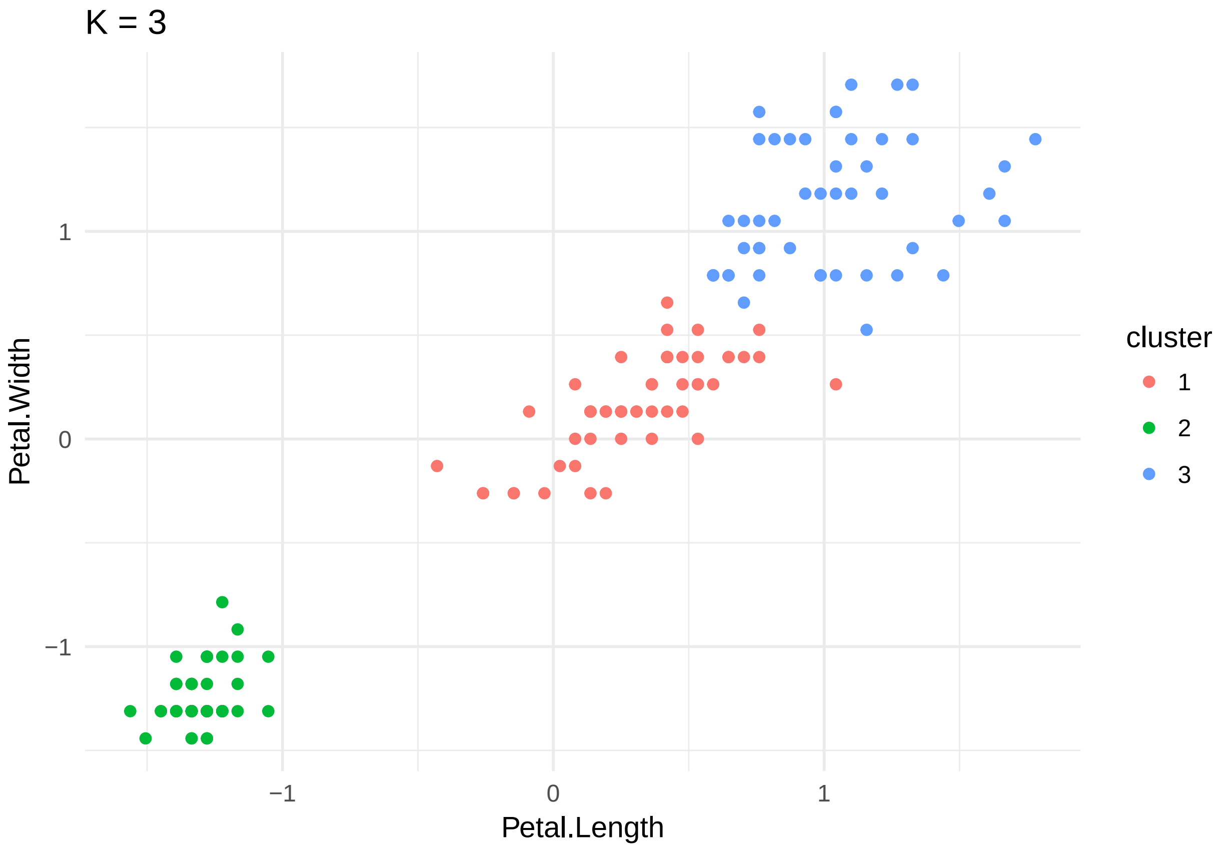
```
##
##     setosa versicolor virginica
##   1      0         48         4
##   2     50          0         0
##   3      0          2        46
```

```
# add clusters to dataframe
df_iris_scaled$cluster <- as.factor(iris_clusters$cluster)
df_iris_scaled$cluster_k4 <- as.factor(iris_clusters_k4$cluster)
# total sum of squares
print(c('total sum of squares: ',iris_clusters$tot.withinss))
```

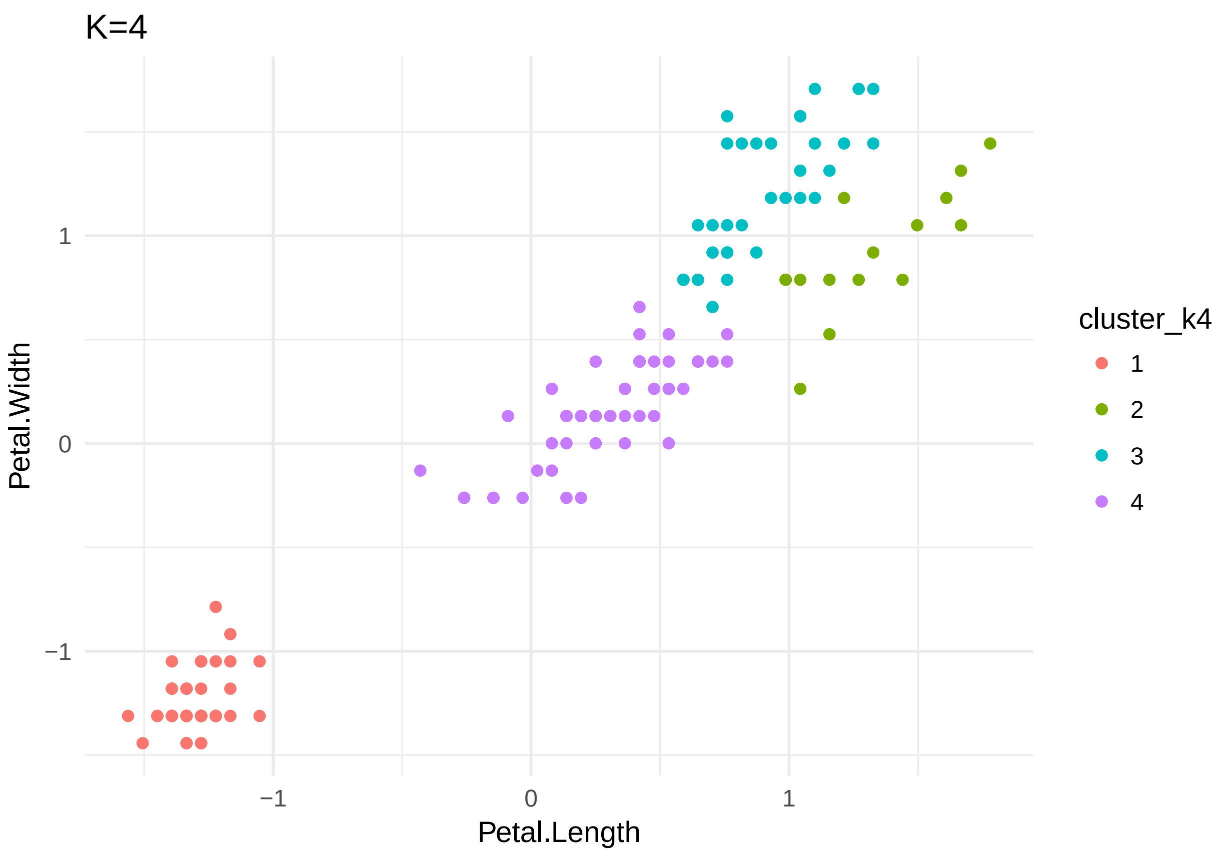
```
## [1] "total sum of squares: " "17.9067828617938"
```

Although a model with k = 4 reduces the total sum of squares from 17.91 to 12.41, the seperation does not intuitively make as much sense from the plot, so a k value of 3 is chosen as ideal for this model.

```
# k = 3
ggplot(data=df_iris_scaled,aes(Petal.Length,Petal.Width,color=cluster)) + geom_point() + theme_minimal(
```



```
# k = 4
ggplot(data=df_iris_scaled,aes(Petal.Length,Petal.Width,color=cluster_k4)) + geom_point() + theme_minima
```



Question 5.1

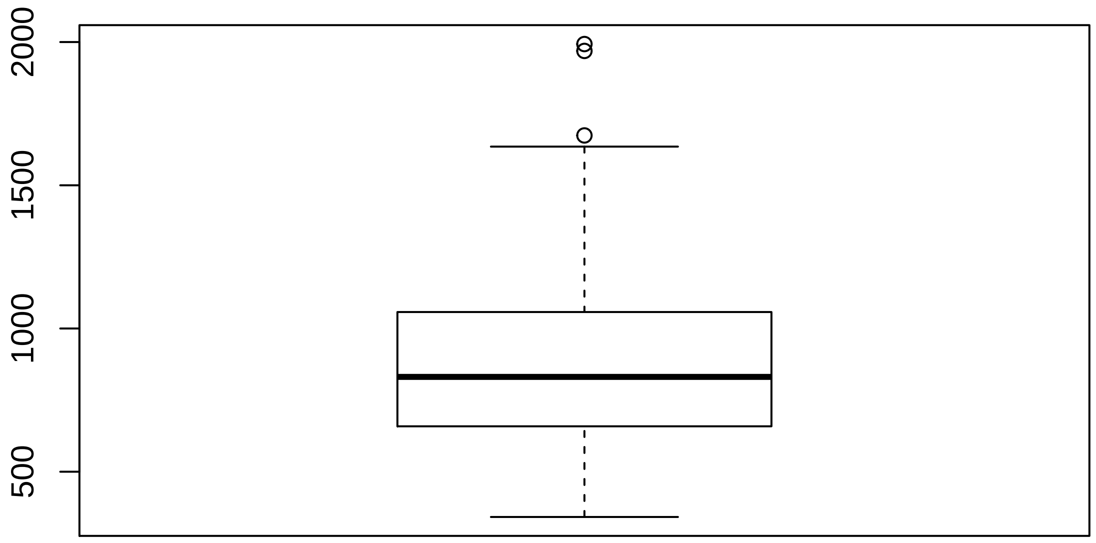
Loading the crime data and displaying a summary of the data to get an understanding of the variables.

```
df_crime <- read.table("5.1uscrimeSummer2018.txt",header=TRUE)
summary(df_crime)
```

```
##        M               So               Ed              Po1
##  Min.   :11.90   Min.   :0.0000   Min.   : 8.70   Min.   : 4.50
##  1st Qu.:13.00   1st Qu.:0.0000   1st Qu.: 9.75   1st Qu.: 6.25
##  Median :13.60   Median :0.0000   Median :10.80   Median : 7.80
##  Mean   :13.86   Mean   :0.3404   Mean   :10.56   Mean   : 8.50
##  3rd Qu.:14.60   3rd Qu.:1.0000   3rd Qu.:11.45   3rd Qu.:10.45
##  Max.   :17.70   Max.   :1.0000   Max.   :12.20   Max.   :16.60
##       Po2               LF              M.F              Pop
##  Min.   : 4.100   Min.   :0.4800   Min.   : 93.40   Min.   :  3.00
##  1st Qu.: 5.850   1st Qu.:0.5305   1st Qu.: 96.45   1st Qu.: 10.00
##  Median : 7.300   Median :0.5600   Median : 97.70   Median : 25.00
##  Mean   : 8.023   Mean   :0.5612   Mean   : 98.30   Mean   : 36.62
##  3rd Qu.: 9.700   3rd Qu.:0.5930   3rd Qu.: 99.20   3rd Qu.: 41.50
##  Max.   :15.700   Max.   :0.6410   Max.   :107.10   Max.   :168.00
##        NW               U1                U2            Wealth
##  Min.   : 0.20   Min.   :0.07000   Min.   :2.000   Min.   :2880
##  1st Qu.: 2.40   1st Qu.:0.08050   1st Qu.:2.750   1st Qu.:4595
##  Median : 7.60   Median :0.09200   Median :3.400   Median :5370
##  Mean   :10.11   Mean   :0.09547   Mean   :3.398   Mean   :5254
##  3rd Qu.:13.25   3rd Qu.:0.10400   3rd Qu.:3.850   3rd Qu.:5915
##  Max.   :42.30   Max.   :0.14200   Max.   :5.800   Max.   :6890
##       Ineq            Prob              Time           Crime
##  Min.   :12.60   Min.   :0.00690   Min.   :12.20   Min.   : 342.0
##  1st Qu.:16.55   1st Qu.:0.03270   1st Qu.:21.60   1st Qu.: 658.5
##  Median :17.60   Median :0.04210   Median :25.80   Median : 831.0
##  Mean   :19.40   Mean   :0.04709   Mean   :26.60   Mean   : 905.1
##  3rd Qu.:22.75   3rd Qu.:0.05445   3rd Qu.:30.45   3rd Qu.:1057.5
##  Max.   :27.60   Max.   :0.11980   Max.   :44.00   Max.   :1993.0
```

Looking at the boxplot of the crime data, we can see that all of the outliers in this series are on the one side (in the higher value range) . A one-sided test for outliers would therefore be better suited to testing for outliers.

```r
boxplot(df_crime$Crime)
```



Performing a one-sided Grubbs test for one outlier results in a p-value of 0.07887. With a confidence level of 0.95, we fail to reject the null hypothesis and thus, there are no outliers in the number of crimes per 100,000 people.

```r
library("outliers")

# one sided grubbs test
grubbs.test(df_crime$Crime,type=10)
```

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

Question 6.1

One situation where a CUSUM model would be appropriate in my every day life would be in tracking whether I have lost strength at the gym or not when I am trying to maintain a certain level of strength. In this case, on any given day I may not be able to lift as much weight and if the weight I am able to lift moves significantly in the downward direction over a number of days, I can be assured that I am indeed losing strength and begin corrective action to counteract.

In this case, I would choose model parameters for the critical value C and threshold T to be related to the standard deviations of the amount of weight I lift. Choosing a C value equal to 2 times the standard deviation is roughly equivalent to only tracking changes with less than 5% chance of occuring by chance under a normal distribution. A T value of 3 times this or equal to 6 times the standard deviation would subjectively seem to be a reasonable warning sign to me that I should make a change in my diet or sleep to counteract the decrease.

Question 6.2.1

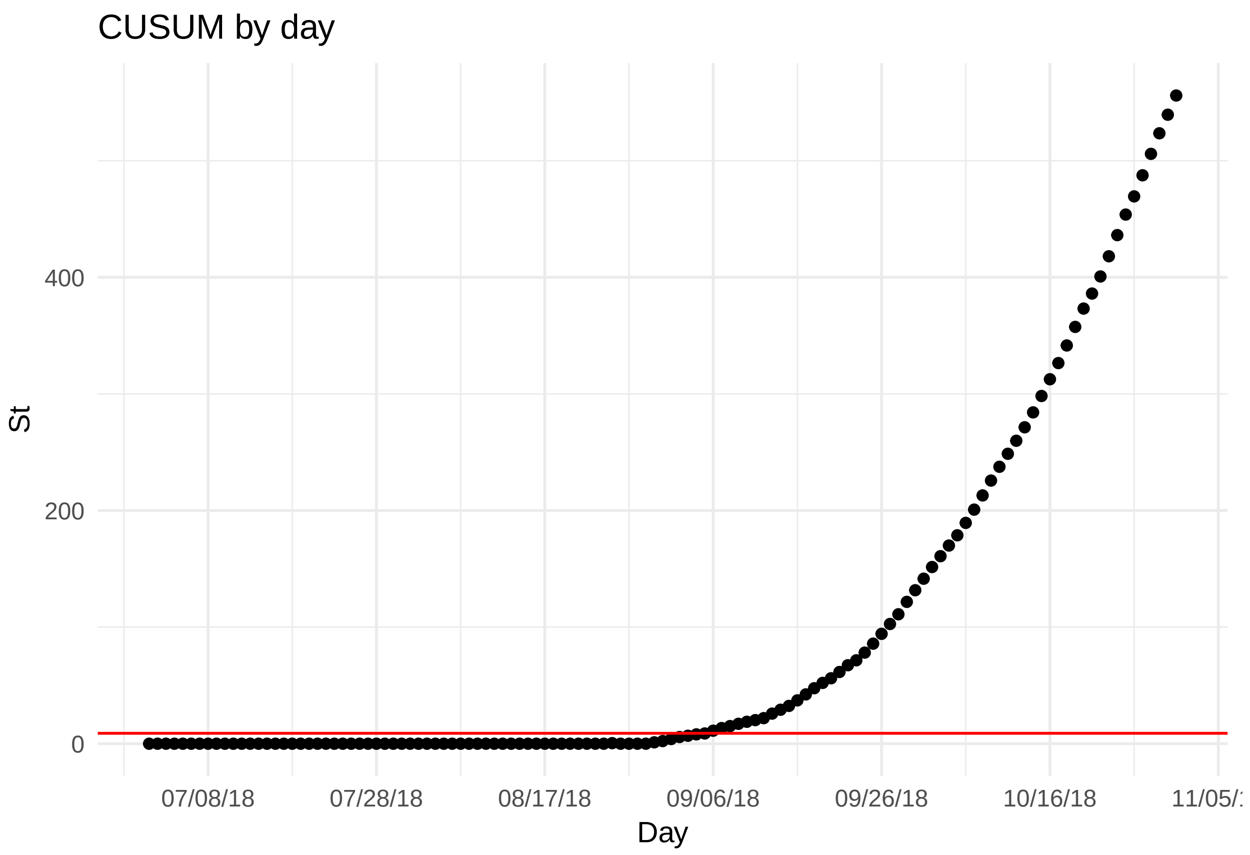
Reading the temperature data and viewing a summary of the data to get an understanding of the variables.

```r
df_temps <- read.table("6.2tempsSummer2018.txt",header=TRUE)

summary(df_temps)
```

```
##       DAY          X1996           X1997           X1998      
##  1-Aug  :  1   Min.   :60.00   Min.   :55.00   Min.   :63.00  
##  1-Jul  :  1   1st Qu.:79.00   1st Qu.:78.50   1st Qu.:79.50  
##  1-Oct  :  1   Median :84.00   Median :84.00   Median :86.00  
##  1-Sep  :  1   Mean   :83.72   Mean   :81.67   Mean   :84.26  
##  10-Aug :  1   3rd Qu.:90.00   3rd Qu.:88.50   3rd Qu.:89.00  
##  10-Jul :  1   Max.   :99.00   Max.   :95.00   Max.   :95.00  
##  (Other):117                                                  
##      X1999           X2000            X2001           X2002      
##  Min.   :57.00   Min.   : 55.00   Min.   :51.00   Min.   :57.00  
##  1st Qu.:75.00   1st Qu.: 77.00   1st Qu.:78.00   1st Qu.:78.00  
##  Median :86.00   Median : 86.00   Median :84.00   Median :87.00  
##  Mean   :83.36   Mean   : 84.03   Mean   :81.55   Mean   :83.59  
##  3rd Qu.:91.00   3rd Qu.: 91.00   3rd Qu.:87.00   3rd Qu.:91.00  
##  Max.   :99.00   Max.   :101.00   Max.   :93.00   Max.   :97.00  
##                                                                  
##      X2003           X2004           X2005           X2006      
##  Min.   :57.00   Min.   :62.00   Min.   :54.00   Min.   :53.00  
##  1st Qu.:78.00   1st Qu.:78.00   1st Qu.:81.50   1st Qu.:79.00  
##  Median :84.00   Median :82.00   Median :85.00   Median :85.00  
##  Mean   :81.48   Mean   :81.76   Mean   :83.36   Mean   :83.05  
##  3rd Qu.:87.00   3rd Qu.:87.00   3rd Qu.:88.00   3rd Qu.:91.00  
##  Max.   :91.00   Max.   :95.00   Max.   :94.00   Max.   :98.00  
##                                                                 
##      X2007           X2008           X2009           X2010      
##  Min.   : 59.0   Min.   :50.00   Min.   :51.00   Min.   :67.00  
##  1st Qu.: 81.0   1st Qu.:79.50   1st Qu.:75.00   1st Qu.:82.00  
##  Median : 86.0   Median :85.00   Median :83.00   Median :90.00  
##  Mean   : 85.4   Mean   :82.51   Mean   :80.99   Mean   :87.21  
##  3rd Qu.: 89.5   3rd Qu.:88.50   3rd Qu.:88.00   3rd Qu.:93.00  
##  Max.   :104.0   Max.   :95.00   Max.   :95.00   Max.   :97.00  
##                                                                 
##      X2011           X2012            X2013           X2014      
##  Min.   :59.00   Min.   : 56.00   Min.   :56.00   Min.   :63.00  
##  1st Qu.:79.00   1st Qu.: 79.50   1st Qu.:77.00   1st Qu.:81.50  
##  Median :89.00   Median : 85.00   Median :84.00   Median :86.00  
##  Mean   :85.28   Mean   : 84.65   Mean   :81.67   Mean   :83.94
##  3rd Qu.:94.00   3rd Qu.: 90.50   3rd Qu.:88.00   3rd Qu.:89.00
##  Max.   :99.00   Max.   :105.00   Max.   :92.00   Max.   :95.00
##
##      X2015
##  Min.   :56.0
##  1st Qu.:77.0
##  Median :85.0
##  Mean   :83.3
##  3rd Qu.:90.0
##  Max.   :97.0
##
```

For this CUSUM model, I used the temperature for each day averaged over all of the years. A critical value C of 2 times the standard deviation of the mean temperatures was used since this would roughly represent a less than 5% chance of occuring by chance assuming normally distributed data. A threshold value T of 10 times the standard deviation was used since this seemed subjectively to me to be a representative enough amount of accumulation to represent a shift in temperatures. A mean value of the first 30 days of temperatures was used as the parameter mu for this CUSUM model. Using this CUSUM model, it would appear that the end of summer is on average around September 6th. This can be seen as the red threshold line in the plot below.

```r
library("ggplot2")
# take average of each day across all years
day_means <- apply(df_temps[,2:21],1,mean)

# data frame to track cusum
df_temp_tracking = data.frame(day = df_temps[,1])
df_temp_tracking$avg_daily = day_means

# St values
St <- rep(0,length(df_temp_tracking[,1]))

# initialize parameters, set C to 2 times the standard deviation and T to five times the standard devia
sd_temp = sd(df_temp_tracking[1:30,2])
cur_s = 0
C = sd_temp * 2
T = sd_temp * 10

mean_temp = mean(df_temp_tracking[1:30,2])
# calculate St values
for (i in seq_along(St)){

  cur_s = max(c(0,cur_s+(mean_temp-df_temp_tracking[i,2]-C)))
  St[i] <- cur_s
}

# store and convert day names to dates
df_temp_tracking$St <- St
df_temp_tracking$day <- as.Date(df_temp_tracking$day, "%d-%b")

# plot with threshold line in red
ggplot(data=df_temp_tracking,aes(day,St)) + geom_point() +scale_x_date(name = 'Day', date_breaks = '20 d
        date_labels = '%D') + geom_hline(aes(yintercept=T),color='red') + labs(title="CUSUM by day") + t
```



```r
# determine first day where threshold passed
thresh_passed = St - T > 0
last_day_of_summer <- df_temp_tracking$day[which(thresh_passed)[1]]
```

```r
print(c("Last day of summer:", format(last_day_of_summer,format='%b %d')))
```

```
## [1] "Last day of summer:" "Sep 06"
```

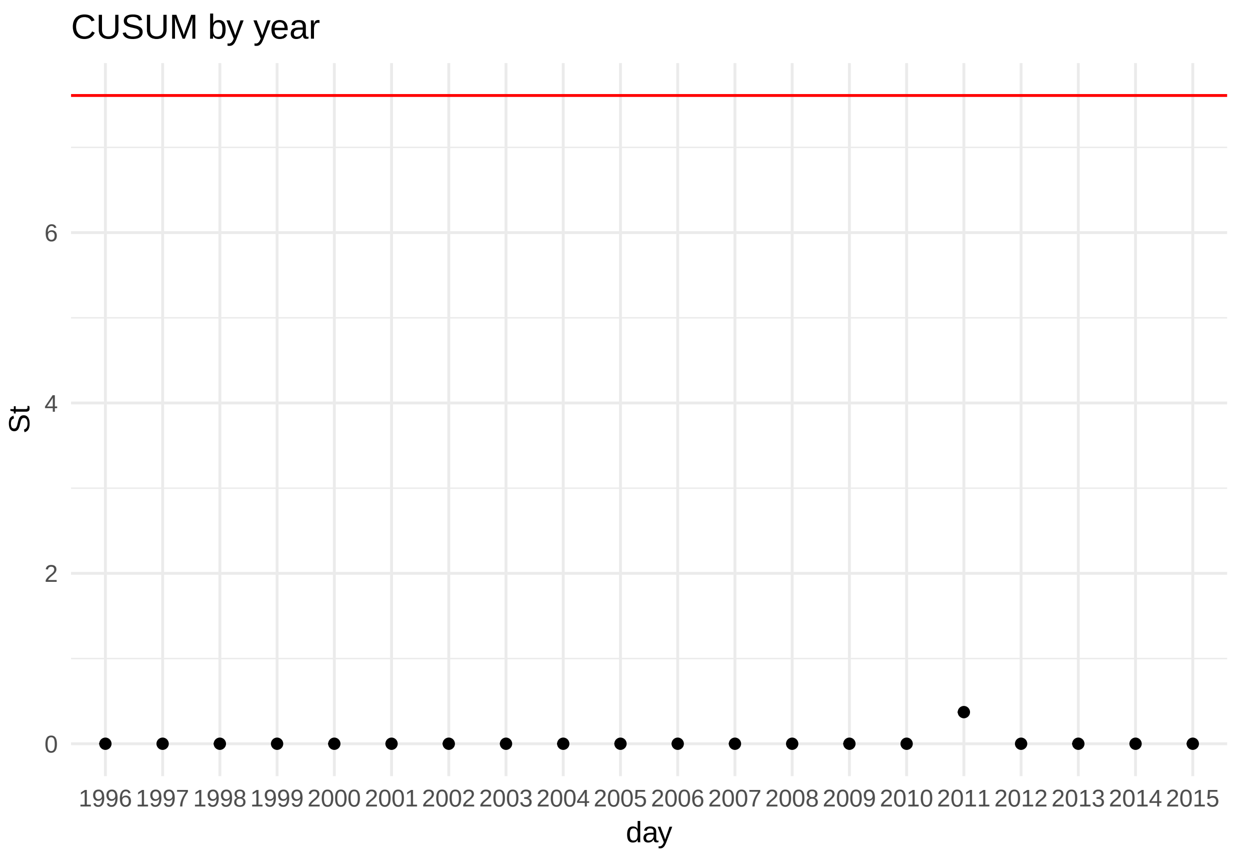
Question 6.2.2

Determining the yearly mean temperatures in the summer using only the days up to and including the end of summer date calculated above:

```r
# get last day of summer
last_day_index = which(thresh_passed)[1]

# get mean for each year's summer using
df_summer <- apply(df_temps[1:last_day_index,2:21],2,mean)
```

Using a CUSUM model with the first five years' average summer temperature to track as the mean parameter mu, a C value of 2 times the standard deviation of the first five years' temperatures and a threshold of 5 times the standard deviation of the first five years' temperatures, it does not appear as though the summers have gotten warmer over time. The first five years were used in these standard deviation and mean calculations since these seemed subjectively to be representative of the early summer temperatures. During 2011, there was an uncommonly warm summer that resulted in an increase to the cusum, but otherwise all mean summer temperatures appear to be within two standard deviations of the mean of the first five summers' temperatures.

```r
year_names <- names(df_summer)

# average of each day
day_means <- apply(df_temps[,2:21],1,mean)

# function to trim X from year name
convert_year <- function(x) {substr(x,2,5)}
# data frame to track cusum
df_temp_tracking_summer = data.frame(day = convert_year(names(df_temps)[2:length(df_temps)]))

df_temp_tracking_summer$temps = unname(df_summer)

# St values
St <- rep(0,length(df_temp_tracking_summer[,1]))

# initialize parameters
sd_temp = sd(df_temp_tracking_summer[1:5,2])
cur_s = 0
C = 2 * sd_temp
T = 5 * sd_temp


# use first five year's mean temperature as mean to compare against
mu = mean(df_temp_tracking_summer[1:5,2])

# iterate and calculate St values
for (i in seq_along(St)){

  cur_s = max(c(0,cur_s+(df_temp_tracking_summer[i,2]-mu-C)))
  St[i] <- cur_s
}

# store St values
df_temp_tracking_summer$St <- St

# plot with threshold line in red
ggplot(data=df_temp_tracking_summer,aes(day,St)) + geom_point() + geom_hline(aes(yintercept=T),color='re
```



```
# determine first day where threshold passed if any (results in NA since does not occur)
thresh_passed = St - T > 0
year_of_change <- df_temp_tracking_summer$day[which(thresh_passed)[1]]
```