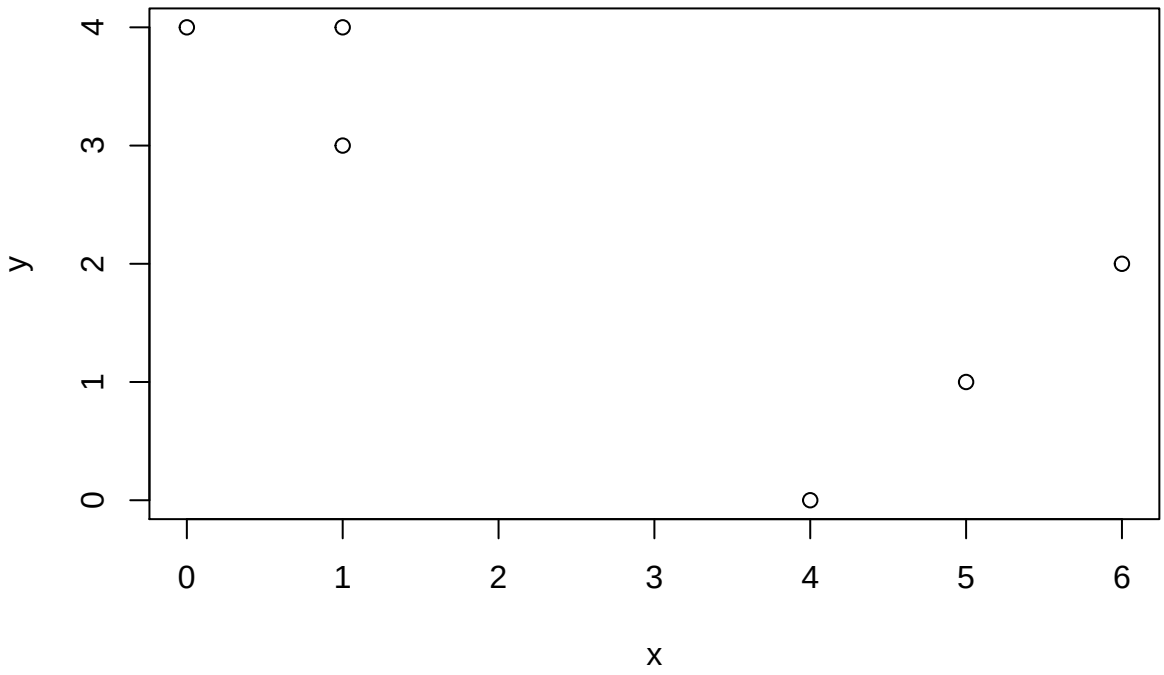
Machine Learning PSET 4

Krish Suchak

Performing k-Means by Hand

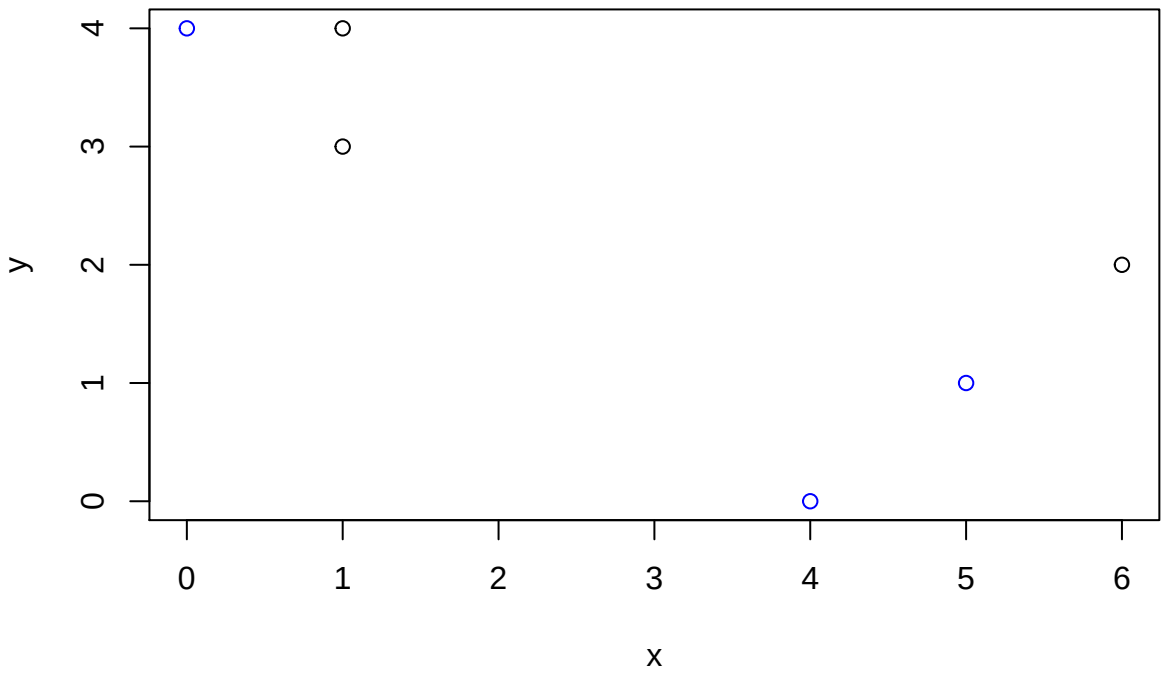
1. Plot the Observations

```
p <- cbind(c(1, 1, 0, 5, 6, 4), c(4, 3, 4, 1, 2, 0))
y <- p[, 1]
x <- p[, 2]
plot(y, x, xlab='x', ylab='y')
```



2. Randomly Assign Cluster Labels to each Obs

```
set.seed(45)
cluster <- sample(seq(0,1), size=6, replace=TRUE)
x_new <- cbind(x, cluster)
y_new <- cbind(y, cluster)
plot(y, x, col=rgb(0, 0, cluster), xlab='x', ylab='y')
```



3. Compute the Centroid for each Cluster

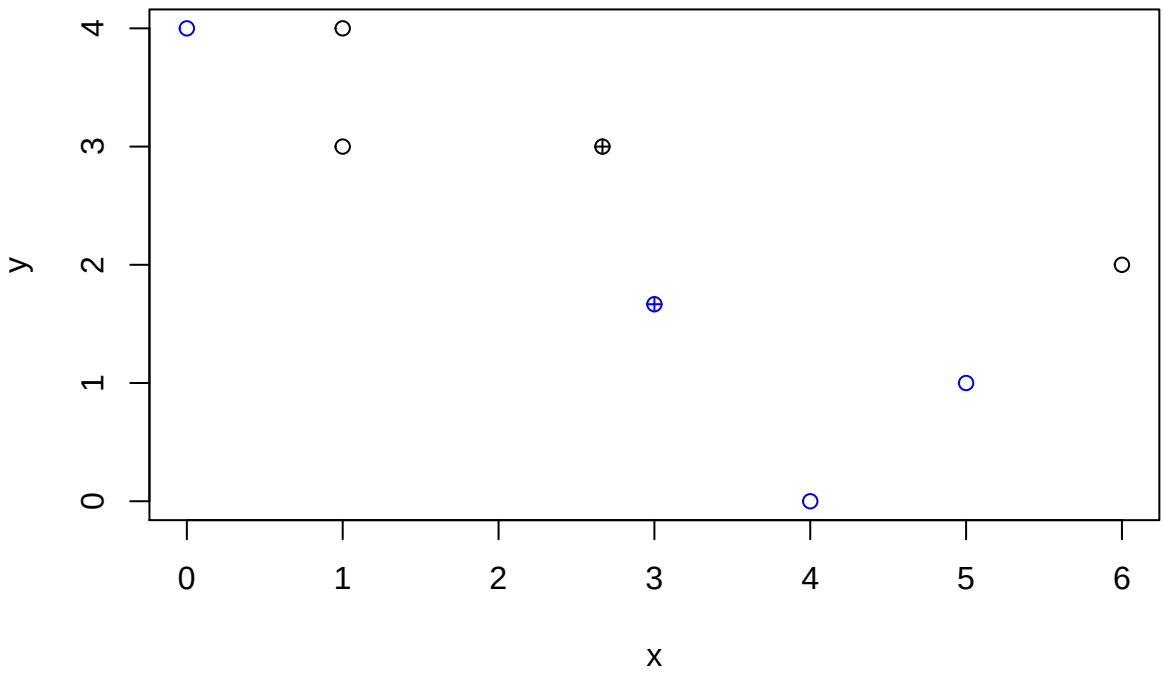
```
    x0 <- mean(y[cluster == 0])
    y0 <- mean(x[cluster == 0])

    x1 <- mean(y[cluster == 1])
    y1 <- mean(x[cluster == 1])

    # centroid for first cluster
    c0 <- c(x0, y0)
    c0
```

```
## [1] 2.666667 3.000000
```

```
    # centroid for second cluster
    c1 <- c(x1, y1)
    c1
```

```
## [1] 3.000000 1.666667
```

```
    plot(y, x, col=rgb(0, 0, cluster), xlab='x', ylab='y')
    points(x0, y0, col="black", pch = 10)
    points(x1, y1, col="blue", pch = 10)
```



4. Assign Closest Centroid for each Obs

```
    distance <- function(x, y) sqrt(sum((x - y) ^ 2))
    out <- NULL
    for (i in 1:nrow(p)) {
      out[i] <- if (distance(p[i, ], c0) < distance(p[i, ], c1)) 0 else 1
    }

    assignment <- cbind(p, out)
    assignment
```

```
##          out
## [1,] 1 4   0
## [2,] 1 3   0
## [3,] 0 4   0
## [4,] 5 1   1
## [5,] 6 2   1
## [6,] 4 0   1
```

5. Repeat Obs Relabeling

```
    x0 <- mean(assignment[,1][assignment[,3]==0])
    y0 <- mean(assignment[,2][assignment[,3]==0])
    x1 <- mean(assignment[,1][assignment[,3]==1])
```

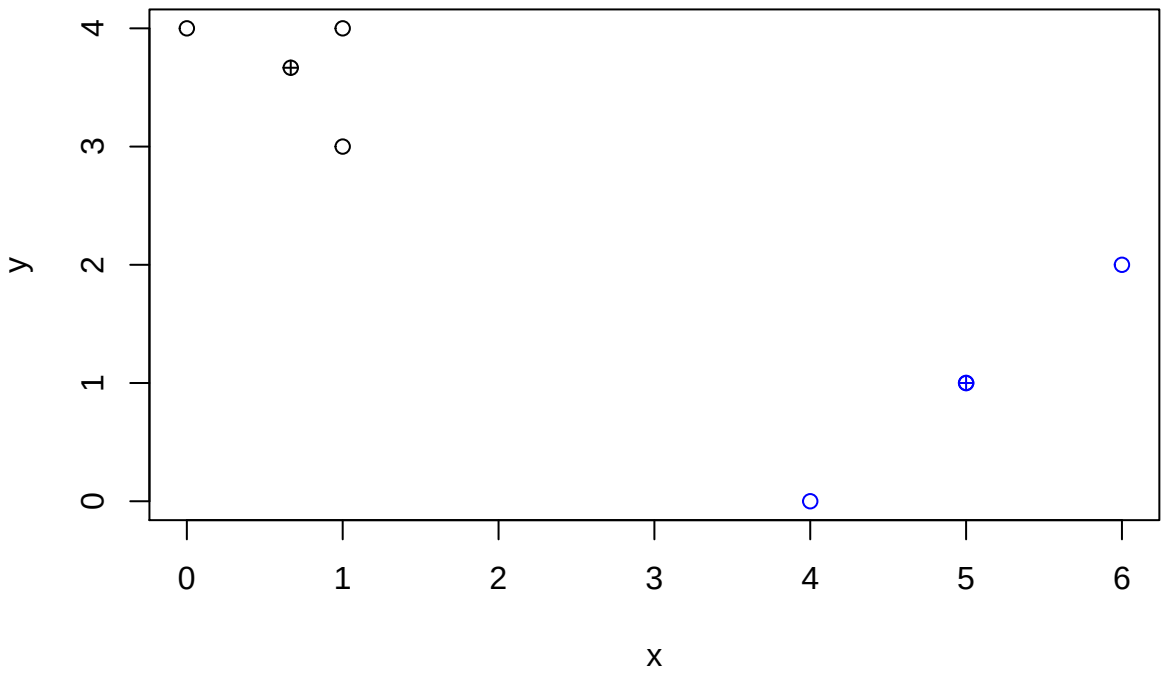
```
    y1 <- mean(assignment[,2][assignment[,3]==1])

    out <- NULL
    c0 <- c(x0,y0)
    c1 <- c(x1,y1)
    for (i in 1:nrow(p)) {
      out[i] <- if (distance(p[i, ], c0) < distance(p[i, ], c1)) 0 else 1
    }
    assignment <- cbind(p, out)
    assignment
```

```
##          out
## [1,] 1 4   0
## [2,] 1 3   0
## [3,] 0 4   0
## [4,] 5 1   1
## [5,] 6 2   1
## [6,] 4 0   1
```

The cluster centroids have stabilized.

6. Replot the Obs by Cluster

```
    plot(y, x, col=rgb(0, 0, out), xlab="x", ylab="y")
    points(x0, y0, col="black", pch = 10)
    points(x1, y1, col="blue", pch = 10)
```



Clustering State Legislative Professionalism

1. Load the Data

```
load("Data and Codebook//legprof-components.v1.0.RData")
data <- x
```

2. Munge the Data

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
mod <- na.omit(data[data$year == "2009" | data$year == "2010",]) #b, #c
munged <- mod %>%
  select(t_slength, slength, salary_real, expend) %>% #a
```

```
    scale() #d
  rownames(munged) <- mod$state #e, associated state names
```

3. Get Clusterability

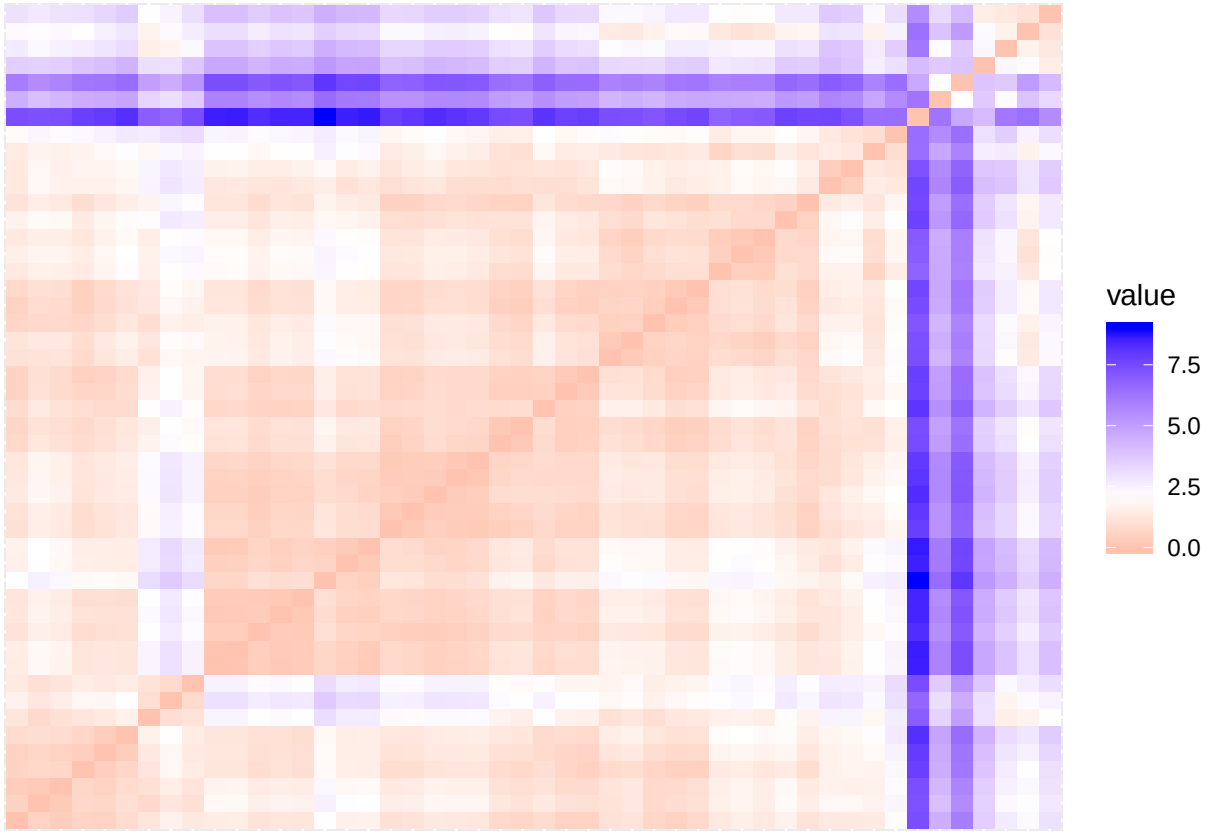
```
  library(factoextra)
```

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

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

```
  get_clust_tendency(munged, 45)
```

```
## $hopkins_stat
## [1] 0.8226008
##
## $plot
```

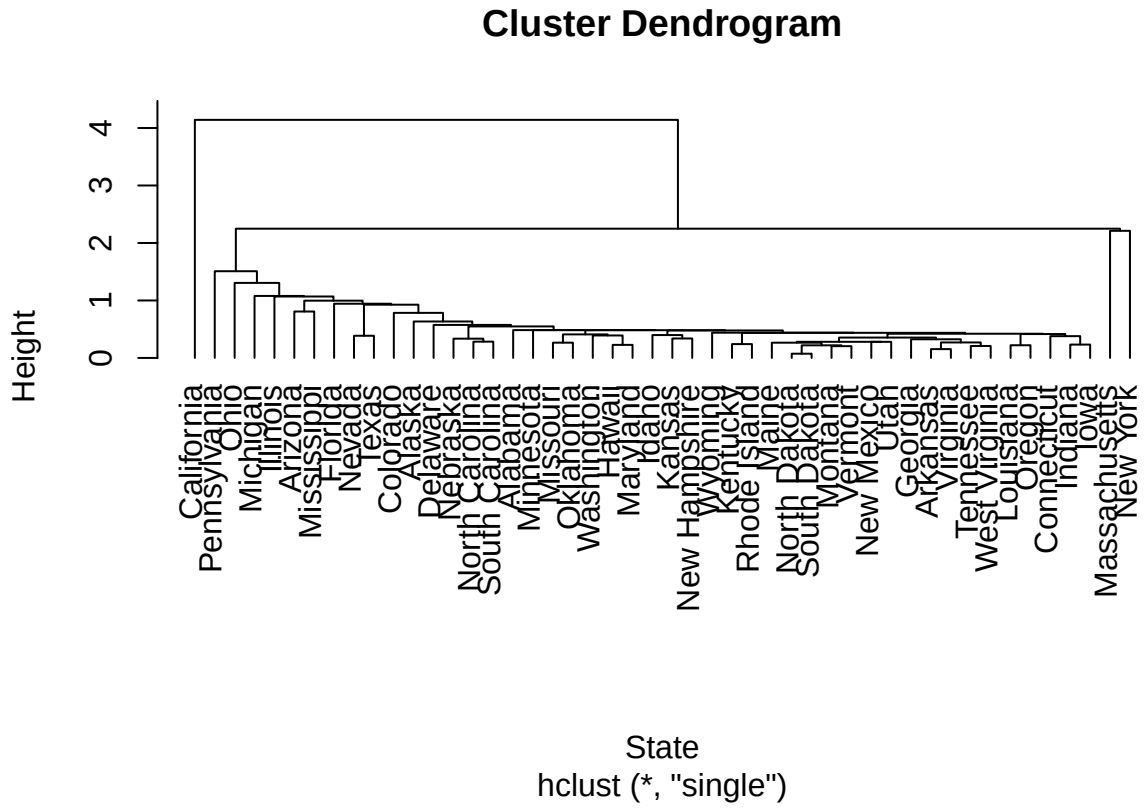


Both the contiguous red and white squares along with the relatively high Hopkins stat of 0.82 suggest high likelihood that this dataset is clusterable.

4. Fit an Agglomerative Hierarchical Clustering Algorithm

```
  set.seed(45)
  hc <- hclust(stats::dist(munged), method = "single")
```

```
    plot(hc, hang = -1, xlab = "State")
```



After fitting a hierarchical clustering model with single linkage, we note that California and New York are similar in their dissimilarity from the other clusters. The Dakotas and Carolinas are extremely similar, respectively. However, we find it difficult to explain the other geographical patterns that emerge among the clusters (the fact that New Hampshire appears to be most similar to Kansas).

5. Fit a k-means Algorithm

```
    set.seed(45)
    model <- kmeans(munged, centers = 2, nstart = 25)
    t <- as.table(model$cluster)
    t <- data.frame(t)
    t[t$Freq == "1",]
```

```
##             Var1 Freq
## 5     California    1
## 21 Massachusetts    1
## 22      Michigan    1
## 31      New York    1
## 34          Ohio    1
## 37  Pennsylvania    1
```

```
    model$centers
```

```
##    t_slength    slength salary_real     expend
```

```
## 1  2.0079549  2.0643454     2.04323  1.4647791
## 2 -0.2868507 -0.2949065    -0.29189 -0.2092542
```

```
    model$size
```

```
## [1]  6 42
```

6 states (California, Massachusetts, Michigan, New York, Ohio, and Pennsylvania) are in Cluster 1. The rest of the states fall in Cluster 2.These states have a much higher level of "professionalism" (that is, a greater session length, salary, and expenditure amount) than do their counterparts (Cluster 2 states).

6. Fit a Gaussian Mixture Model

```
    library(mixtools)
```

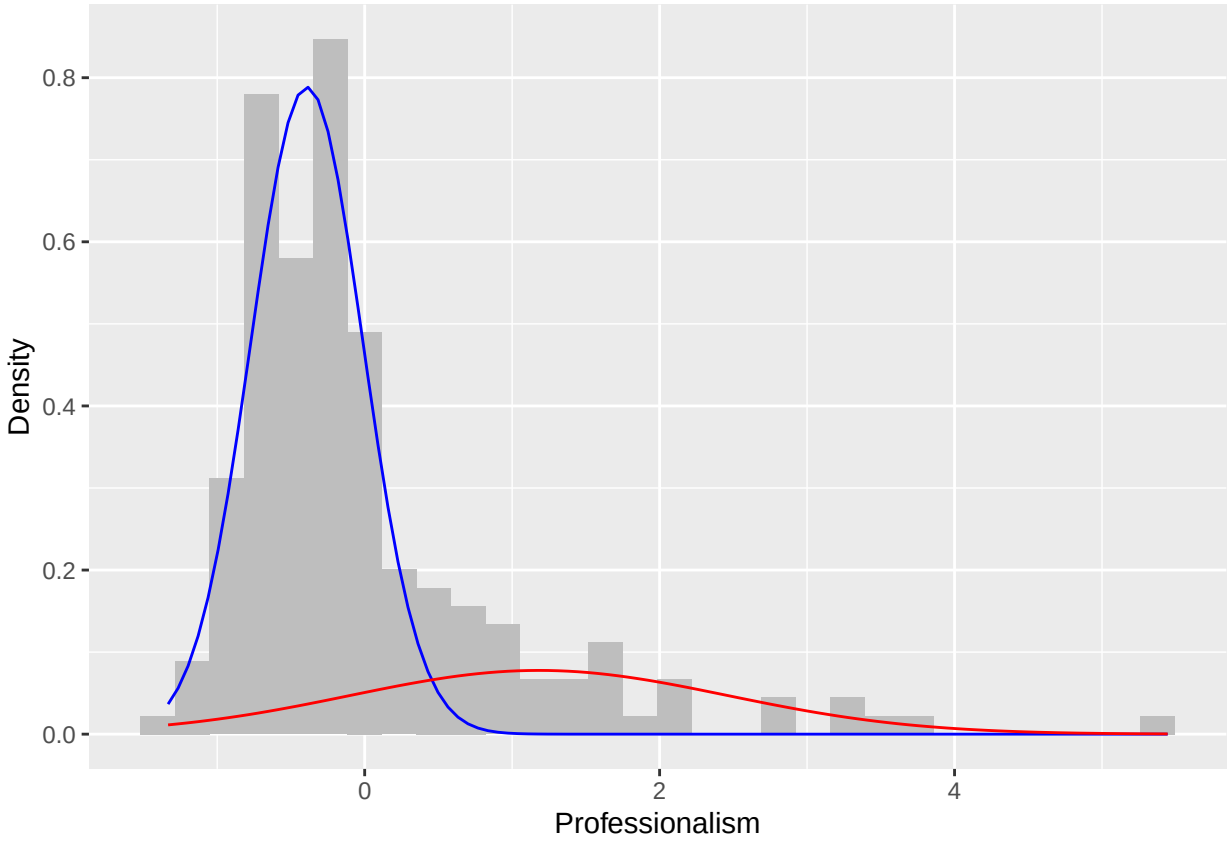
```
## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

```
    library(plotGMM)
    set.seed(50)
    gmm <- normalmixEM(munged, k = 2)
```

```
## number of iterations= 33
```

```
    ggplot(data.frame(x = gmm$x)) +
      geom_histogram(aes(x, ..density..), fill = "gray") +
      stat_function(geom = "line", fun = plot_mix_comps,
      args = list(gmm$mu[1], gmm$sigma[1], lam = gmm$lambda[1]),
      colour = "blue") +
      stat_function(geom = "line", fun = plot_mix_comps,
      args = list(gmm$mu[2], gmm$sigma[2], lam = gmm$lambda[2]),
      colour = "red") +
      xlab("Professionalism") +
      ylab("Density")
```

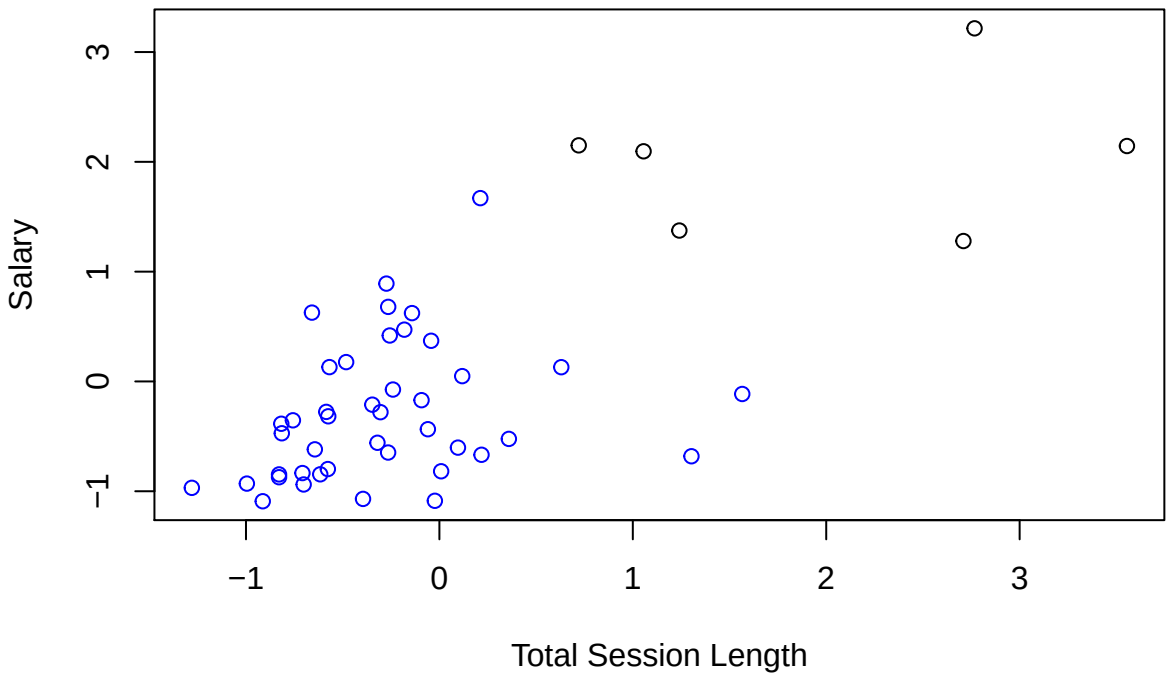
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



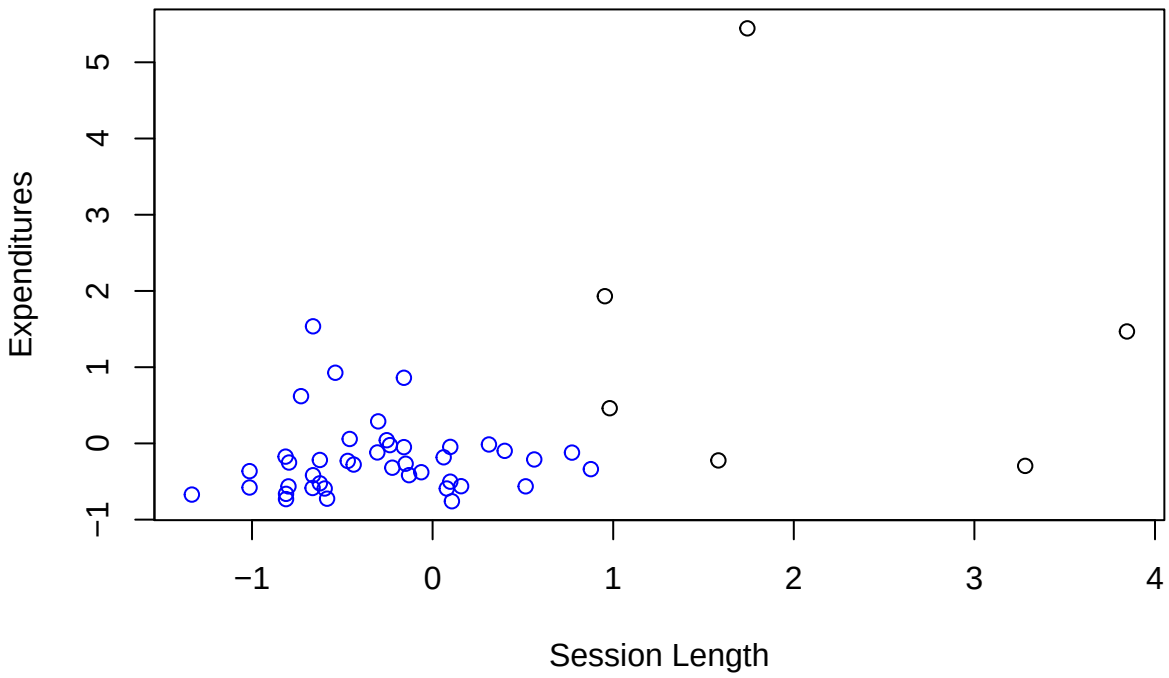
From the figure above, we see a large number of states in Cluster 2, with a professionalism value centroid below 0. Conversely in Cluster 1, we have a fewer number of states and a much more spread out distribution of professionalism values.

7. Compare Plots 2 Feature Combos

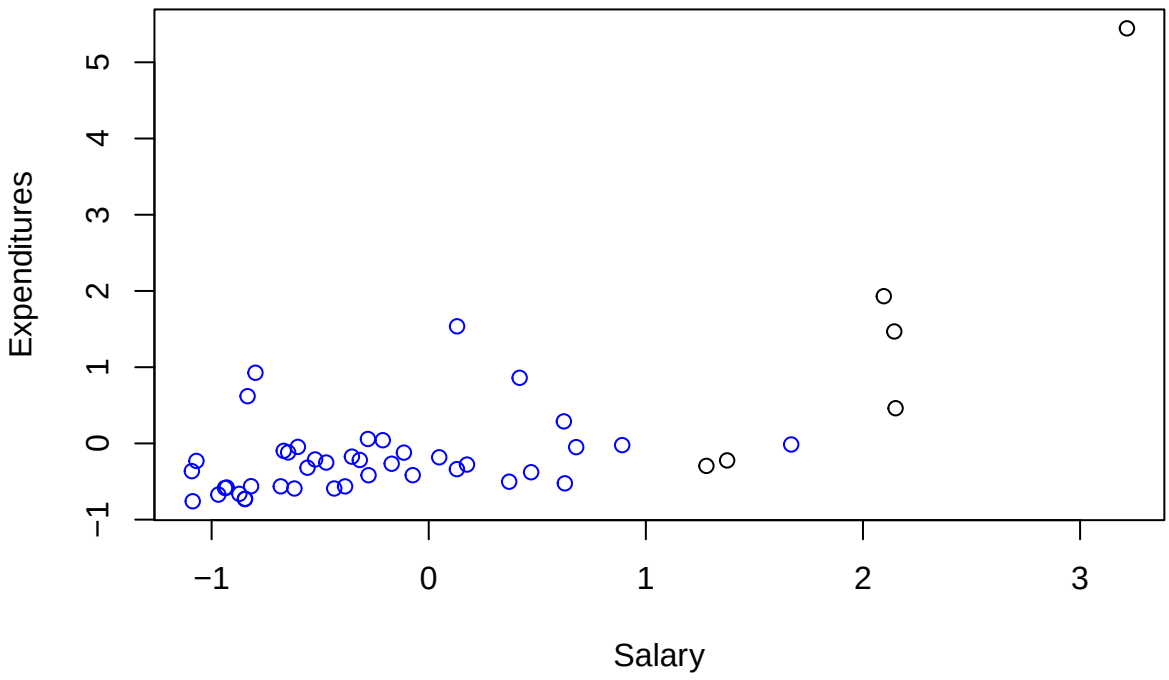
```
rownames(t) <- mod$state
colnames(t)[colnames(t)=="Freq"] <- "Assignment"
t$Var1 <- NULL

merged <- merge(munged,t,by.x = 0, by.y = 0)
merged["Assignment"] = merged["Assignment"]-1
plot(merged[,2],merged[,4],col=rgb(0,0,merged["Assignment"]),
xlab="Total Session Length",ylab="Salary")
```




```
plot(merged[,3],merged[,5],col=rgb(0,0,merged[,"Assignment"]),
xlab="Session Length",ylab="Expenditures")
```




```
plot(merged[,4],merged[,5],col=rgb(0,0,merged[,"Assignment"]),
xlab="Salary",ylab="Expenditures")
```



From the scatter plots, we can visually verify that the k-means model clustered state legislatures well based on a variety of features.

8. Select a Validation Strategy

```
    library(clValid)
```

```
## Loading required package: cluster
```

```
    rownames(munged) <- 1:nrow(munged)
    valid <- clValid(munged, c(2), clMethods = c("hierarchical", "kmeans","model"),
    validation = c("internal"))
```

```
## Loading required package: mclust
```

```
## Package 'mclust' version 5.4.5
## Type 'citation("mclust")' for citing this R package in publications.
```

```
##
## Attaching package: 'mclust'
```

```
## The following object is masked from 'package:mixtools':
##
##     dmvnorm
```

```
    summary(valid)
```

```
##
```

```
## Clustering Methods:
##  hierarchical kmeans model
##
## Cluster sizes:
##  2
##
## Validation Measures:
##                                  2
##
## hierarchical Connectivity   6.0869
##              Dunn           0.3598
##              Silhouette     0.6920
## kmeans       Connectivity   8.5683
##              Dunn           0.1726
##              Silhouette     0.6390
## model        Connectivity  18.7095
##              Dunn           0.0833
##              Silhouette     0.4230
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 6.0869 hierarchical 2
## Dunn         0.3598 hierarchical 2
## Silhouette   0.6920 hierarchical 2
```

9. Discuss the Validation Output

So, for the hierarchical, k-means, and gmm models with k = 2, we got the following silhouette scores: 0.69, 0.64, 0.42. From these results, it seems that the hierarchical model performs best, followed by the k-means model. By iterating through multiple k values as well, we ensure that the suggestion to use k = 2 in the writeup is a fair suggestion indeed.

There are several reasons why one might select a "sub-optimal" clustering method. For example, if one selects the same amount of clusters as data points, one could fit a model with perfect accuracy on the training set (no bias), but extremely high variance. This model would likely be overfit and perform significantly worse on the test set (by modeling too much random noise in the data). Thus, a model like this would not be the optimal choice.

Another reason for selecting a sub-optimal model is outside knowledge. If one has outside knowledge or reason to suspect that there are a certain number of clusters (the data is already labeled but we unlabeled it for the sake of data exploration and clusterability discovery), then there may be reason to use a sub-optimal k value (the number of unique labels in the original labeled dataset).