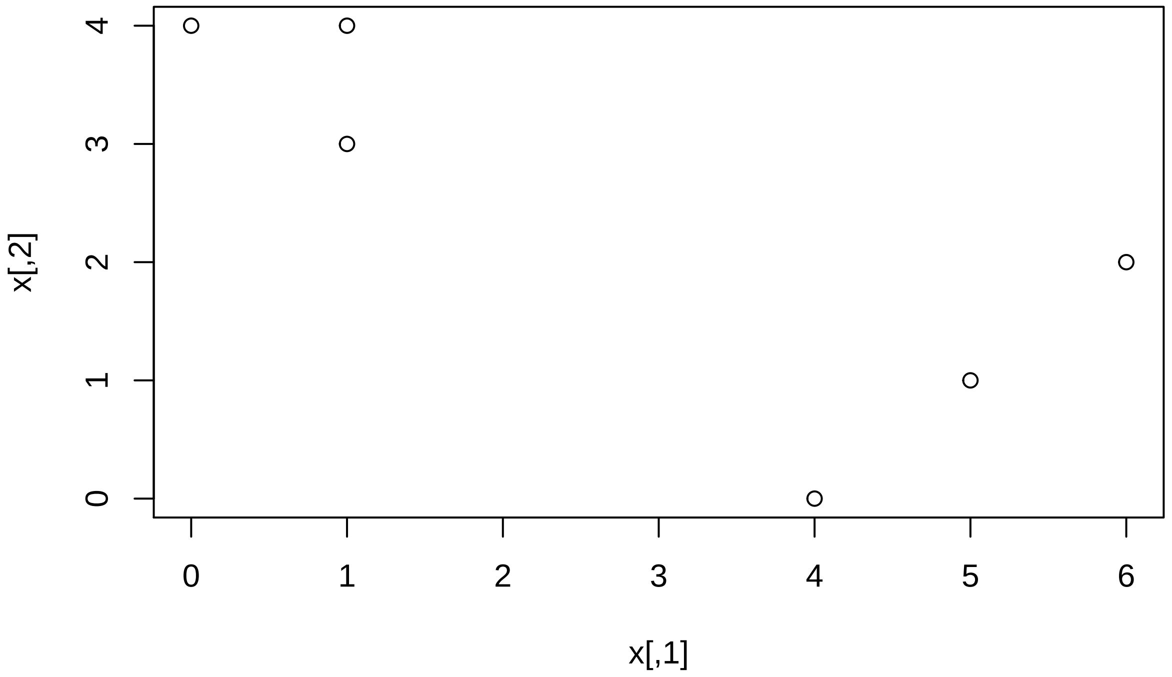
Problem Set 4

Pete Cuppernull

3/2/2020

K-Means by hand

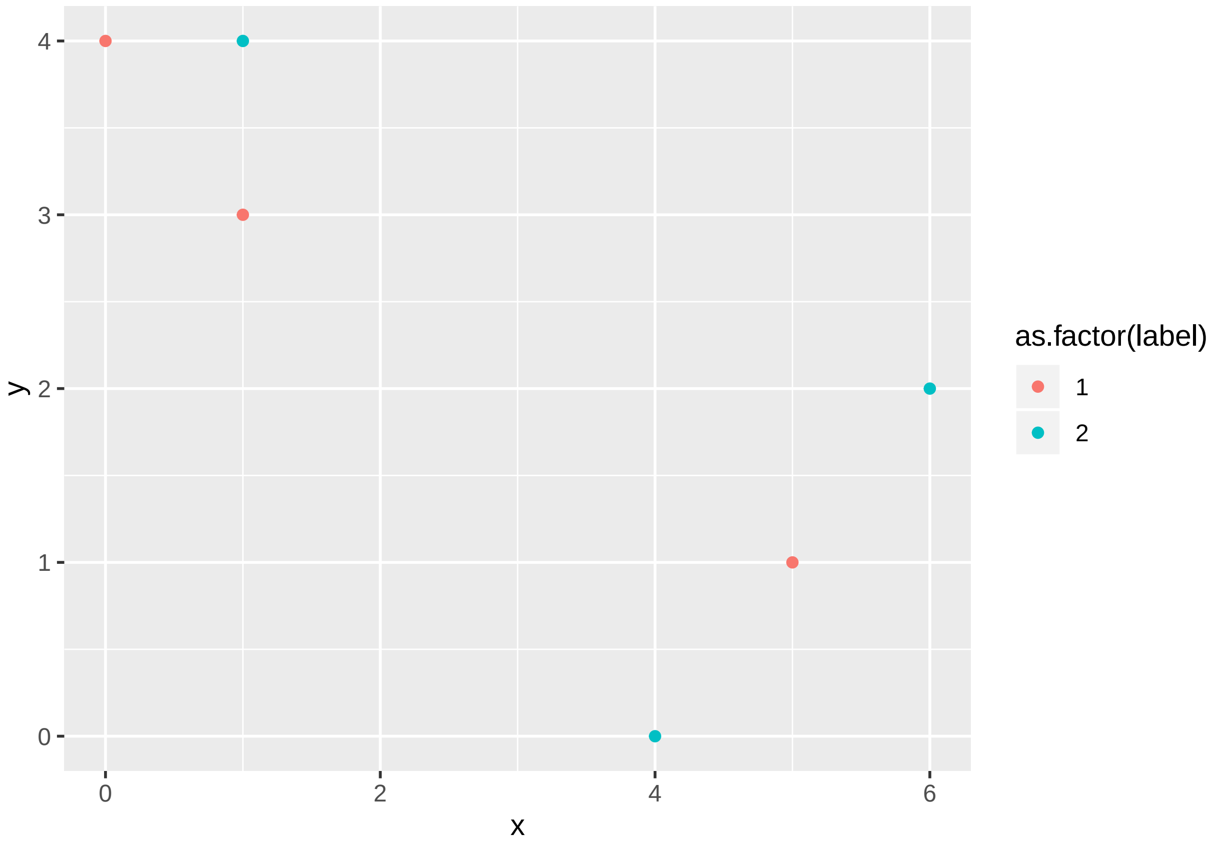
```
set.seed(1414)
x <- cbind(c(1, 1, 0, 5, 6, 4), c(4, 3, 4, 1, 2, 0))


#Plot the observations.
plot(x)
```



```
# Randomly assign a cluster label to each observation.
#Report the cluster labels and plot the results with a different color for each cluster.
colnames(x) <- c("x", "y")
label <- as.factor(sample(1:2, 6, replace=TRUE))
x <- cbind(x, label)

x_df <- as.data.frame(x)

x_df
```

```
##   x y label
## 1 1 4     2
## 2 1 3     1
## 3 0 4     1
## 4 5 1     1
## 5 6 2     2
## 6 4 0     2
```

```
x_df %>%
  ggplot() +
  geom_point(mapping = aes(x, y, color = as.factor(label)))
```



```
#Compute the centroid for each cluster.


centroids <- x_df %>%
  group_by(label) %>%
  mutate(mean(x), mean(y)) %>%
  select(`mean(x)`, `mean(y)`) %>%
  distinct()

centroids
```

```
## # A tibble: 2 x 3
## # Groups:   label [2]
##   label `mean(x)` `mean(y)`
##   <dbl>     <dbl>     <dbl>
## 1     2      3.67      2
## 2     1      2         2.67
```

```r
#Assign each observation to the centroid to which it is closest, in terms of Euclidean distance.
#Report the cluster labels for each observation.
x1 <- centroids[1,2]
x2 <- centroids[2,2]
y1 <- centroids[1,3]
y2 <- centroids[2,3]
xs <- cbind(x1, x2)
ys <- cbind(y1, y2)
points <- cbind(xs, ys)
colnames(points) <- c("x1", "x2", "y1", "y2")

df_cluster <- as.data.frame(cbind(x, points))


df_cluster %>%
  mutate(new_label = if_else(  ((abs(x-x1) + abs(y-y1)) / 2) <
                                 ((abs(x-x2) + abs(y-y2)) / 2), 1, 2)) %>%
  select(x, y, new_label)
```

```
##   x y new_label
## 1 1 4         2
## 2 1 3         2
## 3 0 4         2
## 4 5 1         1
## 5 6 2         1
## 6 4 0         1
```

```r
#Repeat (3) and (4) until the answers/clusters stop changing.

#Write function to iterate
iterate <- function(original_df){
  x_df <- as.data.frame(original_df)

centroids <- x_df %>%
  group_by(label) %>%
  mutate(mean(x), mean(y)) %>%
  select(`mean(x)`, `mean(y)`) %>%
  distinct()

x1 <- centroids[1,2]
x2 <- centroids[2,2]
y1 <- centroids[1,3]
y2 <- centroids[2,3]
xs <- cbind(x1, x2)
ys <- cbind(y1, y2)
points <- cbind(xs, ys)
colnames(points) <- c("x1", "x2", "y1", "y2")

df_cluster <- as.data.frame(cbind(x, points))

new_df <- df_cluster %>%
  mutate(label = if_else(  ((abs(x-x1) + abs(y-y1)) / 2) < ((abs(x-x2) + abs(y-y2)) / 2), 1, 2)) %>%
  select(x, y, label)
```

```
new_df
}

#Now, manually iterate until convergence
##Stage 1
x2 <- iterate(x)
x == x2 #Not converged :(
```

```
##          x    y label
## [1,] TRUE TRUE  TRUE
## [2,] TRUE TRUE FALSE
## [3,] TRUE TRUE FALSE
## [4,] TRUE TRUE  TRUE
## [5,] TRUE TRUE FALSE
## [6,] TRUE TRUE FALSE
```

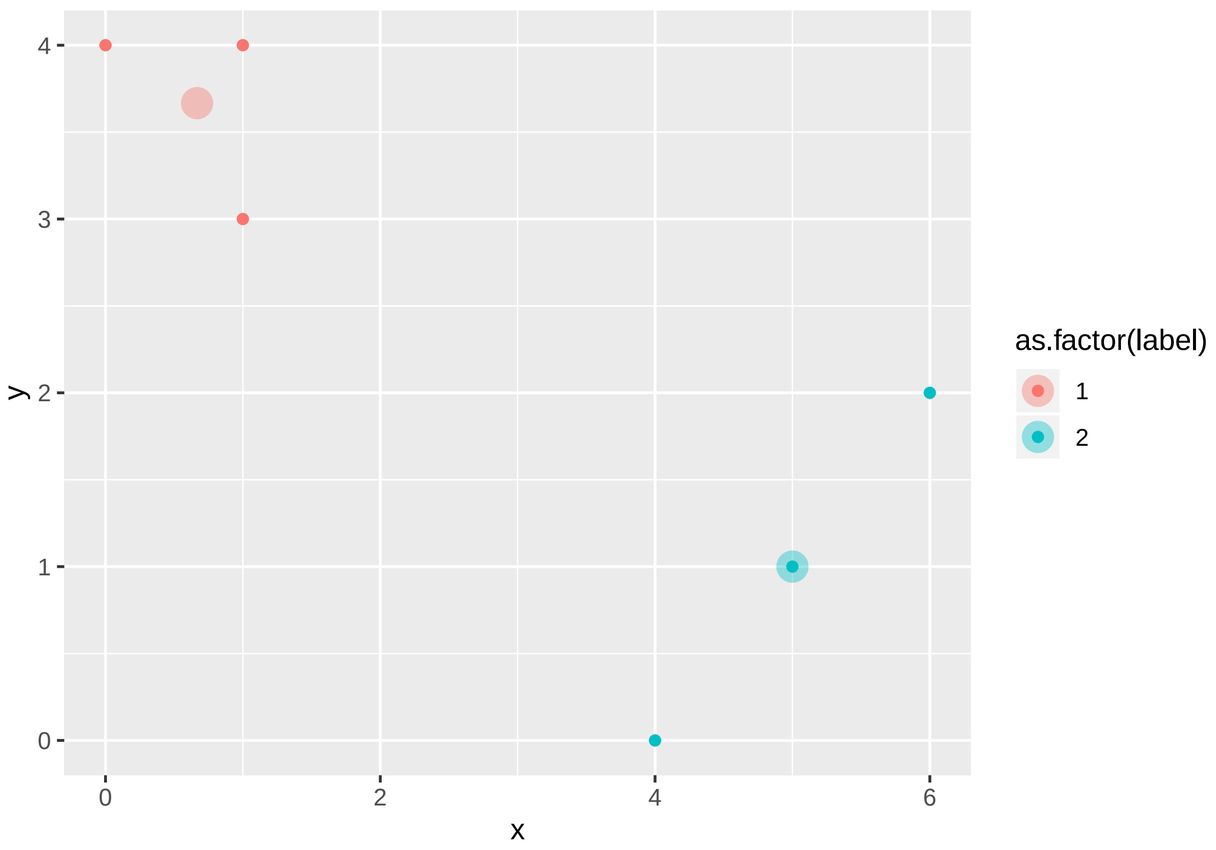
```
x3 <- iterate(x2)

x2 == x3 ## still not converged :(
```

```
##          x    y label
## [1,] TRUE TRUE FALSE
## [2,] TRUE TRUE FALSE
## [3,] TRUE TRUE FALSE
## [4,] TRUE TRUE FALSE
## [5,] TRUE TRUE FALSE
## [6,] TRUE TRUE FALSE
```

```
x4 <- iterate(x3)

x3 == x4 ##converged! :)
```

```
##          x    y label
## [1,] TRUE TRUE  TRUE
## [2,] TRUE TRUE  TRUE
## [3,] TRUE TRUE  TRUE
## [4,] TRUE TRUE  TRUE
## [5,] TRUE TRUE  TRUE
## [6,] TRUE TRUE  TRUE
```

```
#Reproduce the original plot from (1), but this time color the observations according to the clusters
#labels you obtained by iterating the cluster centroid calculation and assignments.

final_centroids <- as.data.frame(x4) %>%
  group_by(label) %>%
  mutate(mean(x), mean(y)) %>%
  select(`mean(x)`, `mean(y)`) %>%
  distinct()

ggplot() +
  geom_point(data = as.data.frame(x4),
             mapping = aes(x, y, color = as.factor(label))) +
  geom_point(data = final_centroids,
             mapping = aes(`mean(x)`, `mean(y)`, color = as.factor(label)),
             size = 5,
             alpha = .4)
```



Clustering State Legislative Professionalism

Munge Data

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

data <- x

##seelct columns, filter for 2010, remove na's, standardize, keep only numerics
predictors <- data %>%
  select(stateabv, t_slength, slength, salary_real, expend, year) %>%
  filter(year == 2010) %>%
  na.omit() %>%
  mutate(t_slength = scale(t_slength),
         slength = scale(slength),
         salary_real = scale(salary_real),
         expend = scale(expend)) %>%
  select(-stateabv, -year)

##save state names separately
states <- data %>%
  select(stateabv, t_slength, slength, salary_real, expend, year) %>%
  filter(year == 2010) %>%
  na.omit() %>%
  select(stateabv)

#another object that I might use later
data_clean <- data %>%
  select(stateabv, t_slength, slength, salary_real, expend, year) %>%
  filter(year == 2010) %>%
  na.omit()  %>%
  select(-stateabv, -year)

#another object that I might use later
data_clean_ahc <- data %>%
  select(stateabv, t_slength, slength, salary_real, expend, year) %>%
  filter(year == 2010) %>%
  na.omit()  %>%
  select(-year)

head(predictors)
```

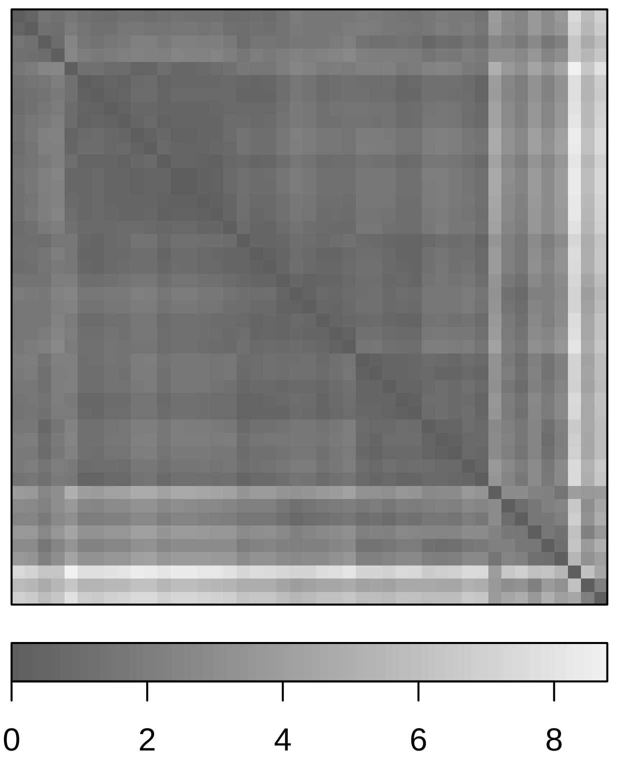
```
##    t_slength    slength salary_real     expend
## 1 -0.3866551 -0.4767609 -1.07156761 -0.2380562
## 2 -0.2493575 -0.1721360  0.37962619  0.8214317
## 3  1.5600771  0.7395113 -0.14005366 -0.1319733
## 4 -0.8035988 -0.7954051 -0.48879929 -0.2585064
## 5  2.7524766  1.6909893  3.10667105  5.2743270
## 6  0.6310228  0.8419701  0.09799177 -0.3427045
```

Assess Clusterability

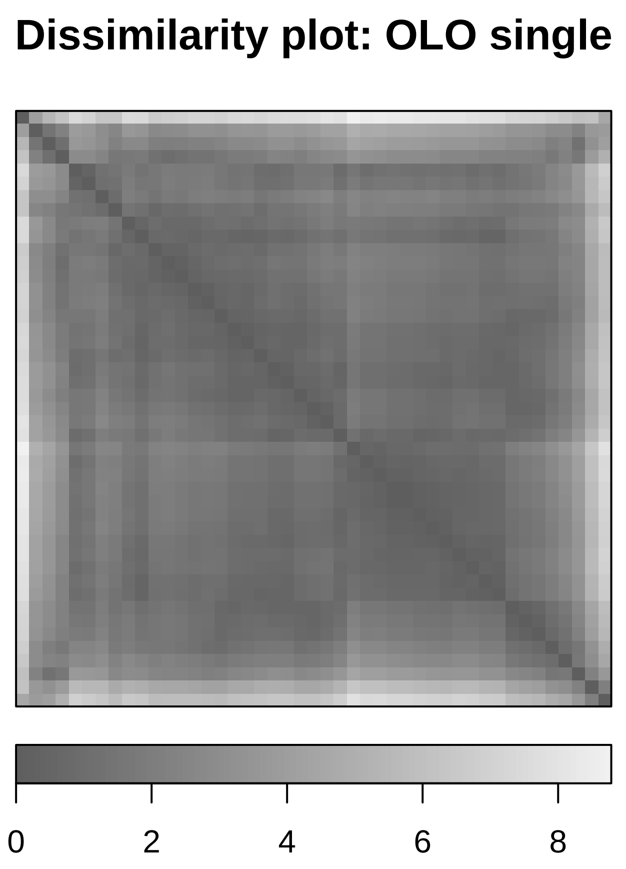
I am going to diagnose clusterability with some dissimilarity plots.

```
predictors2 <- predictors[sample(seq_len(nrow(predictors))),]

#Make distance matrix
predictor_distance <- dist(as.matrix(predictors2), method = "euclidean")

## PLOT MATRICES - I tried several different methods and these seem to be most effective.
#Method 1
dissplot(predictor_distance,
         method = "HC_average",
         options = list(main = "Dissimilarity plot: HC Average"))
```

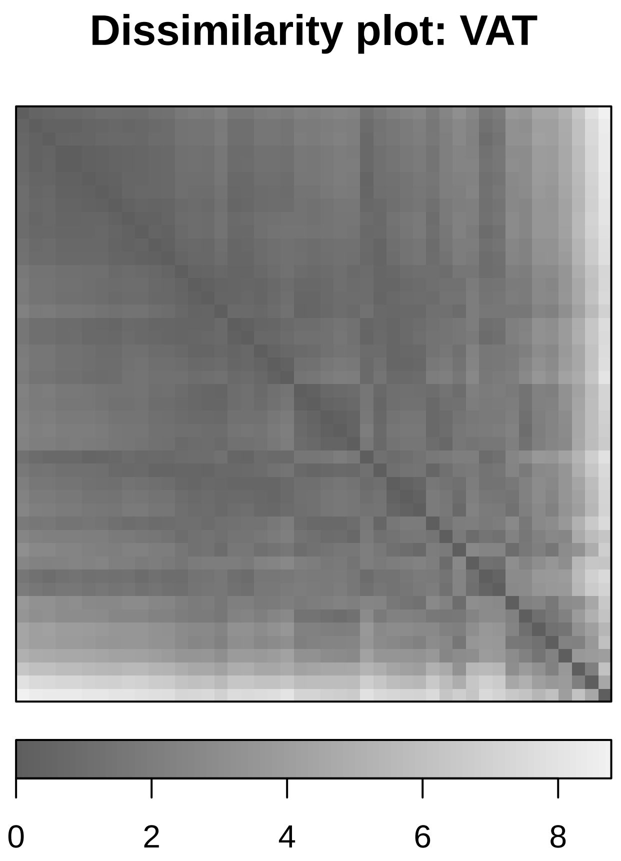
Dissimilarity plot: HC Average



```
#Method 2
dissplot(predictor_distance,
         method = "OLO_single",
         options = list(main = "Dissimilarity plot: OLO single"))
```



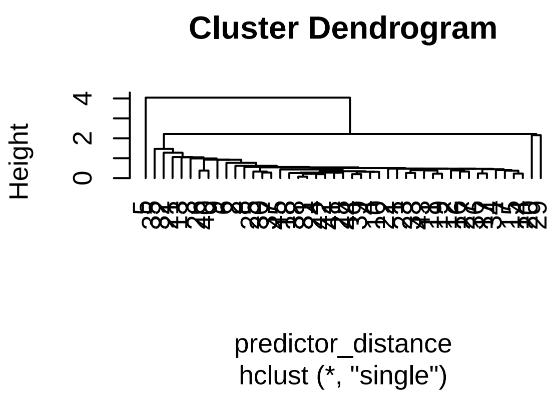
```
#Method 3
dissplot(predictor_distance,
         method = "VAT",
         options = list(main = "Dissimilarity plot: VAT"))
```

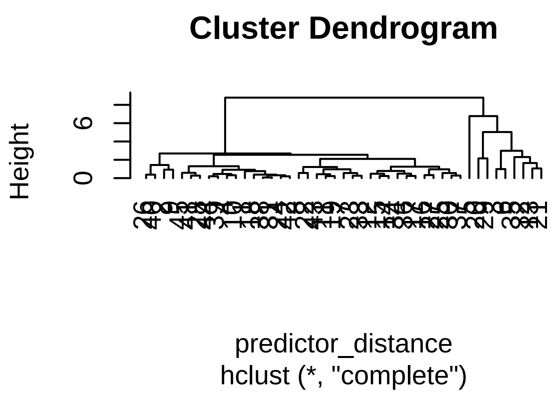


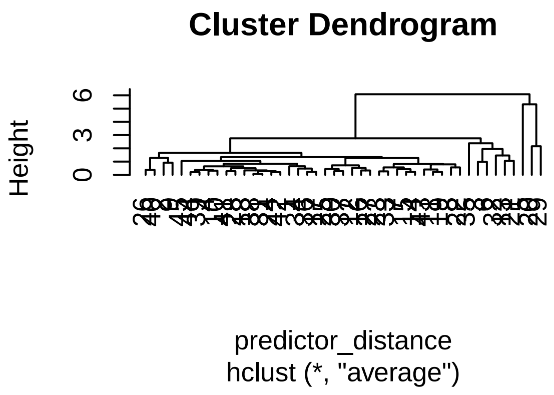
Although a little cloudy, there appears to be stratification and blocking in the dissimilarity plots, indicating a nonrandom structure to the data. While I can't give a numeric likelihood ex ante, I am confident that we can generate intuitive clusters in the data.

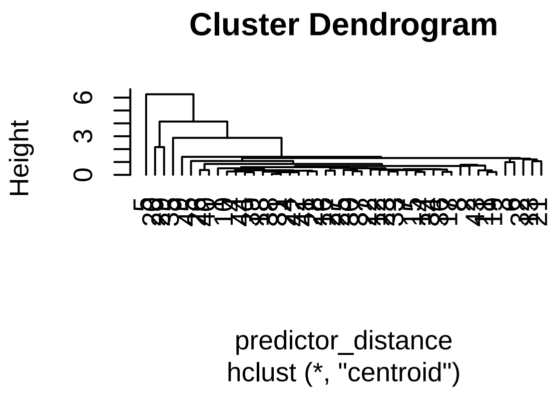
Agglomerative Hierarchical Clustering

```r
#Create trees for different AHC models
par(mfrow = c(2,2))
hc_single <- hclust(predictor_distance,
                    method = "single"); plot(hc_single, hang = -1)

hc_complete <- hclust(predictor_distance,
                      method = "complete"); plot(hc_complete, hang = -1)

hc_average <- hclust(predictor_distance,
                     method = "average"); plot(hc_average, hang = -1)

hc_centroid <- hclust(predictor_distance,
                      method = "centroid"); plot(hc_centroid, hang = -1)
```









```
# reset plot space
par(mfrow = c(1,1))


#There appears to be some different clustering going on for each
##lets cut up the trees and see how many clusters we might have for each method
cuts_single <- cutree(hc_single,
               k = c(2,3))
cuts_comp <- cutree(hc_complete,
               k = c(2,3))
cuts_avg <- cutree(hc_average,
               k = c(2,3))
cuts_cent <- cutree(hc_centroid,
               k = c(2,3))


### make tables
table(`2 Clusters` = cuts_single[,1],
  `3 Clusters` = cuts_single[,2])
```

```
##             3 Clusters
## 2 Clusters  1  2  3
##           1 42  2  0
##           2  0  0  1
```

```
table(`2 Clusters` = cuts_comp[,1],
  `3 Clusters` = cuts_comp[,2])
```

```
##             3 Clusters
## 2 Clusters  1  2  3
##          1 36  0  0
##          2  0  8  1
```

```
table(`2 Clusters` = cuts_avg[,1],
  `3 Clusters` = cuts_avg[,2])
```

```
##           3 Clusters
## 2 Clusters  1  2  3
##          1 42  0  0
##          2  0  2  1
```

```
table(`2 Clusters` = cuts_cent[,1],
  `3 Clusters` = cuts_cent[,2])
```

```
##           3 Clusters
## 2 Clusters  1  2  3
##          1 42  2  0
##          2  0  0  1
```

At a high level, we see the most clustering in the complete model, where the data are split about 80/20 between two clusters. In all the models, moving from 2 to 3 clusters moves only one state into the third cluster.

K-Means

```
kmeans <- kmeans(predictors,
                 centers = 2,
                 nstart = 15)

predictors$Cluster <- as.factor(kmeans$cluster) # save clusters as factor

# Prep for grouping
t <- as.table(kmeans$cluster)
t <- data.frame(t)
rownames(t) <- states$stateabv
colnames(t)[colnames(t)=="Freq"] <- "Assignment"
t$Var1 <- NULL

kmeans_results <- t

kmeans_results %>%
  count(Assignment)
```

```
## # A tibble: 2 x 2
##   Assignment     n
##        <int> <int>
## 1          1     6
## 2          2    39
```

The K-means algorithm with two clusters separated 6 observations out from the rest of the data – this is somewhat consistent with the AHC model above, which also created two groups of disproportional sizes. The states that appear in this second group appear to be higher income, higher population states, such as California, New York, and Illinois.

Gaussian Mixture Model

```
gmm1 <- mvnormalmixEM(predictors[,-5], k = 2) # fit the GMM using EM and 2 comps
```

```
## number of iterations= 18
```

```
gmm_results <- as.data.frame(gmm1$posterior) %>%
  cbind(states) %>%
  mutate(assignment = if_else(comp.1 < comp.2, 2, 1)) %>%
  select(stateabv, assignment)

gmm_results %>%
  count(assignment)
```

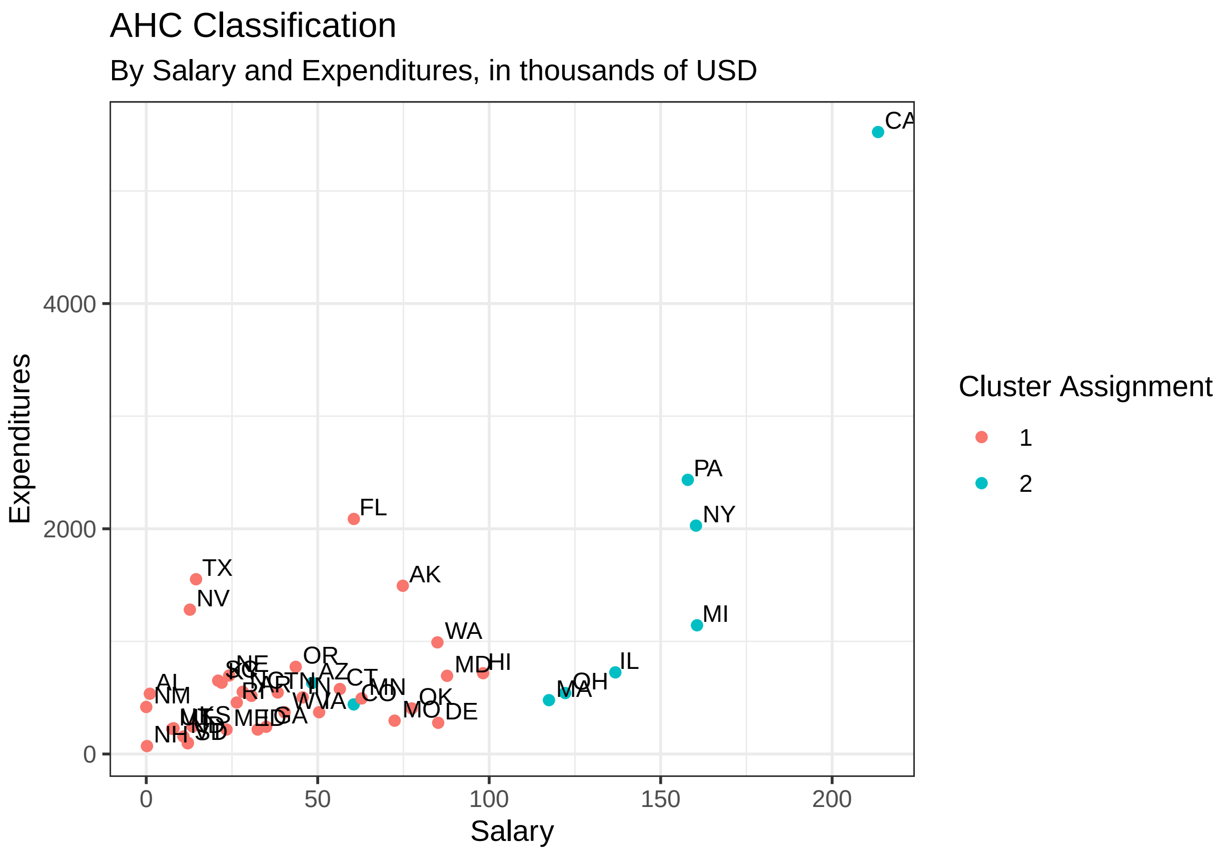
```
## # A tibble: 2 x 2
##   assignment     n
##        <dbl> <int>
## 1          1     6
## 2          2    39
```

Similarly, we have a small group of the same profile that is separated out into a smaller cluster. After this stage, I have a higher degree of confidence that the models are picking up nonrandom variation among the states.

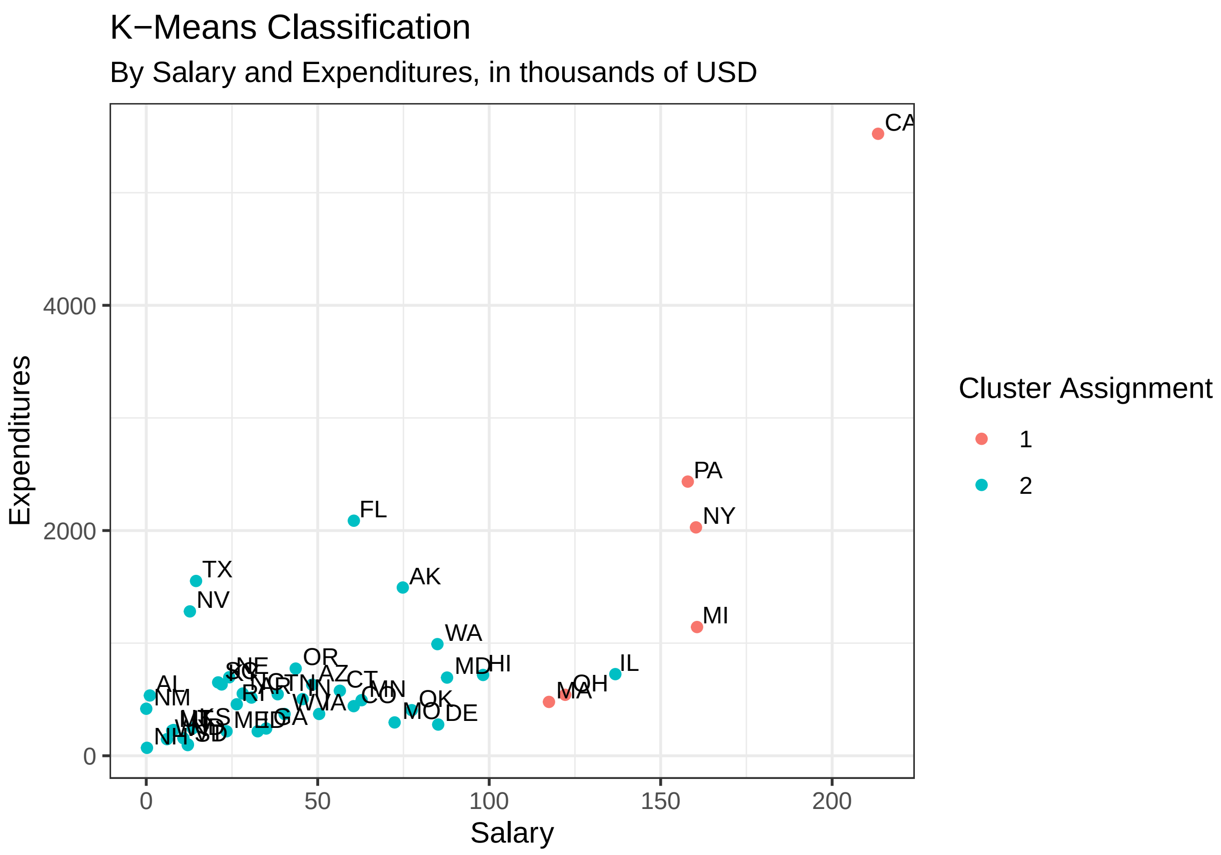
Compare Results

Visualizing based on Salary and Expenditures

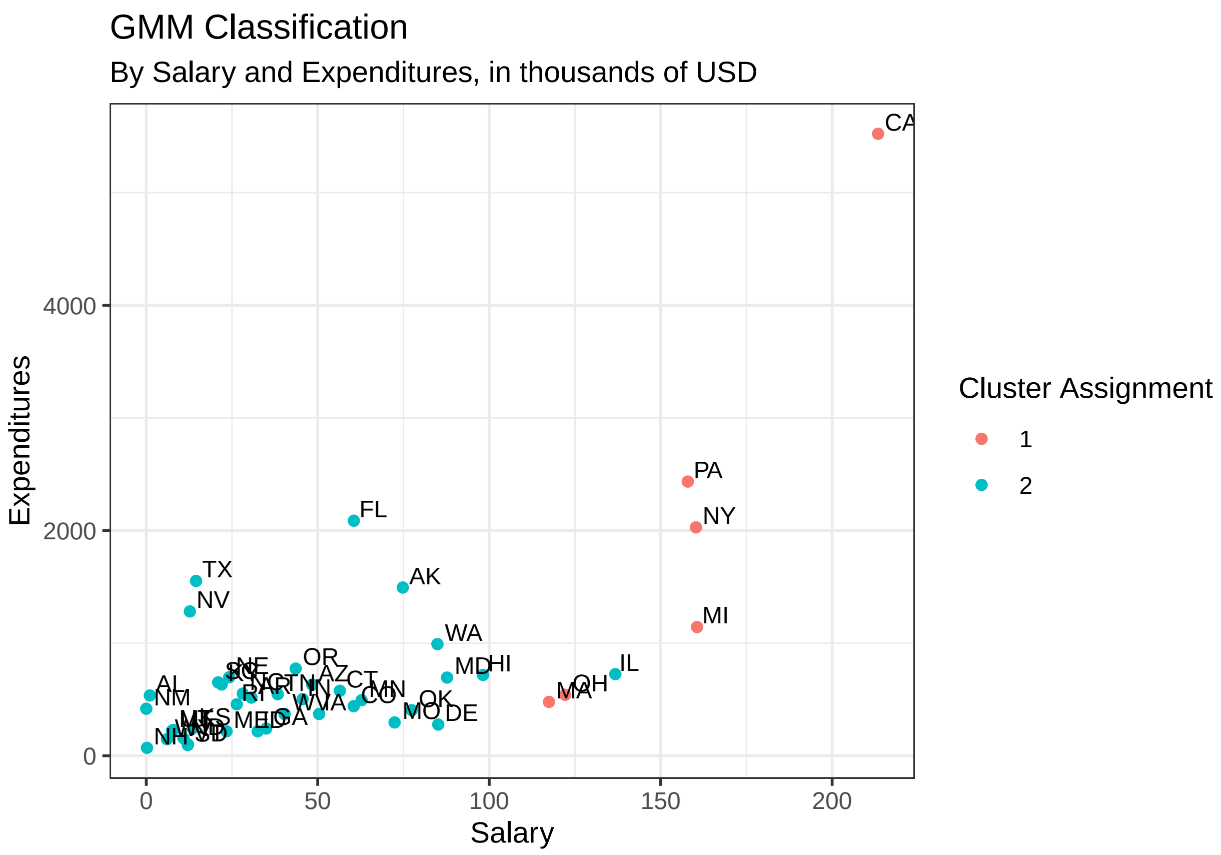
```
#Salary and Expenditures
#AHC
ahc_cuts <- as.data.frame(cuts_comp[,1]) %>%
  add_rownames(var = "statenum")

ahc_salary <- data_clean_ahc %>%
  add_rownames(var = "statenum") %>%
  left_join(ahc_cuts) %>%
  mutate(`Cluster Assignment` = as.factor(`cuts_comp[, 1]`)) %>%
  na.omit() %>%
  ggplot() +
  geom_point(aes(salary_real, expend, color = `Cluster Assignment`)) +
  theme_bw() +
  geom_text(aes(salary_real, expend, label=stateabv),hjust=-.2, vjust=-.2, size = 3) +
  labs(title = "AHC Classification",
       subtitle = "By Salary and Expenditures, in thousands of USD",
       x = "Salary",
       y = "Expenditures")

##K-Means
km_salary <- kmeans_results %>%
  cbind(data_clean) %>%
  add_rownames(var = "stateabv") %>%
  mutate(`Cluster Assignment` = as.factor(Assignment)) %>%
  ggplot() +
  geom_point(aes(salary_real, expend, color = `Cluster Assignment`)) +
  theme_bw() +
  geom_text(aes(salary_real, expend, label=stateabv),hjust=-.2, vjust=-.2, size = 3) +
  labs(title = "K-Means Classification",
       subtitle = "By Salary and Expenditures, in thousands of USD",
       x = "Salary",
       y = "Expenditures")

##GMM
gmm_salary <- gmm_results %>%
  cbind(data_clean) %>%
  mutate(`Cluster Assignment` = as.factor(assignment)) %>%
  ggplot() +
  geom_point(aes(salary_real, expend, color = `Cluster Assignment`)) +
  theme_bw() +
  geom_text(aes(salary_real, expend, label=stateabv),hjust=-.2, vjust=-.2, size = 3) +
  labs(title = "GMM Classification",
       subtitle = "By Salary and Expenditures, in thousands of USD",
       x = "Salary",
       y = "Expenditures")

ahc_salary
```



```
km_salary
```


K−Means Classification
By Salary and Expenditures, in thousands of USD
Expenditures
Salary
Cluster Assignment
1
2
0
2000
4000
0
50
100
150
200
CA
PA
NY
MI
OH
MA
IL
FL
AK
TX
NV
WA
MD
HI
OR
AZ
CT
MN
CO
OK
MO
DE
AL
NM
NH
SD
NE
RI
GA
ME
WI
IA

```
gmm_salary
```



Visualizing based on Session Length

```
##AHC
ahc_length <- data_clean_ahc %>%
  add_rownames(var = "statenum") %>%
  left_join(ahc_cuts) %>%
  mutate(`cuts_comp[, 1]` = if_else(`cuts_comp[, 1]` == 1, 2, 1)) %>%
  mutate(`Cluster Assignment` = as.factor(`cuts_comp[, 1]`)) %>%
  na.omit() %>%
  ggplot() +
  geom_histogram(aes(t_slength, fill = `Cluster Assignment`)) +
  theme_bw() +
   labs(title = "AHC Classification",
       subtitle = "By Total Session Length",
       x = "Session Length (Days)",
       y = "Frequency") +
  scale_y_continuous(breaks= pretty_breaks())
```

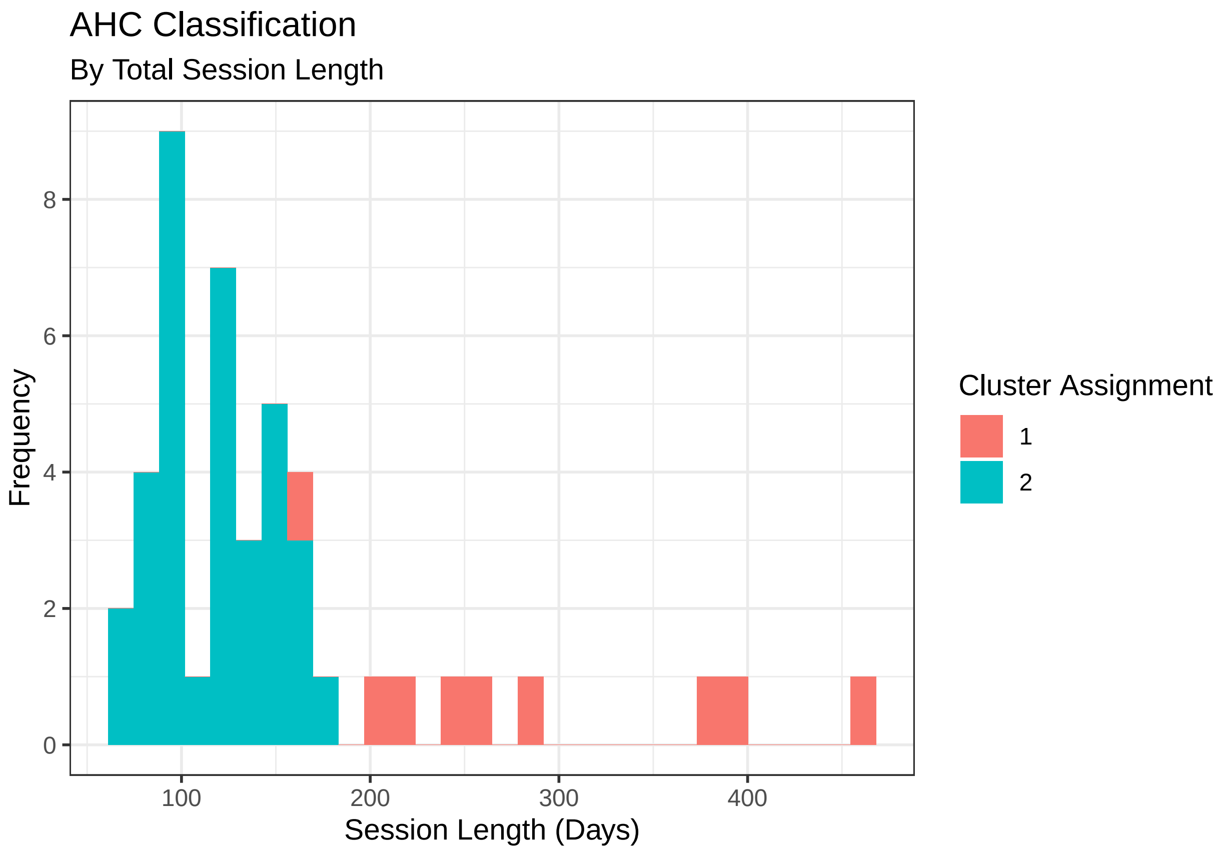
```
## Warning: Deprecated, use tibble::rownames_to_column() instead.

## Joining, by = "statenum"
```

```
##K-Means
km_length <- kmeans_results %>%
  cbind(data_clean) %>%
  add_rownames(var = "stateabv") %>%
  mutate(`Cluster Assignment` = as.factor(Assignment)) %>%
  ggplot() +
  geom_histogram(aes(t_slength, fill = `Cluster Assignment`)) +
  theme_bw() +
   labs(title = "K-Means Classification",
       subtitle = "By Total Session Length",
       x = "Session Length (Days)",
       y = "Frequency") +
  scale_y_continuous(breaks= pretty_breaks())
```

```
## Warning: Deprecated, use tibble::rownames_to_column() instead.
```

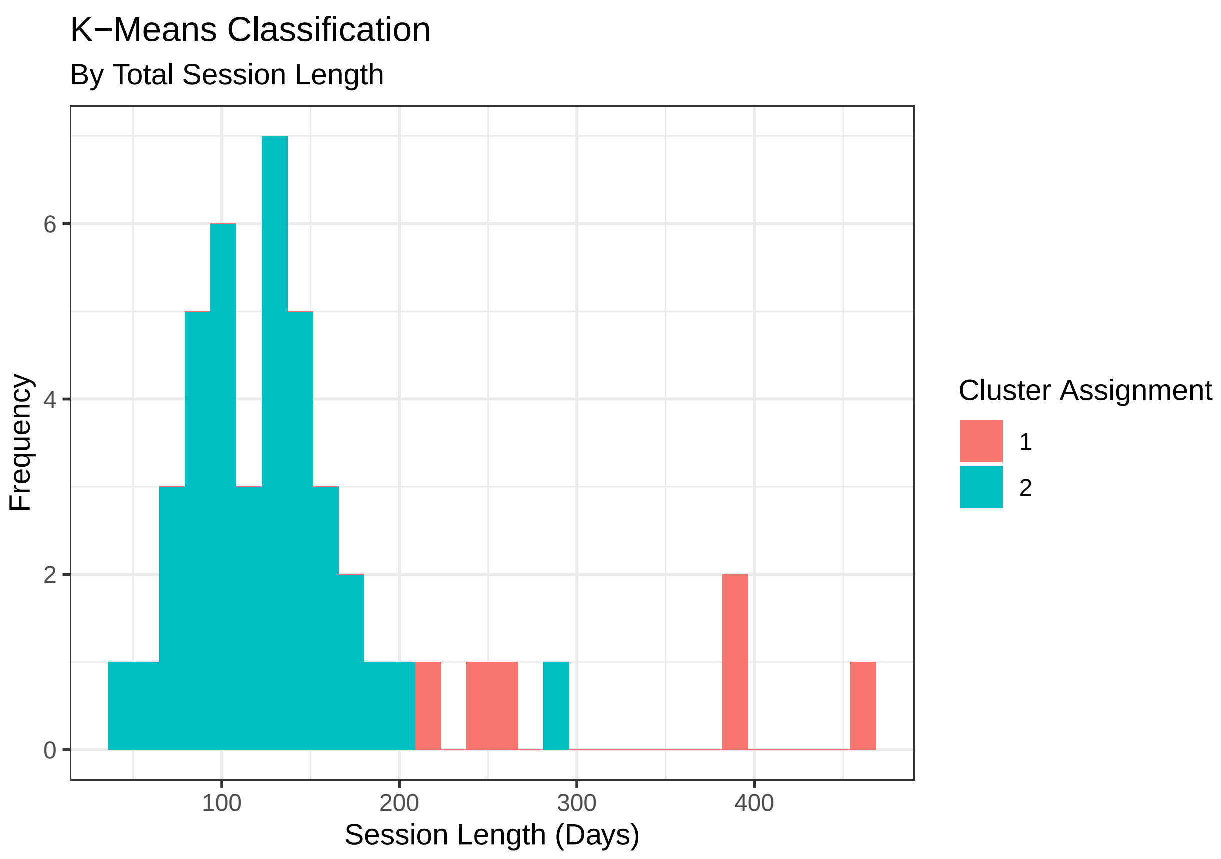
```
##GMM
gmm_length <- gmm_results %>%
  cbind(data_clean) %>%
  mutate(`Cluster Assignment` = as.factor(assignment)) %>%
  ggplot() +
  geom_histogram(aes(t_slength, fill = `Cluster Assignment`)) +
  theme_bw() +
   labs(title = "GMM Classification",
       subtitle = "By Total Session Length",
       x = "Session Length (Days)",
       y = "Frequency") +
  scale_y_continuous(breaks= pretty_breaks())

ahc_length
```

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



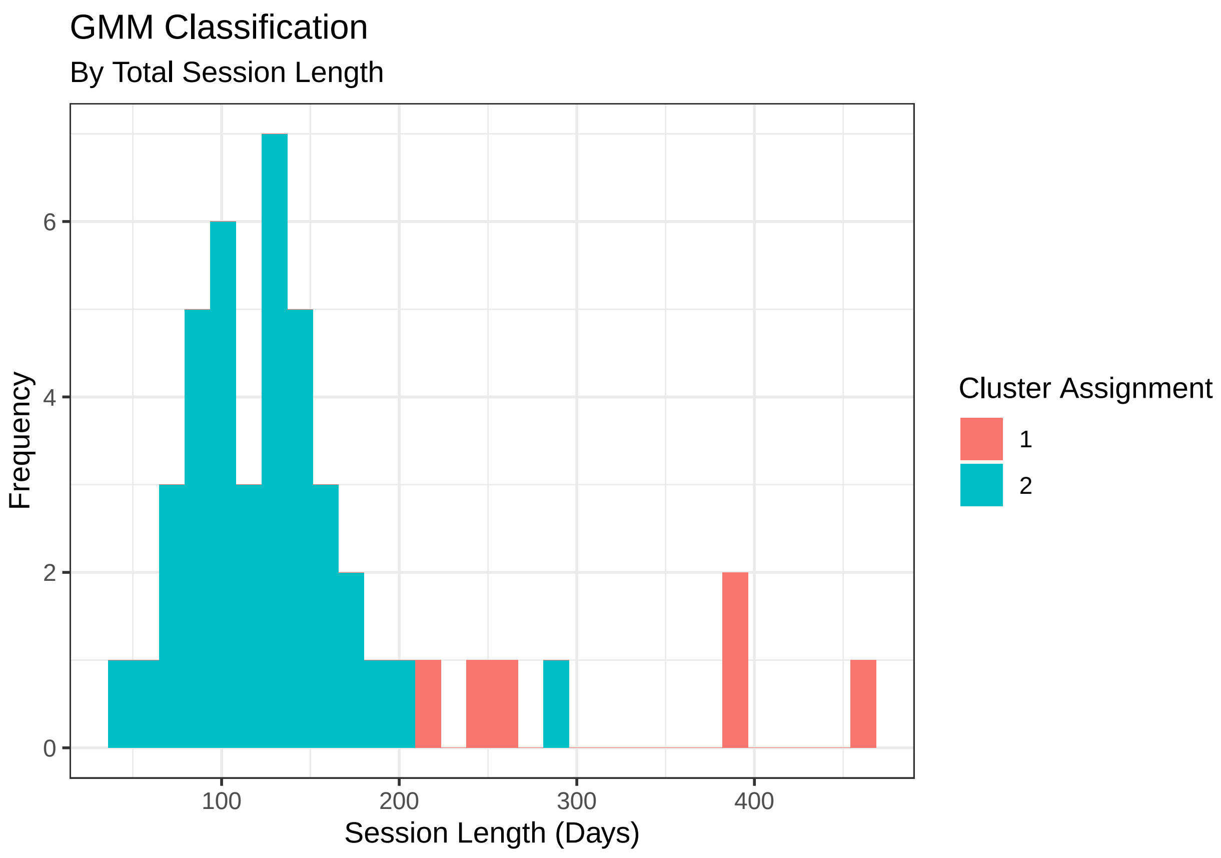
```
km_length
```

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



```
gmm_length
```

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



Validation - Silhouette Width

```
#AHC SW validation
hier_valid <- clValid(as.matrix(predictors),
        nClust = 2:8,
        clMethods = "hierarchical",
        validation = "internal",
        method = "complete")

hier_measures <- measures(hier_valid)

hier_df <- as.data.frame(hier_measures)
colnames(hier_df) <- c("2", "3", "4", "5", "6", "7","8")

hier_final_df <- hier_df %>%
  rownames_to_column() %>%
  gather(key = "clusters",
         value = "score",
         2:8) %>%
  mutate(model = "hierarchical")

#K-means SW validation
kmeans_valid <- clValid(as.matrix(predictors),
        nClust = 2:8,
        clMethods = "kmeans",
```

```r
        validation = "internal")

kmeans_measures <- measures(kmeans_valid)

km_df <- as.data.frame(kmeans_measures)
colnames(km_df) <- c("2", "3", "4", "5", "6", "7","8")

km_final_df <- km_df %>%
  rownames_to_column() %>%
  gather(key = "clusters",
         value = "score",
         2:8) %>%
  mutate(model = "kmeans")

#GMM SW validation
gmm_valid <- clValid(as.matrix(predictors),
        nClust = 2:8,
        clMethods = "model",
        validation = "internal")

gmm_measures <- measures(gmm_valid)

gmm_df <- as.data.frame(gmm_measures)
colnames(gmm_df) <- c("2", "3", "4", "5", "6", "7","8")

gmm_final_df <- gmm_df %>%
  rownames_to_column() %>%
  gather(key = "clusters",
         value = "score",
         2:8) %>%
  mutate(model = "GMM")

##Combine all to a single DF for plotting
final_validations <- rbind(hier_final_df, km_final_df, gmm_final_df) %>%
  mutate(clusters = as.numeric(clusters))

final_validations %>%
  filter(rowname == "Silhouette") %>%
ggplot(aes(clusters, score, color = model)) +
  geom_line() +
  labs(title = "Silhouette Width of Each Model",
       x = "Number of Clusters",
       y = "Silhouette Width")
```

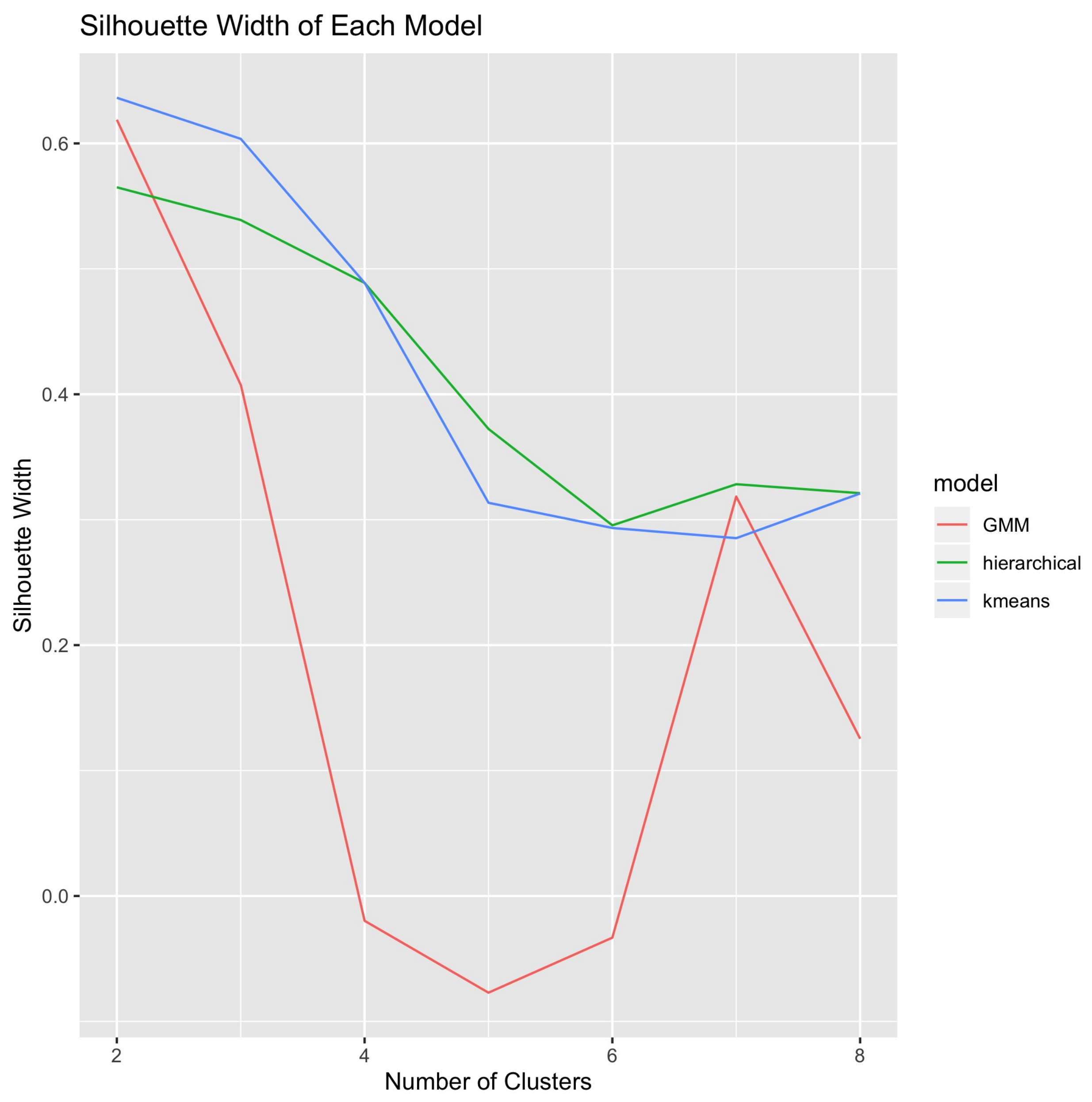


Figure 1: “”

Discussion

From the results of the validation procedure, we can see that the Agglomerative Hierarchical Clustering and K-means models performed mostly similar to each other, where the Gaussian Mixture Model expereince more variation between model versions with different cluster numbers. For all clustering methods, the optimal fit was two clusters, likely indicating that state legislatures can be thought of as a binary “professional” or “not professional”. Strictly leveragiing silhouette width, the optimal clustering method is K-means with two clusters because it has the highest silhouette width value – however, this also seems to be a valid approach because there is less variation across clusters thn the GMM.

Regardless of validation statistics, an example of selecting a sub-optimal clustering method would be selecting

K-means or Agglomerative Hierarchical Clustering when we have a strong theory supporting the notion that the data is a function of multiple normal distributions. For example, if we think that individuals vote fundamentally different based on their party affiliation as Democrat or Republican, it would be a sub-optimal clustering approach to pool all observations as being a function of a single distriibution (and use K-means or Agglomerative Hierarchical Clustering).