Homework 1

Computational Methods for American Politics

Sanittawan Tan

10/22/2019

Load Libraries

```
library(ggplot2)
library(tidyverse)
```

```
## Registered S3 method overwritten by 'rvest':
##   method            from
##   read_xml.response xml2
```

```
## -- Attaching packages ------------------------------------------------ tidyverse 1.2.1 --
```

```
## v tibble  2.1.3     v purrr   0.3.3
## v tidyr   1.0.0     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0
```

```
## -- Conflicts --------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(clustertend)
library(factoextra)
```

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

```
library(seriation)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(mixtools)
```

```
## mixtools package, version 1.1.0, Released 2017-03-10
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

```
library(plotGMM)
library(clValid)
```

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

```
library(cluster)
```

Part 1 - Load data set

```
dir <- "/Users/sanittawan/Documents/Nikki/UChicago/Classes/Autumn_2019/com_methods_am_politics/psets/pr
load(dir)
SLPdta <- as_tibble(x)
```

Part 2 - Munge the data

a) Select only continuous features

```
SLPdta_filtered <- SLPdta %>%
  select(-mds1, -mds2)
```

b) Restrict data to 2009/10 legislative session

```
SLPdta_filtered2 <- SLPdta_filtered %>%
    filter(sessid == "2009/10")
```

c) Omit missing values

```
SLPdta_filtered3 <- na.omit(SLPdta_filtered2)
# idea from http://www.programmingr.com/examples/remove-na-rows-in-r/
```

d) Standardize input features

```
SLPdta_short <- SLPdta_filtered3 %>%
  mutate(std_t_slength = scale(t_slength, center = TRUE, scale = TRUE),
         std_slength = scale(slength, center = TRUE, scale = TRUE),
         std_salary_real = scale(salary_real, center = TRUE, scale = TRUE),
         std_expend = scale(expend, center = TRUE, scale = TRUE))
```

Here I verify that the normalization of the input features work as expected.

```
SLPdta_short %>%
    summarize(
        mean_std_t_slength = round(mean(std_t_slength), digits = 2),
        sd_std_t_slength = sd(std_t_slength),
        mean_std_slength = round(mean(std_slength), digits = 2),
        sd_std_slength = sd(std_slength),
        mean_std_salaray_real = round(mean(std_salary_real), digits = 2),
        sd_std_salarary_real = sd(std_salary_real),
        mean_std_expend = round(mean(std_expend), digits = 2),
        sd_std_expend = sd(std_expend)
    ) %>%
    t()
```

```
##                          [,1]
## mean_std_t_slength          0
## sd_std_t_slength            1
## mean_std_slength            0
## sd_std_slength              1
## mean_std_salaray_real       0
## sd_std_salarary_real        1
## mean_std_expend             0
## sd_std_expend               1
```

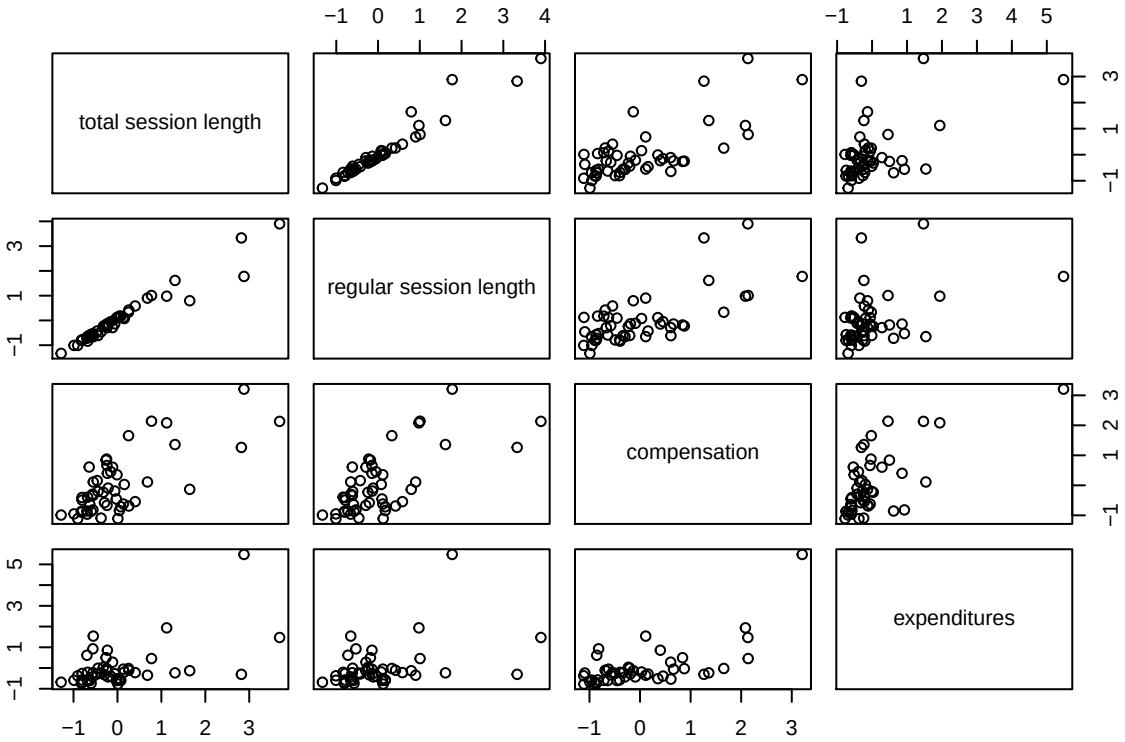
e) Miscellaneous

```
state_names <- unique(SLPdta_filtered$state)
state_abbr <- unique(SLPdta_filtered$stateabv)
```

Part 3 - Diagnose Clusterability

First, I generated series of pair plots between two variables to see if there are any natural clusters in the data. There are 3 types of groupings that we discussed in class: location, shape, and density. It is quite obvious from these plots that there are some groupings based on location and density. Take the plot between compensation and total session length (plot row 3, col 1) as an example. A set of points tend to cluster in the bottom left whereas another set of points tend to spread over to the top right corner. These two set have very different density. Similar pattern also occurs in the total session length against regular session length plot (plot row 2, col 1). A set of points form almost a straight line in the bottom left corner while a handful of the remaining points scatter over to the top right corner.

```
pairs(SLPdta_short[10:13], labels = c("total session length", "regular session length", "compensation",
```



Second, I will compute Hopkins Statistics to see if the data is random. It is necessary to set up a hypothesis testing with the null and alternative hypotheses as follows:

$$H_0 : \text{Data are uniformly distributed} H_a : \text{Data are not uniformly distributed}$$

Generally, a threshold to reject the null hypothesis is below 0.5. We will compute it using an R function from the clustertend package.

```
set.seed(1234)
subset_cols <- SLPdta_short %>% select(10:13)
hopkins(subset_cols, n = nrow(subset_cols) - 1)
```

```
## $H
## [1] 0.1740508
```

```
# idea from http://www.sthda.com/english/wiki/print.php?id=238
```

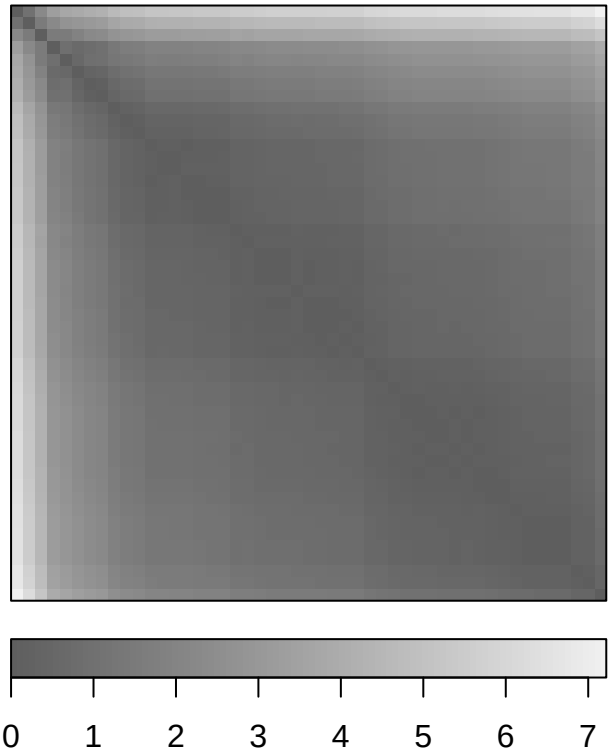
I found that the resulting Hopkins statistics is 0.17 which is far below the 0.5 threshold. Therefore, null hypothesis can be rejected and our data is clusterable.

Second, I will take a visual inspection of the clusterability via the Ordered Dissimilarity Images (ODI) plots.

Total Session Length vs Regular Session Length

There seem to be 2 clusters. One small cluster at the top left corner and the other big cluster spanning diagonally to the bottom right corner.

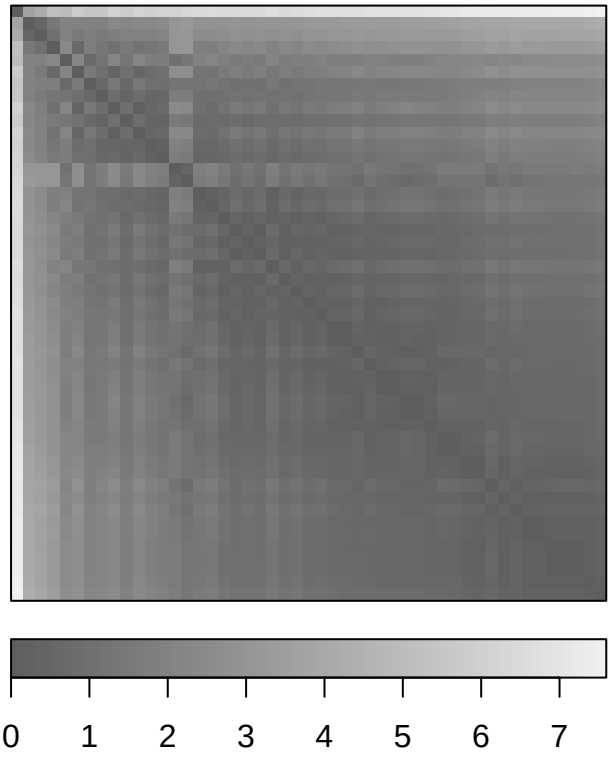
```
lengths_dist <- dist(SLPdta_short[c("std_t_slength", "std_slength")])
p1 <- dissplot(lengths_dist)
```



Salary vs Expenditures

There seem to be one clear cluster at the top left corner. Besides, the pattern does not look so clear.

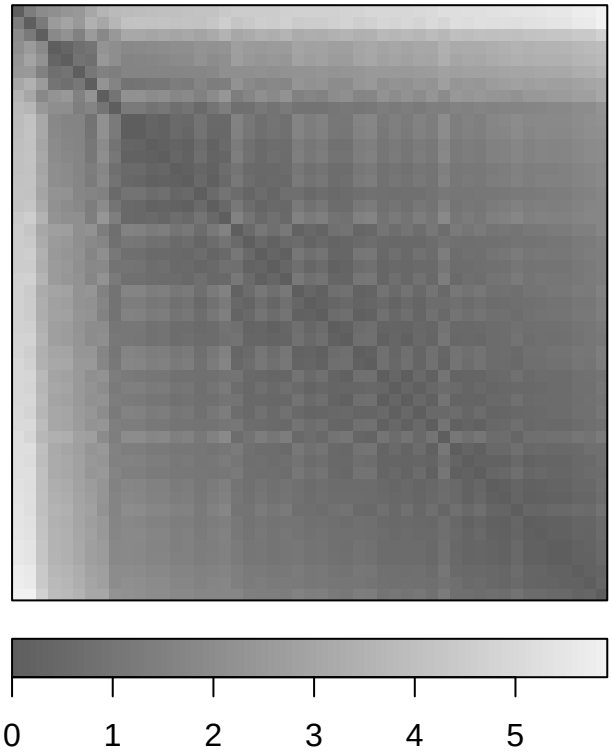
```
fin_dist <- dist(SLPdta_short[c("std_salary_real", "std_expend")])
p2 <- dissplot(fin_dist)
```



Total Session Length vs Salary

There seems to be two clear clusters at the left corner and in the right corner.

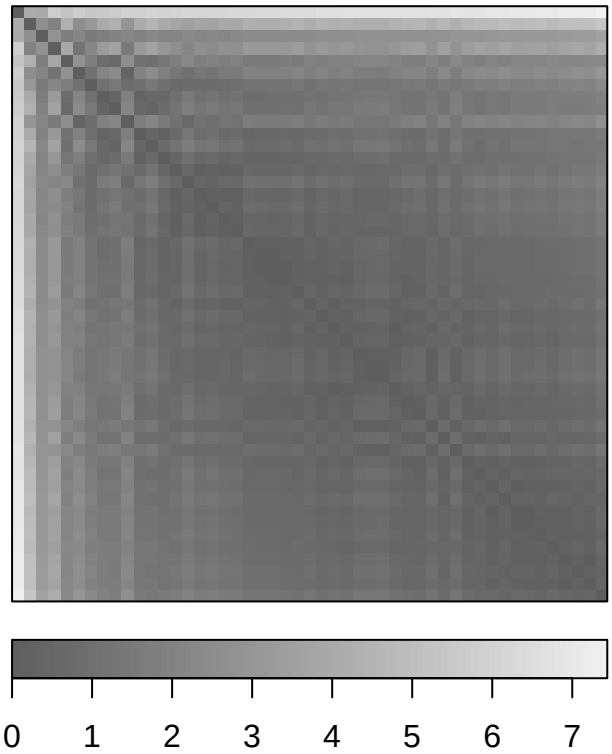
```
tlengthsalary_dist <- dist(SLPdta_short[c("std_t_slength", "std_salary_real")])
p3 <- dissplot(tlengthsalary_dist)
```



Total Session Length vs Expenditures

There seems to be a large cluster in the bottom right though the pattern does not look clear.

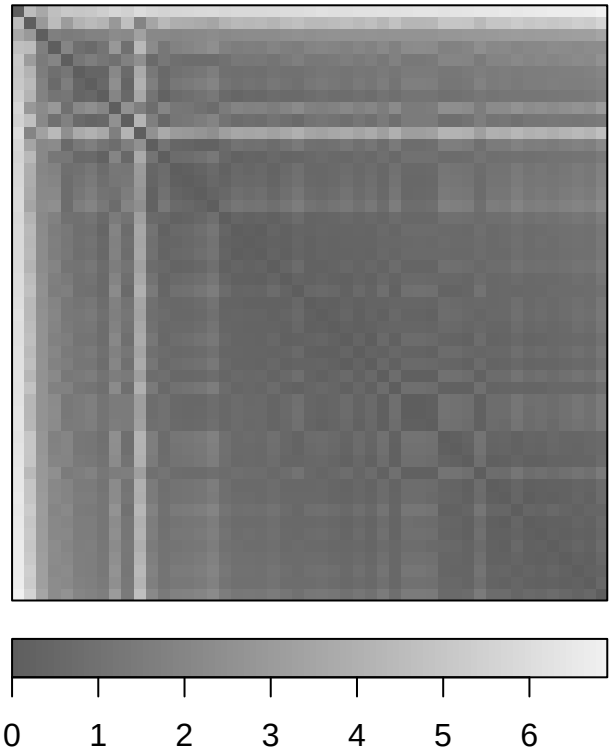
```
tlengthex_dist <- dist(SLPdta_short[c("std_t_slength", "std_expend")])
p4 <- dissplot(tlengthex_dist)
```



Regular Session Length vs Expenditures

There seem to be 4 clusters divided by white lines.

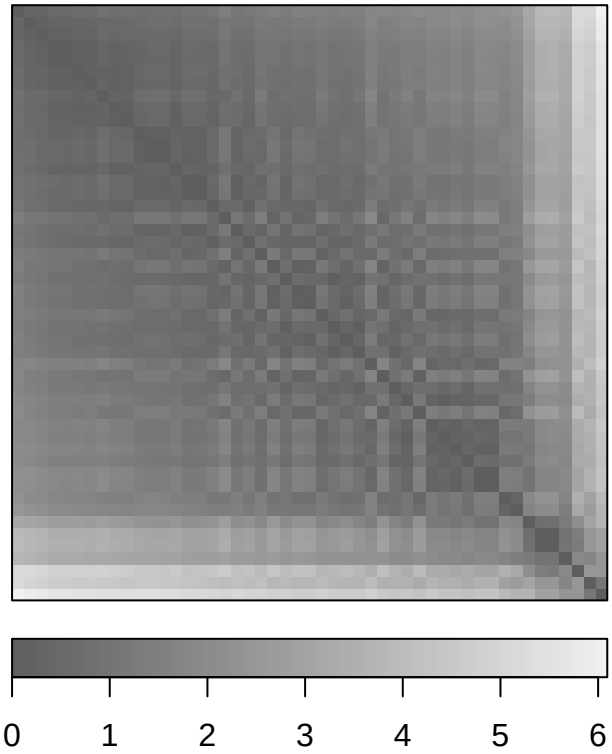
```
rlengthex_dist <- dist(SLPdta_short[c("std_slength", "std_expend")])
p5 <- dissplot(rlengthex_dist)
```



Regular Session Length vs Salary

There seems to be a large cluster on the top left corner (large square) vs small square in the bottom right.

```
rlengthsalary_dist <- dist(SLPdta_short[c("std_slength", "std_salary_real")])
p6 <- dissplot(rlengthsalary_dist)
```



Based on all dianostic tests, namely pair plots, the Hopkins statistics and the ODI plots, that I performed, it is safe to proceed to fitting with clustering methods because the data shows that there are potential clusters and the data is not random. In other words. it is likely that there is natural, non-random structure in the data set.

Part 4 - Agglomerative Hierachical Clustering

As an exercise, I tried out single, complete, average, and centroid linkage methods.

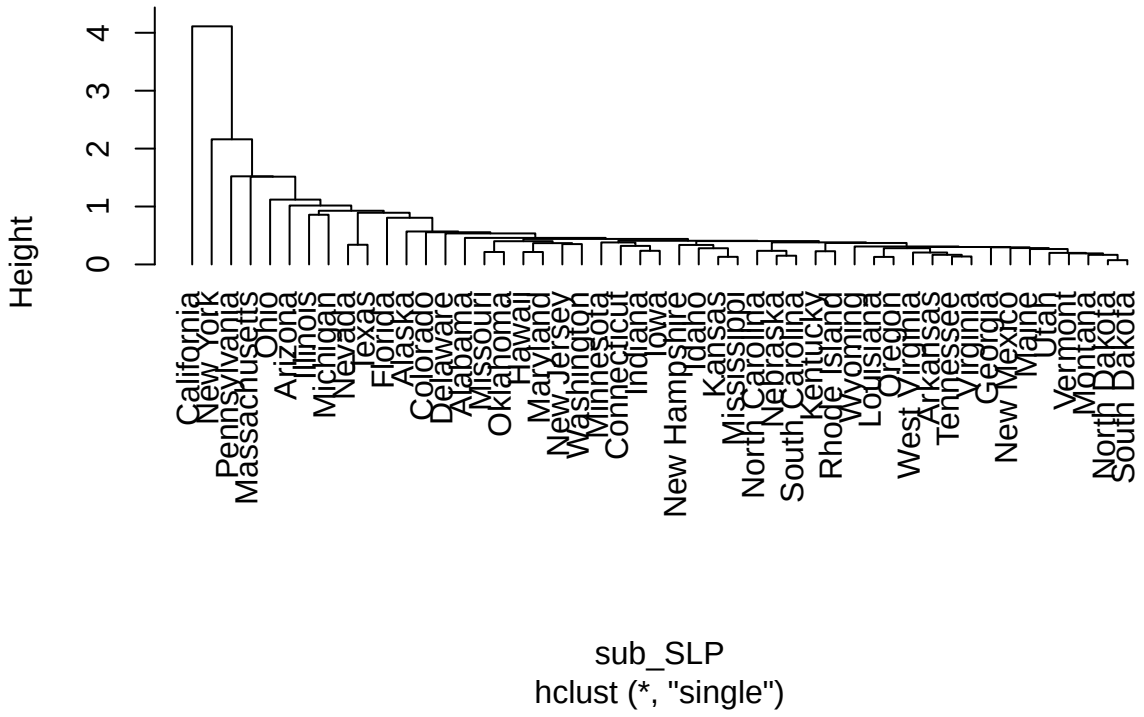
```
SLPdf <- as.data.frame(SLPdta_short)
rownames(SLPdf) <- SLPdf$state
sub_SLP <- SLPdf %>%
  select(std_t_slength, std_salary_real, std_salary_real, std_expend) %>%
  dist()

sub_SLP
```

Single Linkage Method

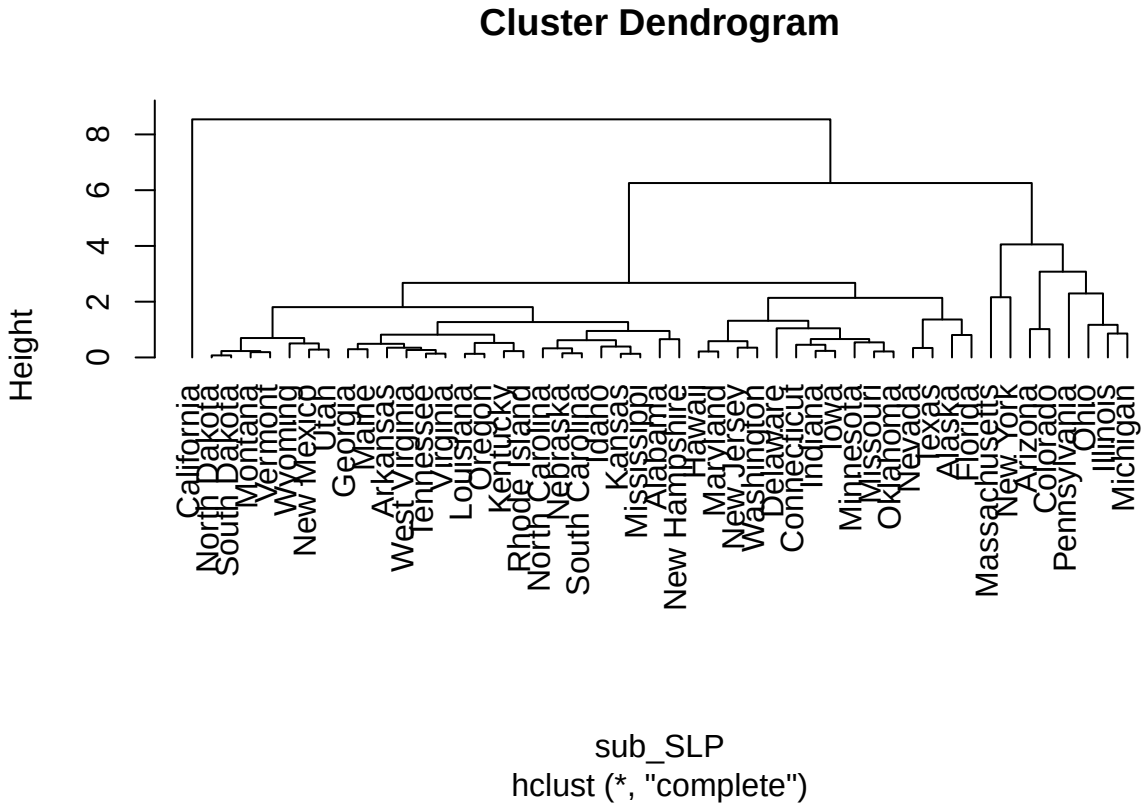
Single linkage which minimizes inter-cluster dissimilarity results in an elongated dendogram as we discussed in class.

```
hc_single <- hclust(sub_SLP,
                    method = "single"); plot(hc_single, hang = -1)
```



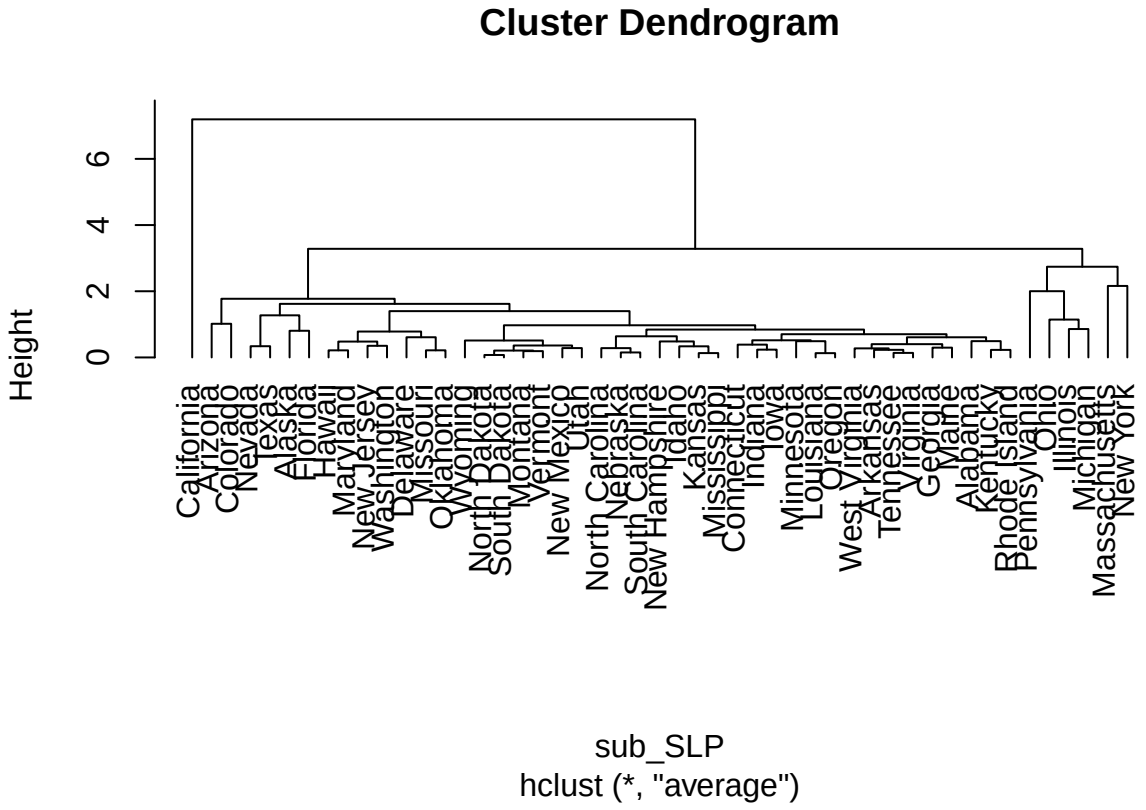
Complete linkage which maximizes inter-cluster dissimilarity results in a balanced dendogram.

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



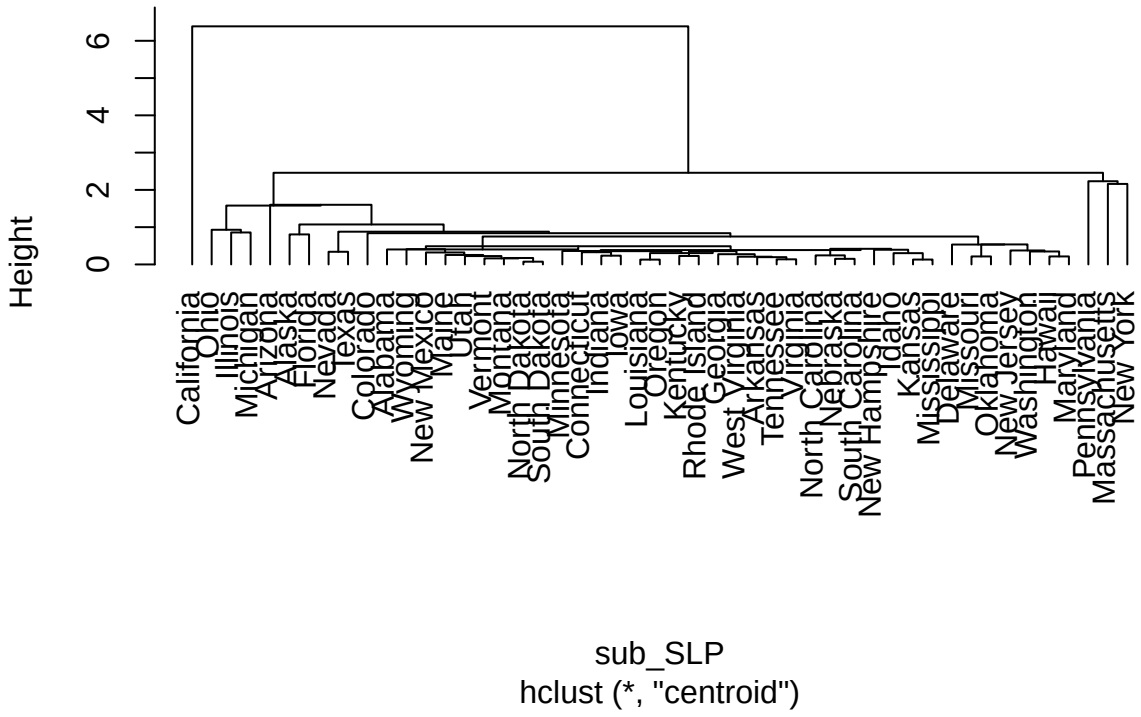
Average linkage which calculates the mean inter-cluster dissimilarity results in a balanced dendogram as well.

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



Centroid linkage calculates the dissimilarity between the centroid of clusters.

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



Discussion

Overall, the linkage methods do have impact on how the data points are clustered. However, one consistent pattern stands out in all methods: California seems to be its own cluster with only one member (itself). Furthermore, in models that use avarage, centroid, and complete methods, if the trees are cut at the height between 2 and 4, there seems to be 3 distinct clusters. One such cluster includes only California which is the smallest one. Another cluster is on the right with a few more states. The list of states in this cluster across all 3 dendograms is interesting since Massachusetts, Pennsylvania, New York appear in all of them. The last cluster is in the middle where most of the states are members.

Looking closer to the members of each cluster, one observation is that the states with full-time or semi-full-time legislators tend to be clustered together. According to the National Conference of State Legislatures, California, Michigan, New York, Pennsylvania, Massachusetts, Alaska, Hawaii, Illinois, Ohio, and Wisconsin have legislators who are full time and with large staffers. This may explain why we see Massachusetts, Pennsylvania, New York appear in the same cluster repeatedly and speak to how the algorithm clusters them based on the session lengths, pay and expenditures which, in turn, suggests that these states' legislatures are more professional or similar to congress than others. However, this explanation still fails to account for why California has its own cluster. This may suggests that there are other factors at play which differentiate California from its peers.

After seeing the results from the agglomerative heirachical clustering, I am curious about the data. Thus, I made a side exploration into 4 features by looking at the feature means and the states.

```
SLP_sum <- SLPdta_short %>%
    group_by(state) %>%
    summarize(mean_ts_length = mean(t_slength, na.rm = TRUE),
              mean_slength = mean(slength, na.rm = TRUE),
              mean_salary = mean(salary_real, na.rm = TRUE),
              mean_expend = mean(expend, na.rm = TRUE)
              )
```

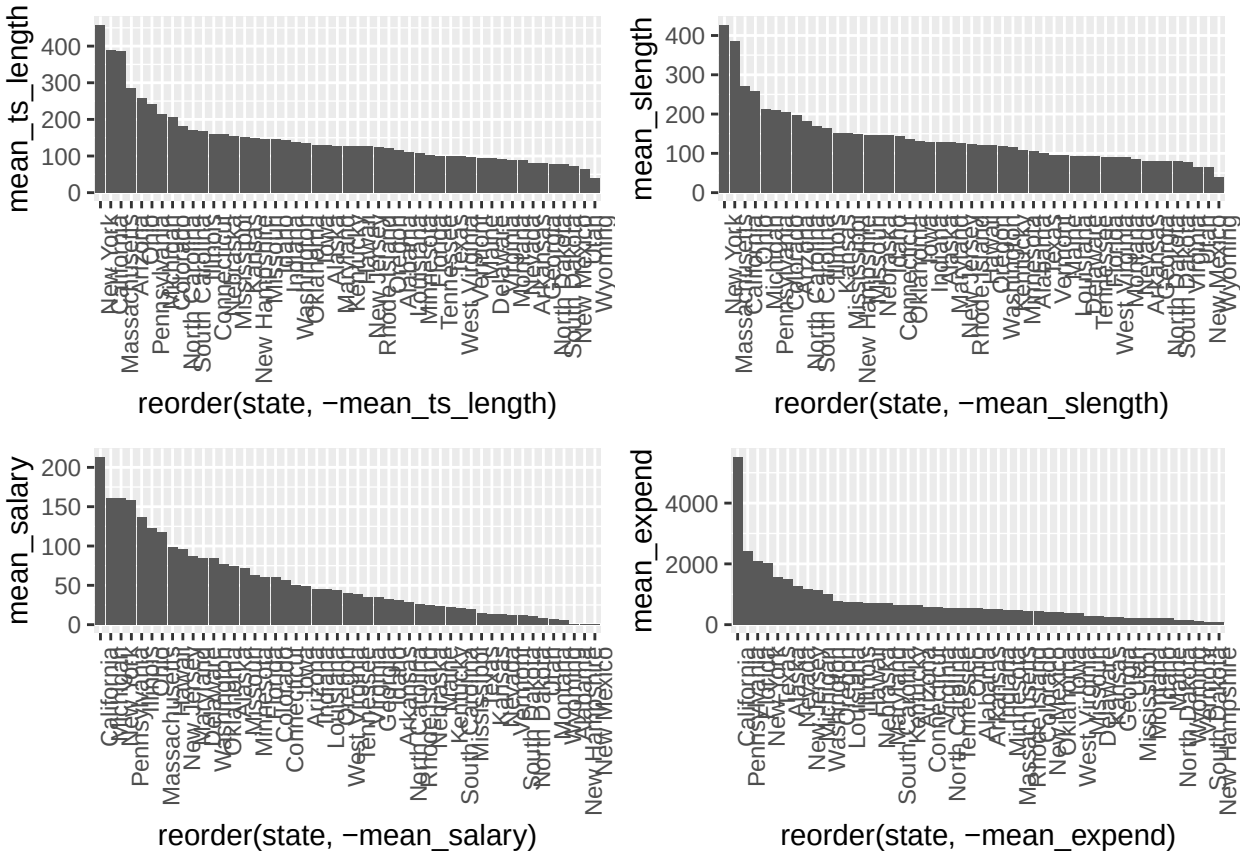
```
g1 <- ggplot(SLP_sum, aes(x = reorder(state, -mean_ts_length), y = mean_ts_length))
g1 <- g1 + geom_bar(stat = "identity")
g1 <- g1 + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
g2 <- ggplot(SLP_sum, aes(x = reorder(state, -mean_slength), y = mean_slength))
g2 <- g2 + geom_bar(stat = "identity")
g2 <- g2 + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
g3 <- ggplot(SLP_sum, aes(x = reorder(state, -mean_salary), y = mean_salary))
g3 <- g3 + geom_bar(stat = "identity")
g3 <- g3 + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
g4 <- ggplot(SLP_sum, aes(x = reorder(state, -mean_expend), y = mean_expend))
g4 <- g4 + geom_bar(stat = "identity")
g4 <- g4 + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
grid.arrange(g1, g2, g3, g4)
```



Part 5 - k-Means

Note: as our raw features are in different units (length of time and dollars), I use the standardized versions of these features before applying k-means.[1]

```
set.seed(1234)
```

[1] See here and here

```
kmeans <- kmeans(SLPdf[10:13],
                 centers = 2,
                 nstart = 15)
```

```
SLPdf$Cluster <- as.factor(kmeans$cluster)
```

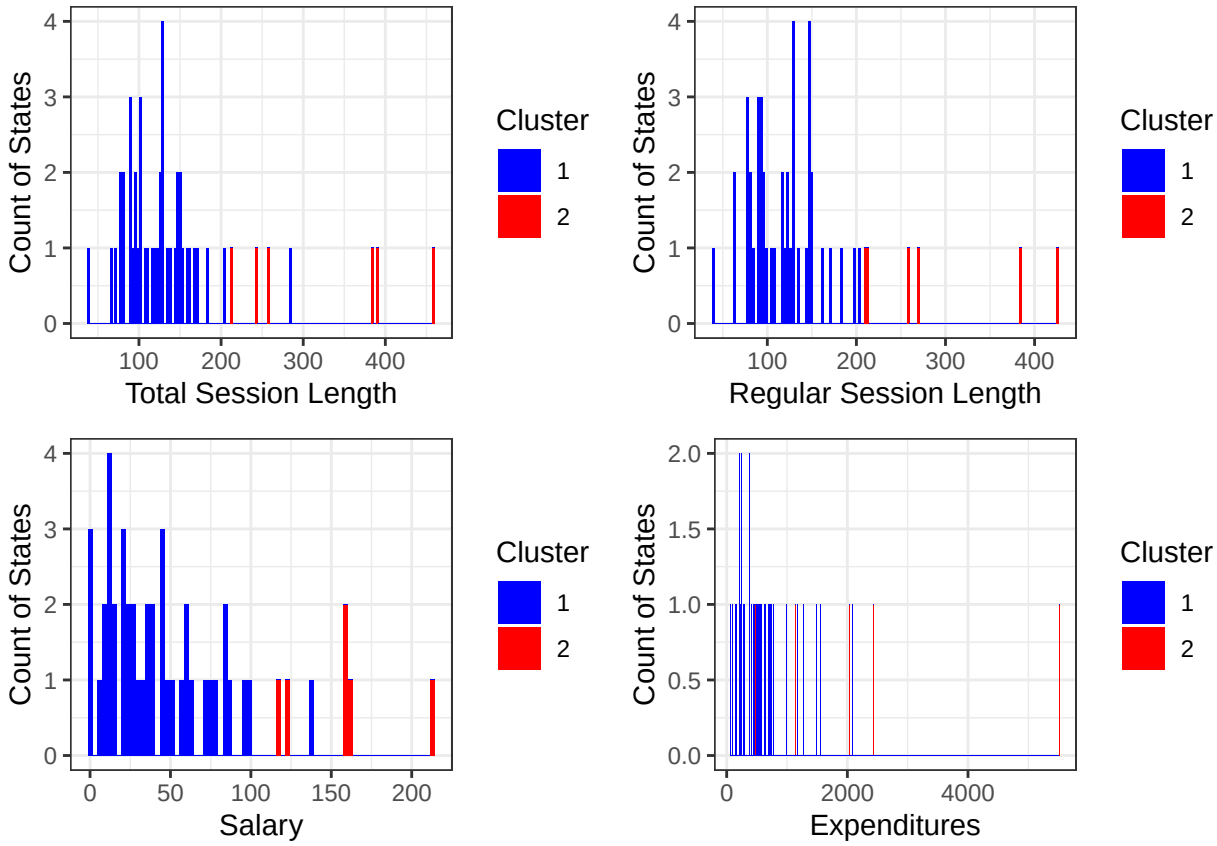
```
k1 <- ggplot(SLPdf, aes(t_slength, fill = Cluster)) +
  geom_histogram(binwidth = 3) +
  theme_bw() +
  scale_fill_manual(values=c("blue", "red")) +
  labs(x = "Total Session Length",
       y = "Count of States")
```

```
k2 <- ggplot(SLPdf, aes(slength, fill = Cluster)) +
  geom_histogram(binwidth = 3) +
  theme_bw() +
  scale_fill_manual(values=c("blue", "red")) +
  labs(x = "Regular Session Length",
       y = "Count of States")
```

```
k3 <- ggplot(SLPdf, aes(salary_real, fill = Cluster)) +
  geom_histogram(binwidth = 3) +
  theme_bw() +
  scale_fill_manual(values=c("blue", "red")) +
  labs(x = "Salary",
       y = "Count of States")
```

```
k4 <- ggplot(SLPdf, aes(expend, fill = Cluster)) +
  geom_histogram(binwidth = 3) +
  theme_bw() +
  scale_fill_manual(values=c("blue", "red")) +
  labs(x = "Expenditures",
       y = "Count of States")
```

```
grid.arrange(k1, k2, k3, k4)
```



Looking at the clusters in a descriptive fashion.

```
t <- as.table(kmeans$cluster)
(t <- data.frame(t))
```

```
##              Var1 Freq
## 1         Alabama    1
## 2          Alaska    1
## 3         Arizona    1
## 4        Arkansas    1
## 5      California    2
## 6        Colorado    1
## 7     Connecticut    1
## 8        Delaware    1
## 9         Florida    1
## 10        Georgia    1
## 11         Hawaii    1
## 12          Idaho    1
## 13       Illinois    1
## 14        Indiana    1
## 15           Iowa    1
## 16         Kansas    1
## 17       Kentucky    1
## 18      Louisiana    1
## 19          Maine    1
## 20       Maryland    1
## 21  Massachusetts    2
## 22       Michigan    2
```

```
## 23      Minnesota    1
## 24    Mississippi    1
## 25       Missouri    1
## 26        Montana    1
## 27       Nebraska    1
## 28         Nevada    1
## 29  New Hampshire    1
## 30     New Jersey    1
## 31     New Mexico    1
## 32       New York    2
## 33 North Carolina    1
## 34   North Dakota    1
## 35           Ohio    2
## 36       Oklahoma    1
## 37         Oregon    1
## 38   Pennsylvania    2
## 39   Rhode Island    1
## 40 South Carolina    1
## 41   South Dakota    1
## 42      Tennessee    1
## 43          Texas    1
## 44           Utah    1
## 45        Vermont    1
## 46       Virginia    1
## 47     Washington    1
## 48  West Virginia    1
## 49        Wyoming    1
```

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

```
subset(t, Assignment == 2)
```

```
##               Assignment
## California             2
## Massachusetts          2
## Michigan               2
## New York               2
## Ohio                   2
## Pennsylvania           2
```

```
subset(t, Assignment == 1)
```

```
##                Assignment
## Alabama                 1
## Alaska                  1
## Arizona                 1
## Arkansas                1
## Colorado                1
## Connecticut             1
## Delaware                1
## Florida                 1
## Georgia                 1
## Hawaii                  1
## Idaho                   1
```

```
## Illinois                1
## Indiana                 1
## Iowa                    1
## Kansas                  1
## Kentucky                1
## Louisiana               1
## Maine                   1
## Maryland                1
## Minnesota               1
## Mississippi             1
## Missouri                1
## Montana                 1
## Nebraska                1
## Nevada                  1
## New Hampshire           1
## New Jersey              1
## New Mexico              1
## North Carolina          1
## North Dakota            1
## Oklahoma                1
## Oregon                  1
## Rhode Island            1
## South Carolina          1
## South Dakota            1
## Tennessee               1
## Texas                   1
## Utah                    1
## Vermont                 1
## Virginia                1
## Washington              1
## West Virginia           1
## Wyoming                 1
```

Discussion

The output from k-means algorithm as shown in the plots suggests that there are some misclassifications or two clusters may not be sufficient. Plotting the clustering results against all four features, one at a time, highlights this observation. For instance, in the total session length plot, there are two states with more than 200 days that are classified in the first cluster which overlaps with the second cluster. Similar issue also occurs in the salary and expenditures plots. The cleanest plot is regular session length.

Examining the states that are in either clusters, I found that the results from k-means, to some extent, agree with the hierachical clustering model. This is because California, Massachusetts, Michigam, New York, Ohio, and Pennsylvania are in its own cluster in the k-means model.

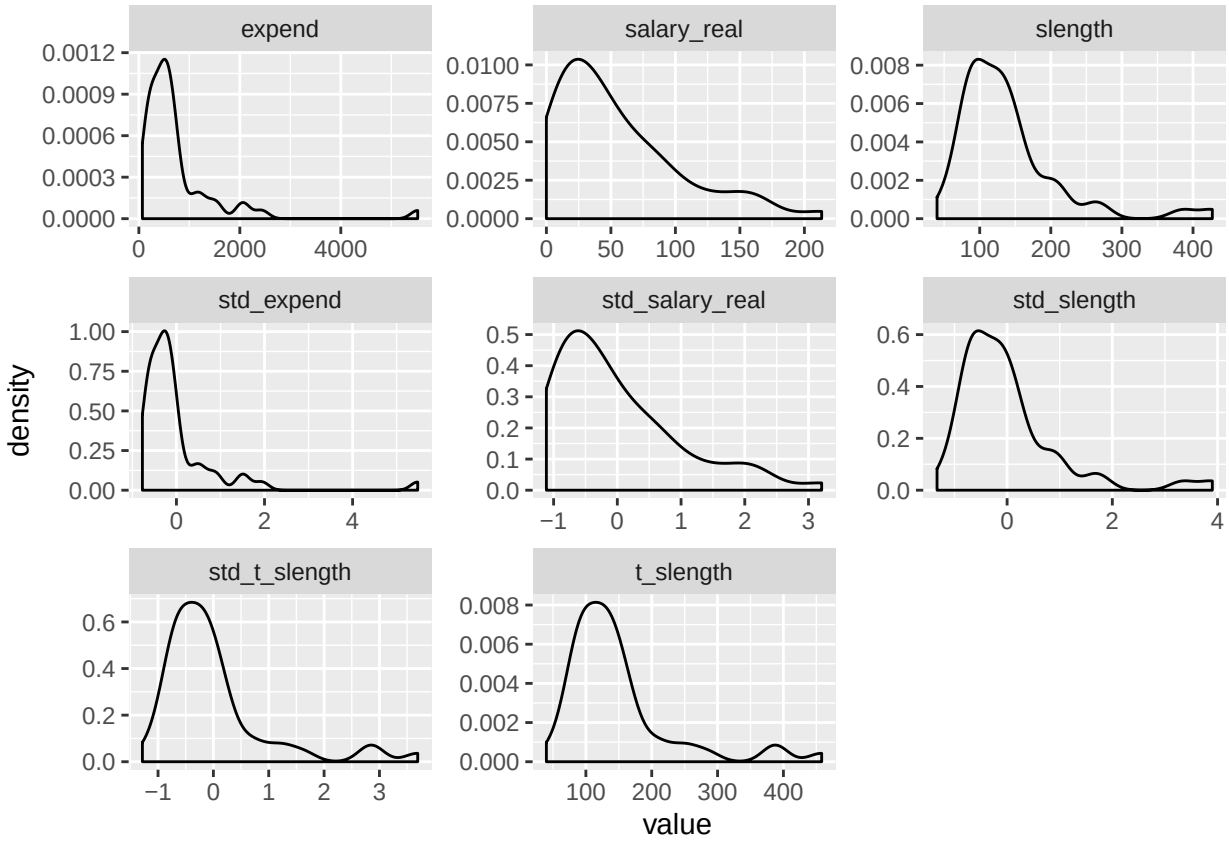
Part 6 - Gaussian Mixture Model (GMM)

This data set has 4 features and we hope to utilize all features for clustering. So, we need to use mixture models for multidimensional data.[2]. To make an argument about what distribution to select from, I created a series of density plot, both for the raw values and the standardized values of the features. The result is that the data normalization only changes the scale; however, the shapes of the distributions remain the same. One can argue that there seem to be varios overlaying normal distributions in these plots (where multiple peaks signify different distributions). Therefore, I will go ahead and choose Gaussian as my choice of distribution.

[2]Checked against p. 29 of Mixtools documentation and this tutorial

```
SLPdta_short %>%
  select(5, 6, 7, 8, 10, 11, 12, 13) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```




```
# modified from https://drsimonj.svbtle.com/quick-plot-of-all-variables
```

Fit the multivariate normal mixture model.

```
set.seed(12346)

# convert to matrix just to make sure elements are of the same type
SLPdf_matrix <- data.matrix(SLPdf[, 10:13])

# fit model
gmm_multi <- mvnormalmixEM(SLPdf_matrix,
                    k = 2, epsilon=1e-04)
```

```
plot_GMM(gmm_multi, k = 2)
# error: Error: Aesthetics must be either length 1 or the same as the data (49): x
```

Unfortunately, I couldn't plot the results from the multivariate model. To get around this problem, I demonstrate that using one feature is also possible for clustering. Here, I selected to fit 2 models: one with

salary data and the other with total session length data.

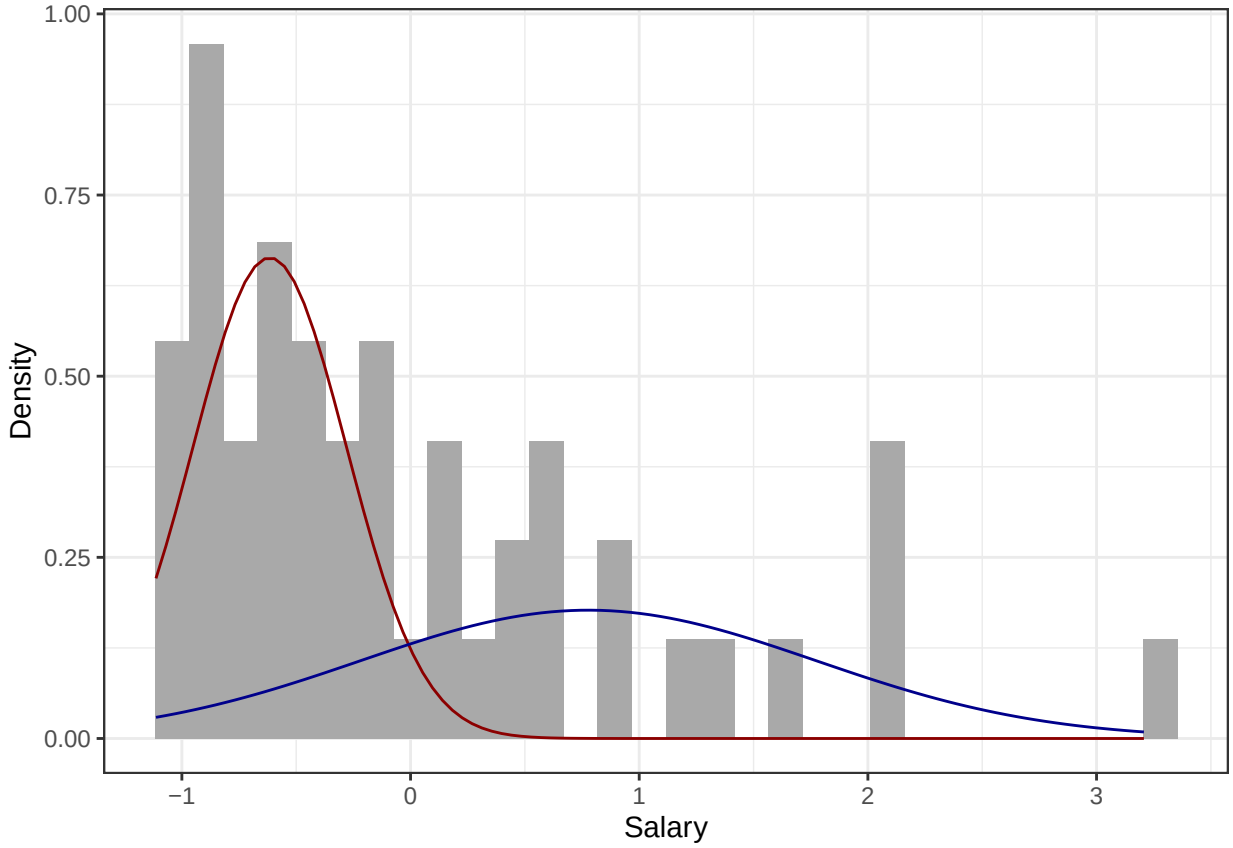
```
set.seed(12345)
sal_matrix <- as.matrix(SLPdf$std_salary_real)
gmm1 <- normalmixEM(sal_matrix,
                    k = 2)
```

```
## number of iterations= 46
```

Check the density plot. It does not look like the density plot fits the data well. This is because the red peak does not correspond to the real data density.

```
ggplot(data.frame(x = gmm1$x)) +
  geom_histogram(aes(x, ..density..), fill = "darkgray") +
  stat_function(geom = "line", fun = plot_mix_comps,
                args = list(gmm1$mu[1], gmm1$sigma[1], lam = gmm1$lambda[1]),
                colour = "darkred") +
  stat_function(geom = "line", fun = plot_mix_comps,
                args = list(gmm1$mu[2], gmm1$sigma[2], lam = gmm1$lambda[2]),
                colour = "darkblue") +
  xlab("Salary") +
  ylab("Density") +
  theme_bw()
```

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



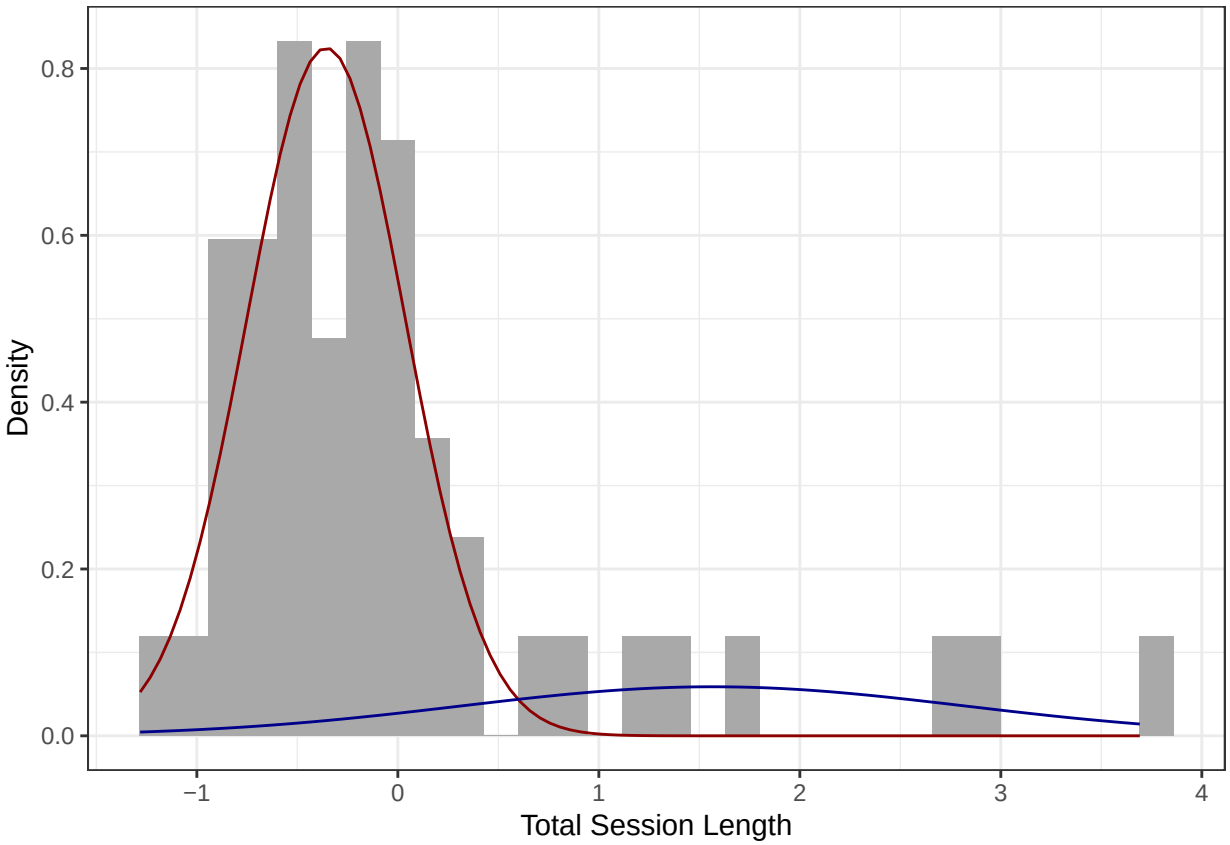

```
tlength_matrix <- as.matrix(SLPdf$std_t_slength)
gmm2 <- normalmixEM(tlength_matrix,
                    k = 2)
```

```
## number of iterations= 41
```

Check the density plot. This looks like the model fits the total session length better with two clusters. The distributions seem to correspond to the data well.

```
ggplot(data.frame(x = gmm2$x)) +
  geom_histogram(aes(x, ..density..), fill = "darkgray") +
  stat_function(geom = "line", fun = plot_mix_comps,
                args = list(gmm2$mu[1], gmm2$sigma[1], lam = gmm2$lambda[1]),
                colour = "darkred") +
  stat_function(geom = "line", fun = plot_mix_comps,
                args = list(gmm2$mu[2], gmm2$sigma[2], lam = gmm2$lambda[2]),
                colour = "darkblue") +
  xlab("Total Session Length") +
  ylab("Density") +
  theme_bw()
```

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



Part 7 - Visual Inspection of Outputs

Agglomerative Heirachical Clustering

Please see the plotted dendograms in part 4.

k-means

Here I show series of pairwise plots based on the k-means clustering algorithm's results. Two of the six plots suggest that there may be a problem with k-means since there are overlapping clusters. Since k-means uses hard-partitioning, it may not be as effective as GMM model for this type of data.

```
kc1 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("t_slength", "slength"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Regular Session Length")
```

```
kc2 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("t_slength", "salary_real"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Salary")
```

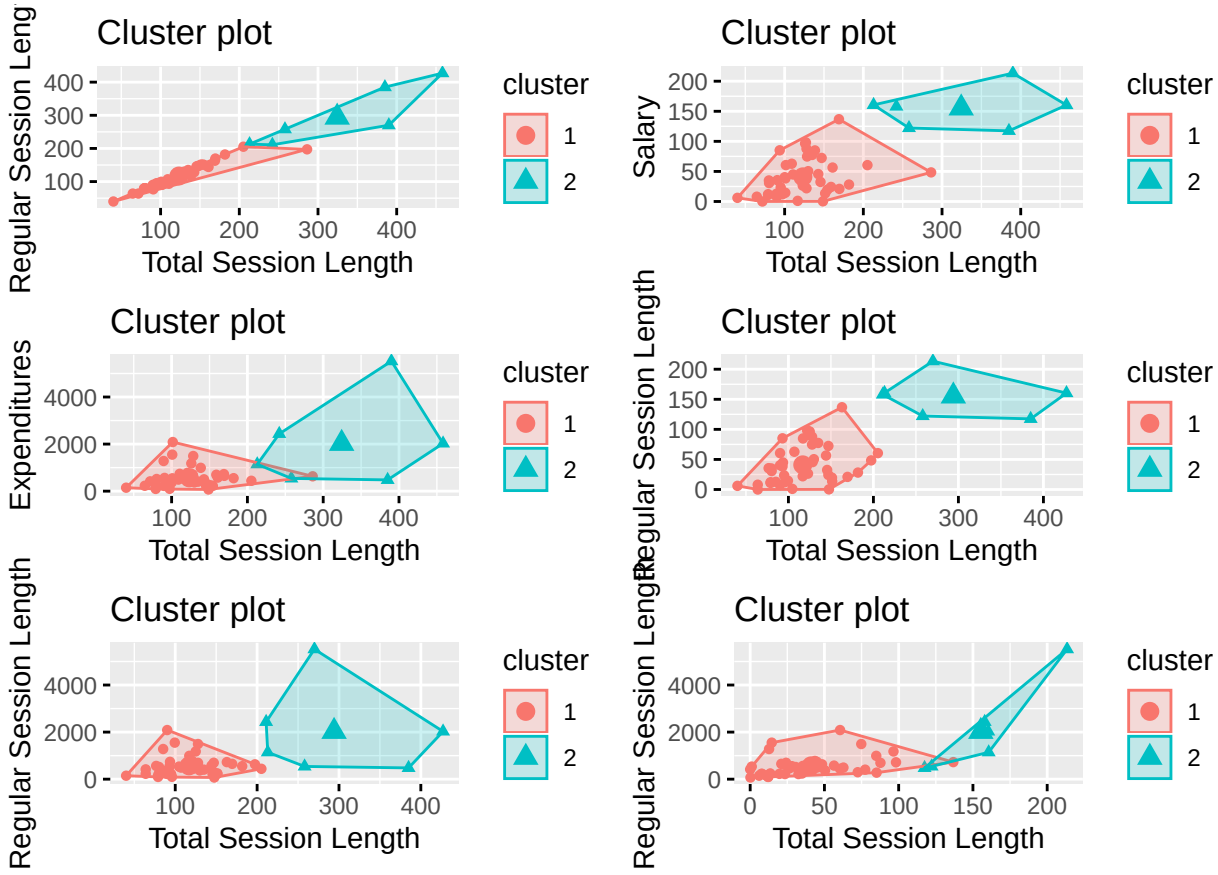
```
kc3 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("t_slength", "expend"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Expenditures")
```

```
kc4 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("slength", "salary_real"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Regular Session Length")
```

```
kc5 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("slength", "expend"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Regular Session Length")
```

```
kc6 <- fviz_cluster(kmeans, data = SLPdf, cluster = kmeans$cluster,
             choose.vars = c("salary_real", "expend"),
             geom = "point", stand = FALSE,
             ellipse = TRUE,
             xlab = "Total Session Length",
             ylab = "Regular Session Length")
```

```
grid.arrange(kc1, kc2, kc3, kc4, kc5, kc6)
```



Gaussian Mixture Model

Please see the density plots in part 6.

Part 8 - Validation Strategy

I select average silouette width because this validation strategy checks how well observations lie in a cluster across many values of k. The higher this value, the better configuration of the clusters.

```
SLPdf_matrix <- as.matrix(SLPdf[, 10:13])

internal_all <- clValid(SLPdf_matrix, 2:10,
                    clMethods = c("hierarchical", "kmeans", "model"),
                    validation = "internal"); summary(internal_all)
```

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

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

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

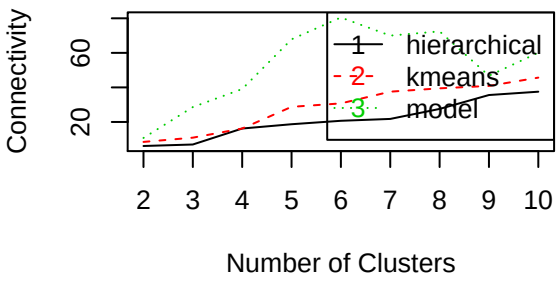
```
## The following object is masked from 'package:mixtools':
##
##     dmvnorm
```

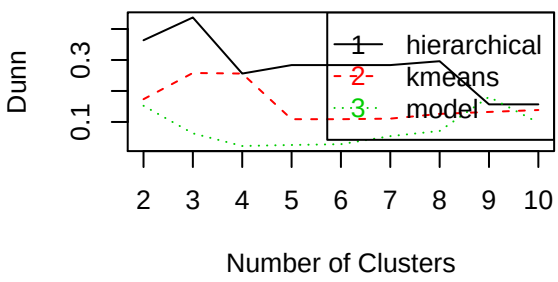
```
## The following object is masked from 'package:purrr':
##
##     map
```

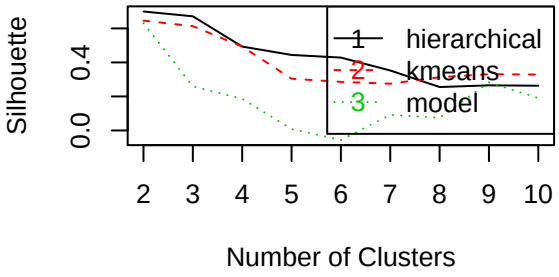
```
##
## Clustering Methods:
##  hierarchical kmeans model
##
## Cluster sizes:
##  2 3 4 5 6 7 8 9 10
##
## Validation Measures:
##                                  2       3       4       5       6       7       8       9      10
##
## hierarchical Connectivity   6.0869  6.9536 16.1885 18.6774 20.6774 21.7607 27.5476 35.5813 37.5147
##              Dunn           0.3637  0.4371  0.2562  0.2836  0.2836  0.2836  0.2960  0.1568  0.1568
##              Silhouette     0.6994  0.6711  0.4932  0.4440  0.4284  0.3525  0.2553  0.2652  0.2630
## kmeans       Connectivity   8.4460 10.8960 16.1885 28.7437 30.7437 37.5266 39.4552 40.8694 45.6623
##              Dunn           0.1735  0.2581  0.2562  0.1090  0.1090  0.1108  0.1260  0.1324  0.1386
##              Silhouette     0.6458  0.6131  0.4932  0.3042  0.2858  0.2750  0.3131  0.3307  0.3288
## model        Connectivity  10.7393 28.6119 39.0687 67.8401 80.4806 69.9774 72.4377 46.7254 60.0976
##              Dunn           0.1522  0.0633  0.0225  0.0258  0.0283  0.0543  0.0710  0.1810  0.0977
##              Silhouette     0.6314  0.2588  0.1861  0.0085 -0.0562  0.0917  0.0752  0.2831  0.1905
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 6.0869 hierarchical 2
## Dunn         0.4371 hierarchical 3
## Silhouette   0.6994 hierarchical 2
```

```
par(mfrow = c(2, 2))

plot(internal_all, legend = TRUE,
     type = "l",
     main = " ")
```








```
optimalScores(internal_all)
```

```
##                   Score       Method Clusters
## Connectivity 6.0869048 hierarchical        2
## Dunn         0.4371370 hierarchical        3
## Silhouette   0.6993513 hierarchical        2
```

Based on the Silouette validation strategy, the agglomerative hierachical clustering method seems to be the nest method for this data set with 2 clusters. As the code compares three different models across three validation strategies, we see that the hierchical model also scores best across all validation strategies.

Part 9 - Discussion on Validation Output

a) Key takeaway from the fit

The key takeaway is that 2 clusters seem to be the best way to group data points based on state legislative professionalism. One possible interpretation is that one group of states like Massachusetts, California, New York etc. appears to be more "professional" than the rest based on the session length, compensation and expenditures. However, if we were to include other measures such as gender and race, the groupings may result differently.

b) Optimal approach and the optimal value of k

Based on the validation strategies, the optimal approach is hierachical clustering and the optimal value of k is 2.

c) Reasons for selecting technically sub-optimal partitioning method

I can think of three reasons. The first reason is that the number of clusters yielded by the optimal method may go against the natural clusters that researchers know as the domain expert. Second, the k value produced

by the optimal method may not help us reduce the dimension of the data as we hope. For example, what if the k value from the algorithm is so large that it is almost equal to n kinds (e.g. states, gender, districts etc.) in the data set? In this scenario, the clustering may not be as useful as it is not reducing information for us. Hence, selecting a sub-optimal method that yields smaller number of clusters may, in fact, be more reasonable. The last scenario that I can imagine is when two or more validation strategies are used to evaluate the algorithms and they produce conflicting results. Researchers may use domain expertise to select a method that may not be optimal but comport with their understanding of the data/issues.