MACS 33002 PSET 4

Deniz Turkcapar

2/29/2020

Performing k-Means By Hand

In this first part of the problem set, your goal is to gain a deeper conceptual understanding of the iterative process of k-means, which operates by initializing random cluster assignments, then updates cluster centroids, then cluster assignments, and so on, until convergence, which is defined by no furtehr changes to cluster configurations (i.e., optimal clusters definined by minimum sums of squares *within* clusters, and maximum sums of squares *between* clusters.

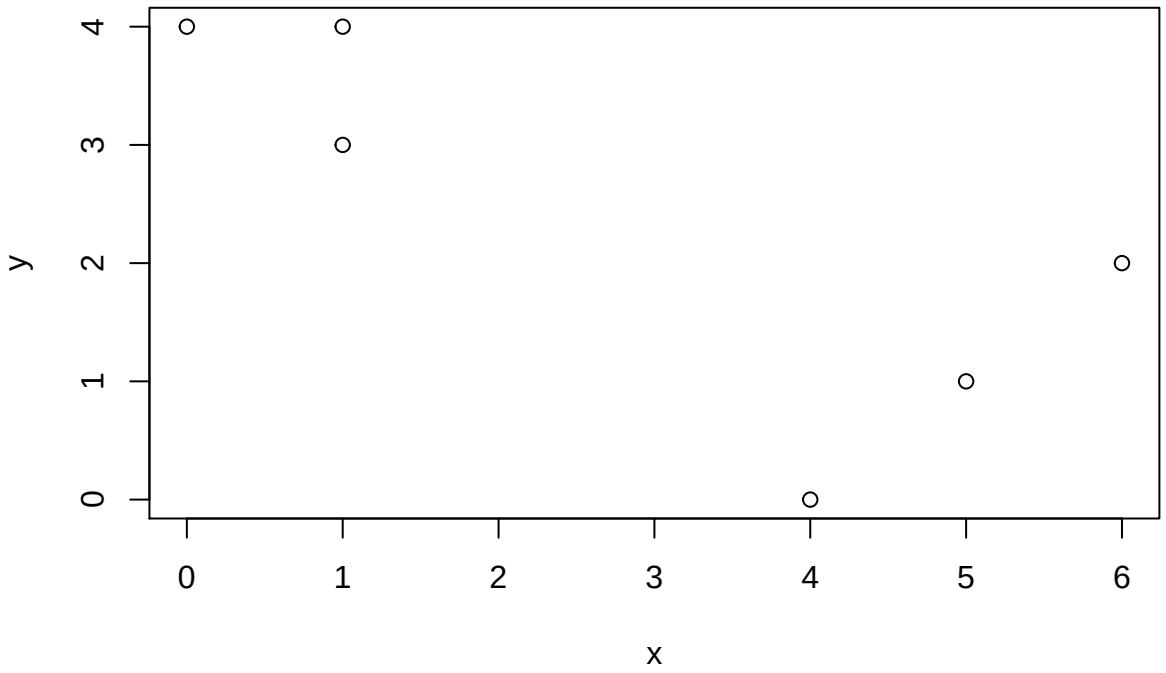
In short, then, by answering each of the following questions, you will be performing k-means clustering "by hand" to see and demonstrate this iterative process. Your simulated data includes `n = 6` observations and `p = 2` features, and you should set the number of clusters, `k`, equal to two (i.e., you are hunting for 2 clusters within these data). I will get you started with the observations in the set. Run the following line to create your simulated data:

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

1. (5 points) Plot the observations.

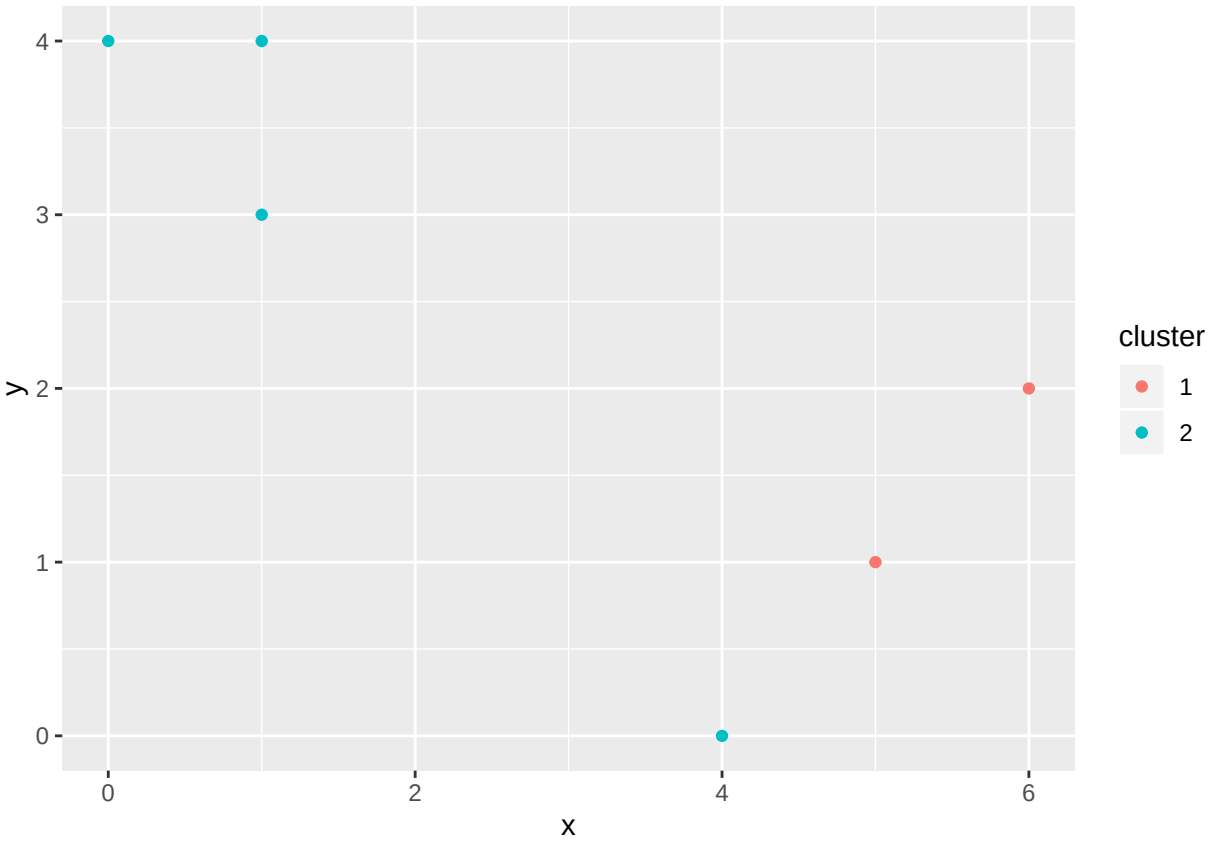
I will give the columns random names (x and y) so that the code and graphics are easier to read. I included the values that we were specified to use above.

```
df <- tibble(x = c(1, 1, 0, 5, 6, 4),
             y = c(4, 3, 4, 1, 2, 0))
plot(df)
```



2. (5 points) Randomly assign a cluster label to each observation. Report the cluster labels for each observation *and* plot the results with a different color for each cluster (*remember to set your seed first*).

```
set.seed(12)
k <- 2 # Setting k = 2 as given in the prompt
df <- df %>%
  mutate(cluster = as.factor(sample(1:k, length(x), replace = TRUE)))

ggplot(df, aes(x = x, y = y, color=cluster)) + geom_point() + theme_gray()
```




```
cluster.label <- sample(2, length(df$x), replace = T)
cluster.label
```

```
[1] 1 1 2 2 2 1
```

3. (10 points) Compute the centroid for each cluster. Computing the centroid for each cluster:

```
df %>%
  group_by(cluster) %>%
  summarize_all(~mean(.))
```

```
# A tibble: 2 x 3
  cluster     x     y
  <fct>   <dbl> <dbl>
1 1         5.5  1.5
2 2         1.5  2.75
```

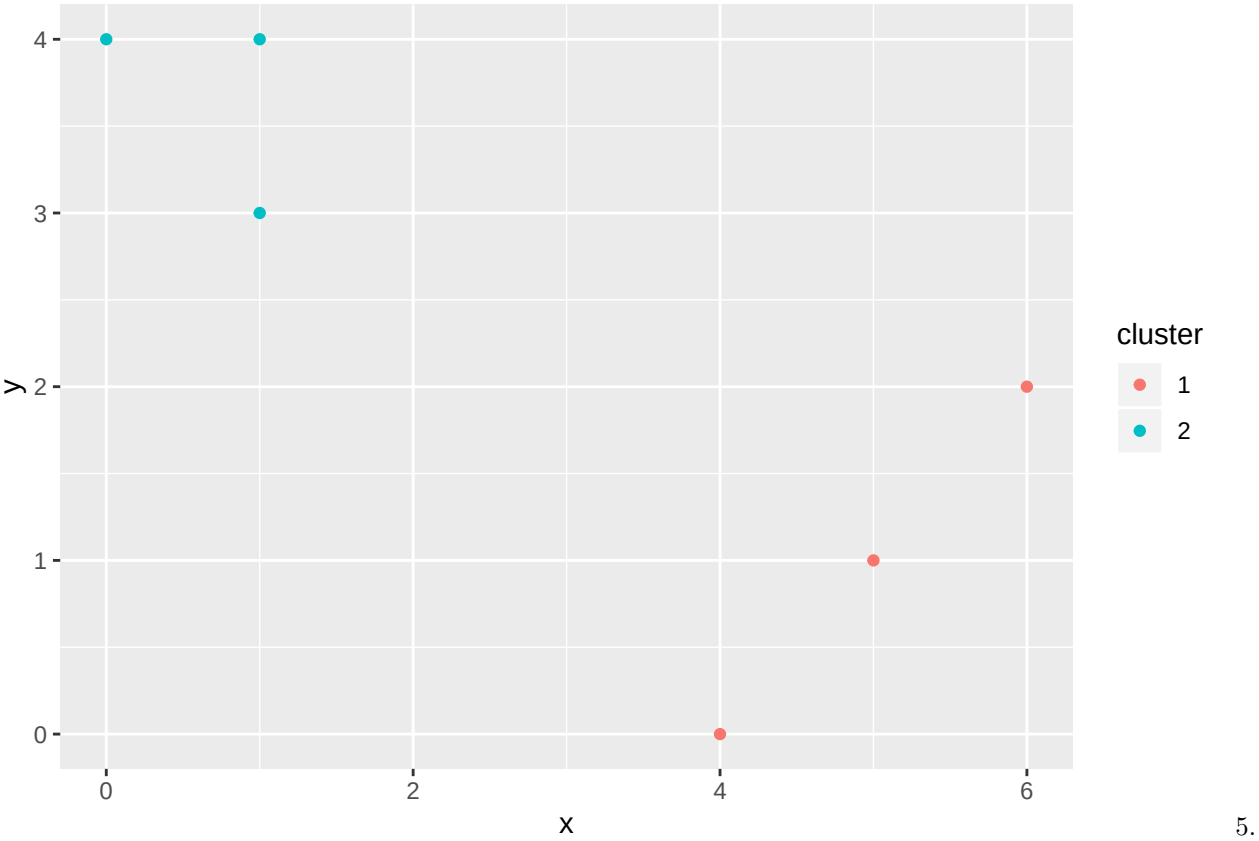
4. (10 points) Assign each observation to the centroid to which it is closest, in terms of Euclidean distance. Report the cluster labels for each observation.

```
update_cluster <- function(update_operations){
  re_clustered <- update_operations %>%
    group_by(cluster) %>%
    summarize_all(~mean(.)) %>%
    pivot_wider(names_from = "cluster", values_from = c("x", "y"))

  assign_dist <- update_operations%>%
    cbind(re_clustered) %>%
    mutate(first_dist = (x - x_1)^2 + (y - y_1)^2,
           second_dist = (x - x_2)^2 + (y - y_2)^2,
```

```
          cluster = as.factor(if_else(first_dist < second_dist, 1, 2)))

  assign_dist %>% select(x,y,cluster) %>% return()
}

updated_cluster <-update_cluster(df)
updated_cluster
```

```
  x y cluster
1 1 4       2
2 1 3       2
3 0 4       2
4 5 1       1
5 6 2       1
6 4 0       1
```

```
ggplot(updated_cluster, aes(x = x, y = y, color = cluster)) + geom_point() + theme_gray()
```



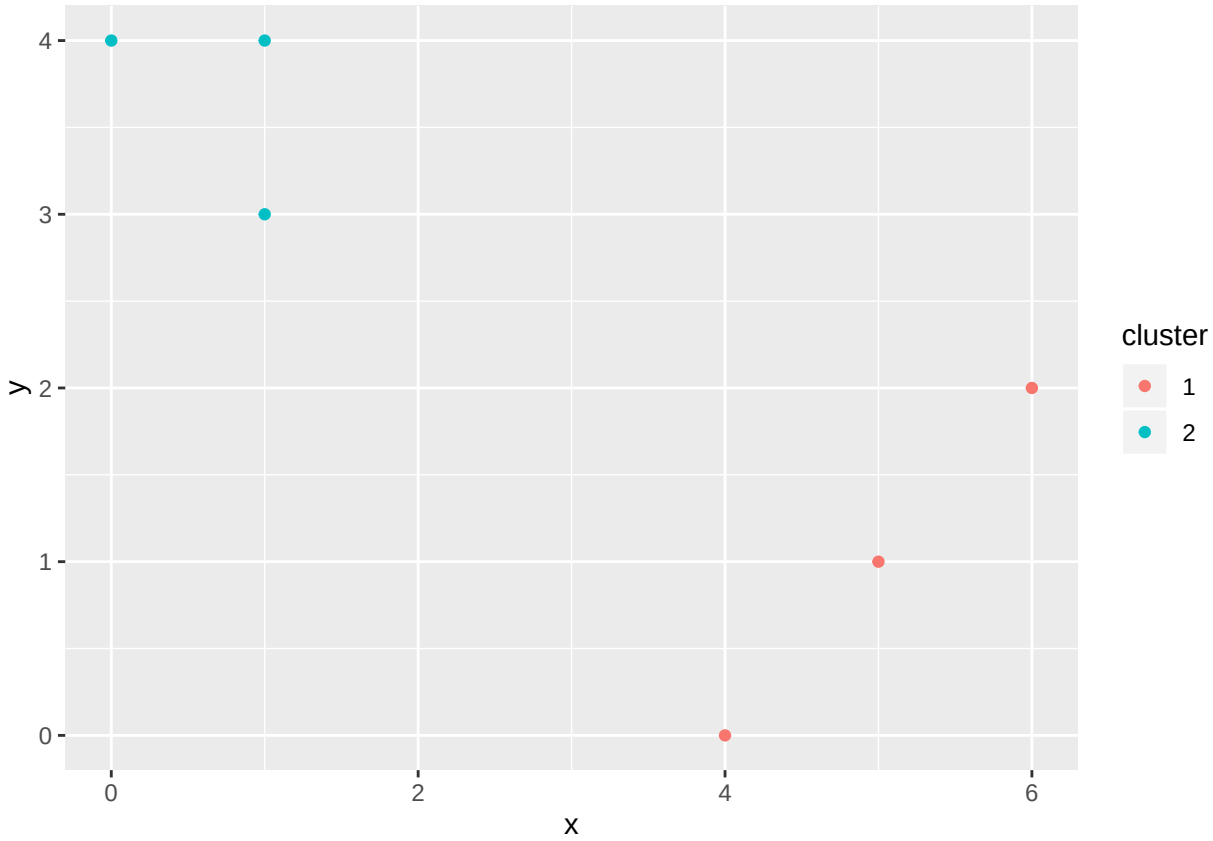
5. (5 points) Repeat (3) and (4) until the answers/clusters stop changing.

```
different = TRUE

while(different){
  original_cluster <- df$cluster
  df <- update_cluster(df)
  different <- all(original_cluster != df$cluster)
}
```

6. (10 points) 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.

```
ggplot(df, aes(x = x, y = y, color = cluster)) + geom_point() + theme_gray()
```



Clustering State Legislative Professionalism

1. Load the state legislative professionalism data. See the codebook (or above) for further reference.

```
#load("Data and Codebook/legprof-components.v1.0.RData")
load(file ="Data and Codebook/legprof-components.v1.0.RData")
leg_prof_data <- x
```

2. (5 points) Munge the data:
    a. select only the continuous features that should capture a state legislature's level of "professionalism" (session length (total and regular), salary, and expenditures);
    b. restrict the data to only include the 2009/10 legislative session for consistency;
    c. omit all missing values;
    d. standardize the input features;
    e. and anything else you think necessary to get this subset of data into workable form (hint: consider storing the state names as a separate object to be used in plotting later)

```
# take a look at the data to see how it's constructed(the variables, length, etc.)
library(skimr)
```

```
Attaching package: 'skimr'
```

```
The following object is masked from 'package:mosaic':

    n_missing
```

```r
skim(leg_prof_data)
```

Table 1: Data summary

Name	leg_prof_data
Number of rows	950
Number of columns	11
Column type frequency:	
AsIs	2
factor	1
numeric	8
Group variables	None

Variable type: AsIs

skim_variable	n_missing	complete_rate	n_unique	min_length	max_length
stateabv	0	1	50	1	1
state	0	1	50	1	1

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
sessid	0	1	TRUE	19	197: 50, 197: 50, 197: 50, 197: 50

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
fips	0	1.00	29.32	15.63	1.00	17.00	29.50	42.00	56.00	
t_slength	61	0.94	147.60	86.09	36.00	91.00	128.51	171.00	549.54	
slength	61	0.94	136.39	81.18	36.00	85.20	120.00	158.00	521.85	
salary_real	5	0.99	55.82	47.11	0.00	20.11	41.96	80.08	254.94	
expend	5	0.99	599.51	724.19	40.14	219.93	395.10	650.29	5523.10	
year	0	1.00	1992.09	10.96	1974.00	1982.00	1992.00	2002.00	2011.00	
mds1	61	0.94	0.00	1.48	-1.85	-0.93	-0.31	0.41	8.56	
mds2	61	0.94	0.00	0.72	-3.14	-0.34	0.09	0.30	3.35	

```r
# Munge the data
munged_data <- leg_prof_data %>%
  filter(sessid == "2009/10") %>% # part b
  mutate(rowname = stateabv) %>% # part e
  column_to_rownames() %>% # part e
  select(t_slength, slength, salary_real, expend) %>% # part a
  drop_na() %>% # part c
  scale() # part d

# part e
```

```r
states <- leg_prof_data %>%
  filter(sessid == "2009/10") %>%
  select(state, t_slength, slength, salary_real, expend) %>%
  na.omit() %>%
  select(state)

states
```

```
           state
1        Alabama
2         Alaska
3        Arizona
4       Arkansas
5     California
6       Colorado
7    Connecticut
8       Delaware
9        Florida
10       Georgia
11        Hawaii
12         Idaho
13      Illinois
14       Indiana
15          Iowa
16        Kansas
17      Kentucky
18     Louisiana
19         Maine
20      Maryland
21 Massachusetts
22      Michigan
23     Minnesota
24   Mississippi
25      Missouri
26       Montana
27      Nebraska
28        Nevada
29 New Hampshire
30    New Jersey
31    New Mexico
32      New York
33 North Carolina
34  North Dakota
35          Ohio
36      Oklahoma
37        Oregon
38  Pennsylvania
39  Rhode Island
40 South Carolina
41  South Dakota
42     Tennessee
43         Texas
44          Utah
45       Vermont
```

```
46       Virginia
47     Washington
48  West Virginia
50        Wyoming
```

3. (5 points) Diagnose clusterability in any way you'd prefer (e.g., sparse sampling, ODI, etc.); display the results and discuss the likelihood that natural, non-random structure exist in these data. *Hint:* We didn't cover how to do this R in class, but consider `dissplot()` from the `seriation` package, the `factoextra` package, and others for calculating, presenting, and exploring the clusterability of some feature space.

```
library(seriation)
```

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


Attaching package: 'seriation'

The following object is masked from 'package:modelr':

    permute

The following object is masked from 'package:lattice':

    panel.lines
```

```
library(factoextra)
```

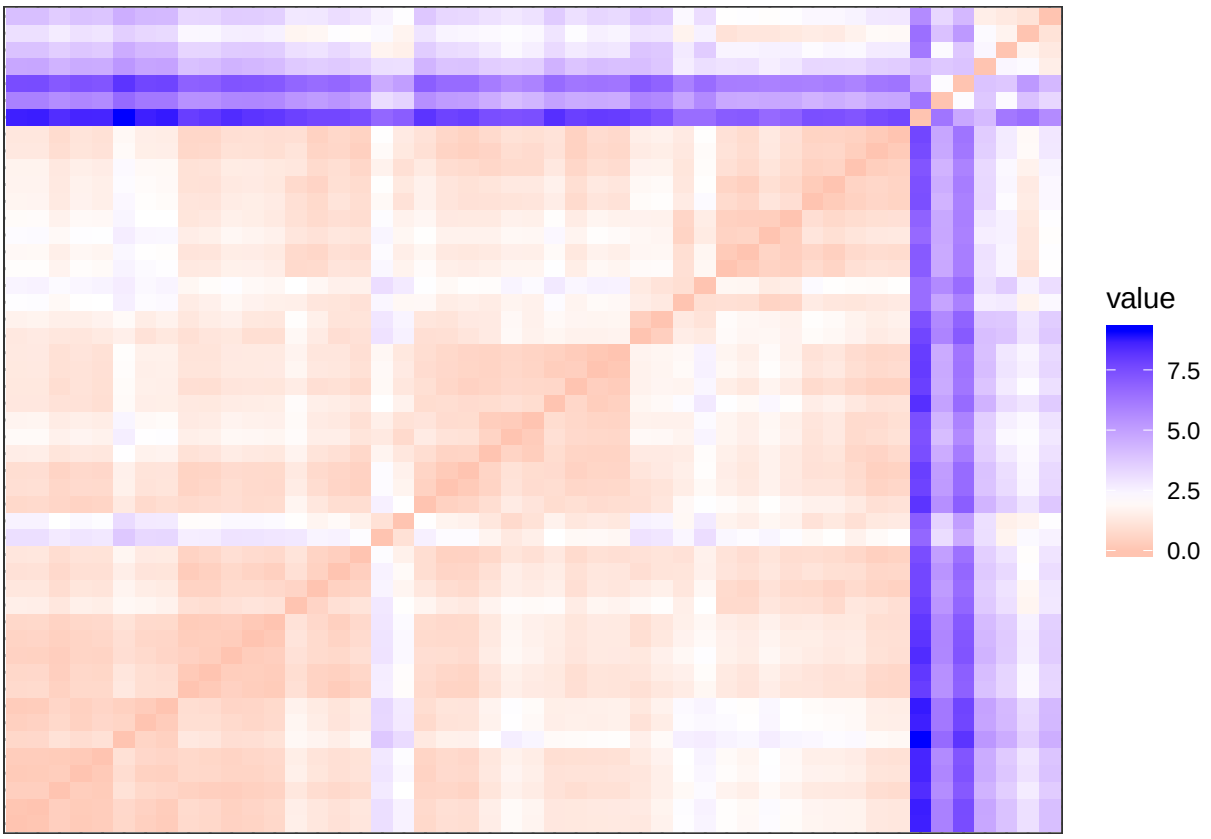
```
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
length(munged_data)
```

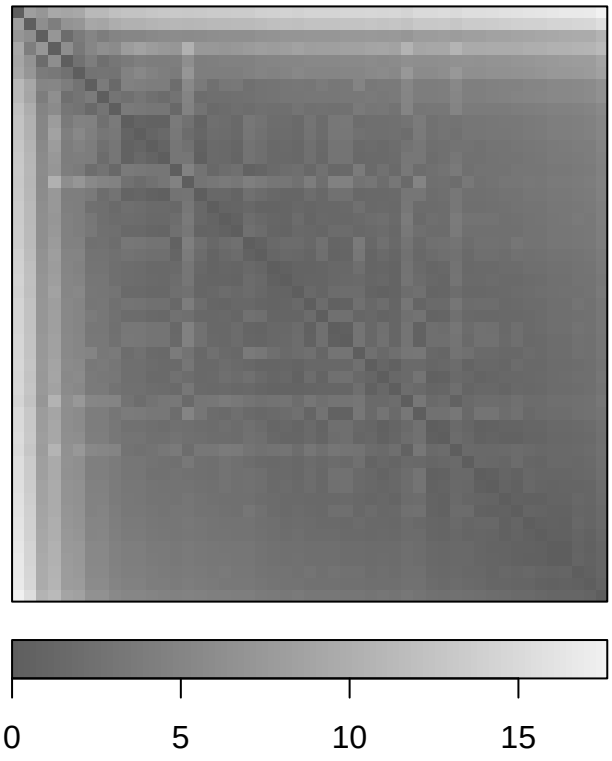
```
[1] 196
```

```
get_clust_tendency(munged_data, 38)
```

```
$hopkins_stat
[1] 0.8386

$plot
```




```
get_dist_clust <- dist(munged_data, method = "manhattan")
dissplot(get_dist_clust)
```

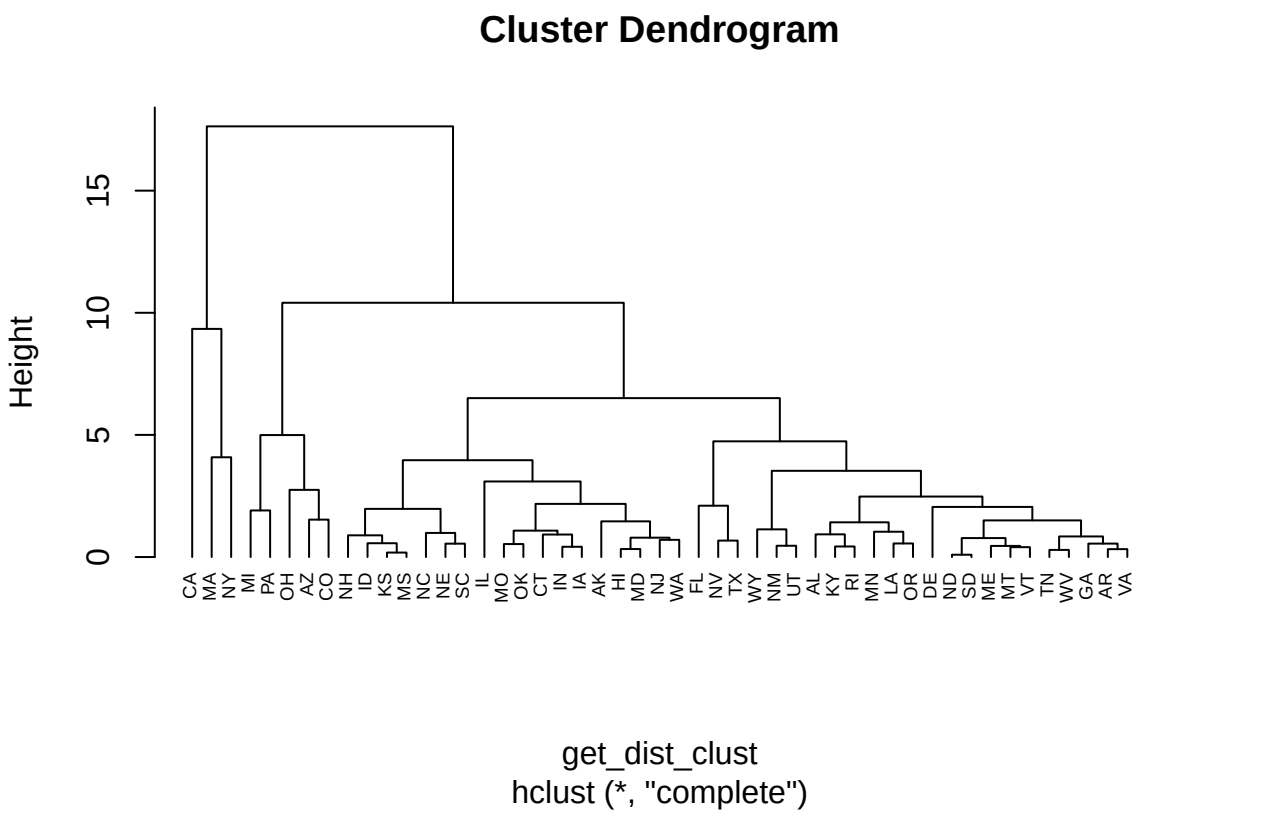


The hopkins statistic is approximately 0.84, which is a great sign. The closer the hopkins statistic is to 1, the better that there are observable similarities between some observations. Therefore, we can say that the observations that we have are clusterable. Looking at the colors in the graph, we can observe that there are a few red and white squares that share a border, which shows that there is high similarity between some observations. The high similarity suggests that the observations are clusterable with success.

In the ODI, we can observe a few squares but the unclear structure and positioning of them make it hard to see the success of clustering. The clearest ones can be observed in the 45 degree line, which is a good sign of clusterability.

4. (5 points) Fit an **agglomerative hierarchical** clustering algorithm using any linkage method you prefer, to these data and present the results. Give a quick, high level summary of the output and general patterns.

```
cluster_dendrogram <- hclust(get_dist_clust,
                        method = "complete")
plot(cluster_dendrogram, cex = 0.6, hang = -1)
```



From the Cluster Dendrogram above, we can observe that Kansas and Mississippi are the most similar to each other, and North Dakota and South Dakota are the most similar to each other. Thus, we can say that some neighboring states are more similar to each other than the farther ones. The height of the dendrogram measures the closeness of either individual data points or clusters. The biggest dissimilarity observed in the data is the California state, which makes sense considering the extra emphasis on freedom that California historically emphasizes in their state. The maximum height of the dendrogram is approximately 17, whereas the minimum height is approximately 0.4, as observed from the similarity of the states North Dakota and South Dakota.

5. (5 points) Fit a **k-means** algorithm to these data and present the results. Give a quick, high level summary of the output and general patterns. Initialize the algorithm at `k = 2`, and then check this assumption in the validation questions below.

```r
set.seed(1234)
kmeans_fit <- kmeans(munged_data, centers = 2, nstart = 15)
kmeans_fit
```

```
K-means clustering with 2 clusters of sizes 43, 6

Cluster means:
  t_slength slength salary_real  expend
1    -0.293 -0.2932     -0.2834 -0.2048
2     2.100  2.1015      2.0308  1.4677

Clustering vector:
AL AK AZ AR CA CO CT DE FL GA HI ID IL IN IA KS KY LA ME MD MA MI MN
 1  1  1  1  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  2  1
MS MO MT NE NV NH NJ NM NY NC ND OH OK OR PA RI SC SD TN TX UT VT VA
 1  1  1  1  1  1  1  1  2  1  1  2  1  1  2  1  1  1  1  1  1  1
WA WV WY
 1  1  1

Within cluster sum of squares by cluster:
[1] 48.37 40.36
 (between_SS / total_SS =  53.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
```

```r
str(kmeans_fit)
```

```
List of 9
 $ cluster     : Named int [1:49] 1 1 1 1 2 1 1 1 1 1 ...
  ..- attr(*, "names")= chr [1:49] "AL" "AK" "AZ" "AR" ...
 $ centers     : num [1:2, 1:4] -0.293 2.1 -0.293 2.101 -0.283 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:4] "t_slength" "slength" "salary_real" "expend"
 $ totss       : num 192
 $ withinss    : num [1:2] 48.4 40.4
 $ tot.withinss: num 88.7
 $ betweenss   : num 103
 $ size        : int [1:2] 43 6
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

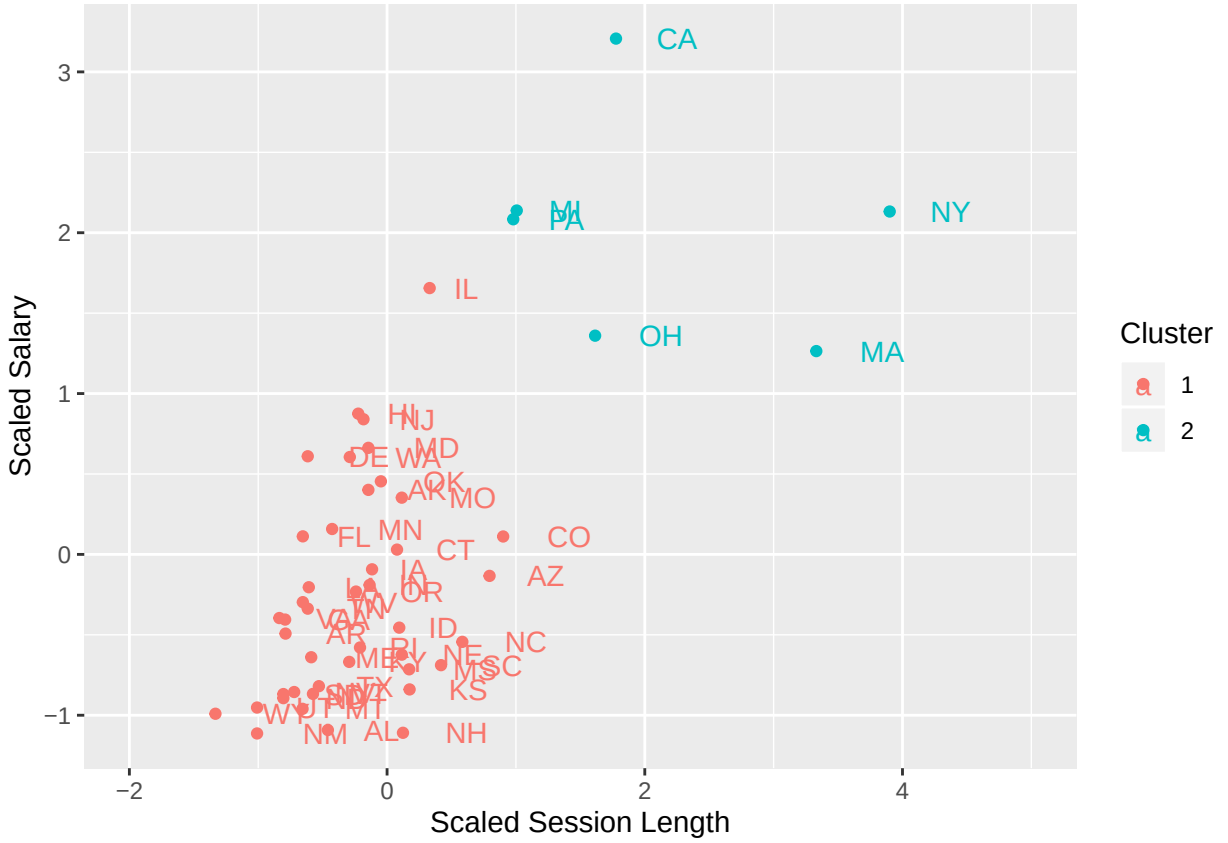
```r
col_1 <- states
col_2 <- as.data.frame(kmeans_fit$cluster)
cbind(col_1, col_2)
```

```
            state kmeans_fit$cluster
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
50        Wyoming                 1
```

```r
# Visualizing the clusters:
munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
  ggplot(aes(x = slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
```

```
  labs(x = "Scaled Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))
```



Our k-means clustering algorithm put 43 states in cluster 2 and the remaining 6 states in cluster 1. We may need to increase our "k" in order to obtain results that are more sensitive to certain dissimilarities that `k = 2` cannot account for. Overall, the k-means algorithm returned a similar result to the Cluster Dendrogram. Therefore, we can reaffirm some stability in our clustering results, and that clustering is applicable for the given observations as we can see certain dissimilarities and similarities.

6. (5 points) Fit a **Gaussian mixture model via the EM algorithm** to these data and present the results. Give a quick, high level summary of the output and general patterns. Initialize the algorithm at `k = 2`, and then check this assumption in the validation questions below.

```
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-051877
```

```
set.seed(123)
gaussian_model <-mvnormalmixEM(munged_data, k = 2)
```

```
number of iterations= 25
```

```
gaussian_model$sigma
```

```
[[1]]
        [,1]    [,2]   [,3]    [,4]
[1,] 0.24528 0.27955 0.2097 0.03105
```

```
[2,] 0.27955 0.32492 0.2376 0.02479
[3,] 0.20967 0.23755 0.5807 0.13661
[4,] 0.03105 0.02479 0.1366 0.23482

[[2]]
       [,1]    [,2]   [,3]    [,4]
[1,] 0.8556  1.0197 0.3980  0.3509
[2,] 1.0197  1.5611 0.3427 -0.3956
[3,] 0.3980  0.3427 1.2313  1.9834
[4,] 0.3509 -0.3956 1.9834  4.3511
```

```
gaussian_model$mu
```

```
[[1]]
[1] -0.2763 -0.2451 -0.1944 -0.1921

[[2]]
[1] 2.432 2.157 1.711 1.690
```

```
gaussian_model$lambda
```

```
[1] 0.898 0.102
```

```
gaussian_model$posterior
```

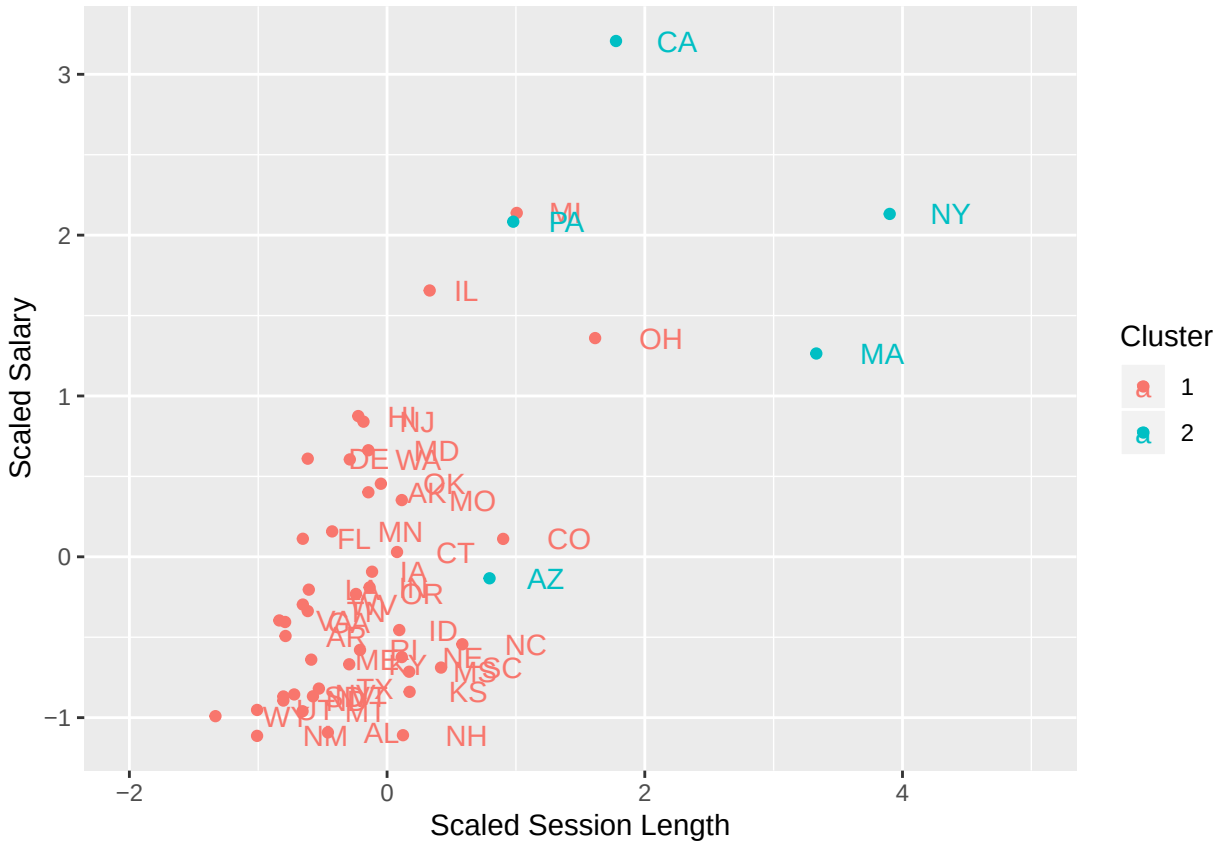
```
          comp.1     comp.2
 [1,]  1.000e+00  0.000e+00
 [2,]  1.000e+00  0.000e+00
 [3,]  3.114e-52  1.000e+00
 [4,]  1.000e+00  0.000e+00
 [5,] 1.823e-100  1.000e+00
 [6,]  1.000e+00 3.241e-261
 [7,]  1.000e+00 1.991e-182
 [8,]  1.000e+00  7.564e-06
 [9,]  1.000e+00  0.000e+00
[10,]  1.000e+00  0.000e+00
[11,]  1.000e+00  2.246e-09
[12,]  1.000e+00  0.000e+00
[13,]  1.000e+00 1.315e-119
[14,]  1.000e+00 2.282e-278
[15,]  1.000e+00 2.194e-284
[16,]  1.000e+00  0.000e+00
[17,]  1.000e+00  0.000e+00
[18,]  1.000e+00  0.000e+00
[19,]  1.000e+00  0.000e+00
[20,]  1.000e+00  1.817e-58
[21,]  5.917e-08  1.000e+00
[22,]  1.000e+00 8.791e-120
[23,]  1.000e+00 1.241e-165
[24,]  1.000e+00  0.000e+00
[25,]  1.000e+00  6.267e-69
[26,]  1.000e+00  0.000e+00
[27,]  1.000e+00  0.000e+00
[28,]  1.000e+00  0.000e+00
[29,]  1.000e+00  0.000e+00
[30,]  1.000e+00 2.491e-113
```

```
[31,]  1.000e+00  0.000e+00
[32,]  5.823e-19  1.000e+00
[33,]  1.000e+00  0.000e+00
[34,]  1.000e+00  0.000e+00
[35,]  1.000e+00  2.458e-14
[36,]  1.000e+00  1.576e-58
[37,]  1.000e+00  0.000e+00
[38,]  7.098e-07  1.000e+00
[39,]  1.000e+00  0.000e+00
[40,]  1.000e+00  0.000e+00
[41,]  1.000e+00  0.000e+00
[42,]  1.000e+00  0.000e+00
[43,]  1.000e+00  0.000e+00
[44,]  1.000e+00  0.000e+00
[45,]  1.000e+00  0.000e+00
[46,]  1.000e+00  0.000e+00
[47,]  1.000e+00  2.543e-46
[48,]  1.000e+00 1.199e-284
[49,]  1.000e+00  0.000e+00
```

```r
# We can assign to their appropriate bins according to the treshold of whichcluster they will most like

assign <- data.frame(cbind(gaussian_model$x, gaussian_model$posterior))
assign$component <- ifelse(assign$comp.1 > 0.5, 1, 2)
table(assign$component)
```

```
 1  2
44  5
```

```r
# We can also visualize this:
munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  labs(x = "Scaled Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))
```



7. (15 points) Compare output of all in visually useful, simple ways (e.g., present the dendrogram, plot by state cluster assignment across two features like salary and expenditures, etc.). There should be several plots of comparison and output.

We can observe that the **Gaussian mixture model via the EM algorithm** put 44 states in cluster 1 and the reest of 5 states in cluster 2, which is a different output than the k-means clustering algorithm. I ran the code above multiple times and each time I changed the seed, I observed that the Gaussian mixture model did worse than k-means algorithm most of the time. A useful way to compare all 3 ways of clustering could be plotting by state cluster assignment across two examplary features, like salary and session length, etc. The comparison plots are below:

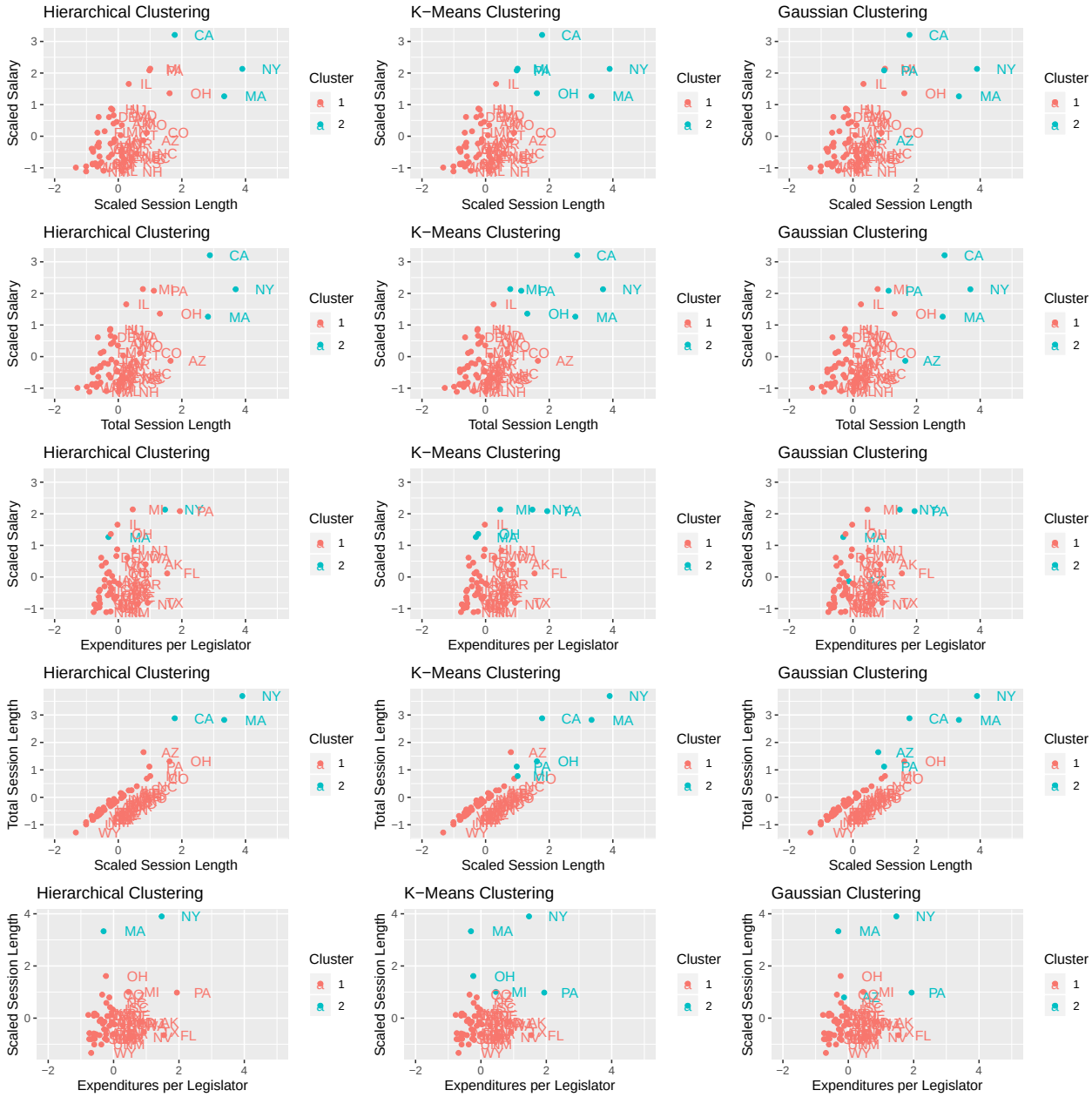
```
# Hierarchical Clustering x = "Scaled Session Length", y = "Scaled Salary"
options(repr.plot.width = 4, repr.plot.height = 3)
plot1 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(cutree(cluster_dendrogram, k = 2))) %>%
  ggplot(aes(x = slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Hierarchical Clustering") +
  labs(x = "Scaled Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# k-mean x = "Scaled Session Length", y = "Scaled Salary"
plot2 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
```

```
  ggplot(aes(x = slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("K-Means Clustering") +
  labs(x = "Scaled Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Gaussian x = "Scaled Session Length", y = "Scaled Salary"
plot3 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Gaussian Clustering") +
  labs(x = "Scaled Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))


# Hierarchical Clustering x = "Total Session Length", y = "Scaled Salary"
plot4 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(cutree(cluster_dendrogram, k = 2))) %>%
  ggplot(aes(x = t_slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Hierarchical Clustering") +
  labs(x = "Total Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# k-mean x = "Total Session Length", y = "Scaled Salary"
plot5 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
  ggplot(aes(x = t_slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("K-Means Clustering") +
  labs(x = "Total Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Gaussian x = "Total Session Length", y = "Scaled Salary"
plot6 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = t_slength, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Gaussian Clustering") +
```

```
  labs(x = "Total Session Length", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))


# Hierarchical Clustering x = "Expenditures per Legislator", y = "Scaled Salary"
plot7 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(cutree(cluster_dendrogram, k = 2))) %>%
  ggplot(aes(x = expend, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Hierarchical Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# k-mean x = "Expenditures per Legislator", y = "Scaled Salary"
plot8 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
  ggplot(aes(x = expend, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("K-Means Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Gaussian x = "Expenditures per Legislator", y = "Scaled Salary"
plot9 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = expend, y = salary_real, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Gaussian Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Salary", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Hierarchical Clustering x = "Scaled Session Length", y = "Total Session Length"
plot10 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(cutree(cluster_dendrogram, k = 2))) %>%
  ggplot(aes(x = slength, y = t_slength, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Hierarchical Clustering") +
  labs(x = "Scaled Session Length", y = "Total Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))
```

```r
# k-mean x = "Scaled Session Length", y = "Total Session Length"
plot11 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
  ggplot(aes(x = slength, y = t_slength, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("K-Means Clustering") +
  labs(x = "Scaled Session Length", y = "Total Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Gaussian x = "Scaled Session Length", y = "Total Session Length"
plot12 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = slength, y = t_slength, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Gaussian Clustering") +
  labs(x = "Scaled Session Length", y = "Total Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# K-Means clustering x = "Expenditures per Legislator", y = "Scaled Session Length"
plot13 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(cutree(cluster_dendrogram, k = 2))) %>%
  ggplot(aes(x = expend, y = slength, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Hierarchical Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# k-mean x = "Expenditures per Legislator", y = "Scaled Session Length"
plot14 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(kmeans_fit$cluster)) %>%
  ggplot(aes(x = expend, y = slength, color = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("K-Means Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

# Gaussian x = "Expenditures per Legislator", y = "Scaled Session Length"
plot15 <- munged_data %>%
  as_tibble() %>%
  mutate(cluster= as.factor(apply(gaussian_model$posterior, 1, which.max))) %>%
  ggplot(aes(x = expend, y = slength, color = cluster)) +
```

```
  geom_point() +
  geom_text(aes(label = rownames(munged_data)), hjust = - 1) +
  ggtitle("Gaussian Clustering") +
  labs(x = "Expenditures per Legislator", y = "Scaled Session Length", color = "Cluster") +
  theme_gray() +
  coord_cartesian(xlim = c(-2, 5))

grid.arrange(plot1, plot2, plot3, plot4, plot5, plot6, plot7, plot8, plot9, plot10, plot11, plot12, plo
```



On average, there aren't drastic differences between the outputs of the three different clustering methods. We can observe that all three clustering algorithms are differentiating between the clusters in expected ways. The k-means clustering and the hierarchical clustering methods seem to be clearer on average about the distinctions between the two clusters, although this would not be a significant difference for our analyses.

8. (5 points) Select a *single* validation strategy (e.g., compactness via min(WSS), average silhouette width, etc.), and calculate for all three algorithms. Display and compare your results for all three algorithms you fit (hierarchical, k-means, GMM). Hint: Here again, we didn't cover this in R in class, but think about using the `clValid` package, though there are many other packages and ways to validate cluster patterns across iterations.

```
library(clValid)
```

```
Loading required package: cluster
```

```
library(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
```

```
munged_data_mat <- as.matrix(munged_data)
clvalid <- clValid(munged_data_mat, 2:5, validation = "internal", clMethods = c("model", "kmeans", "hie
summary(clvalid)
```

```
Clustering Methods:
 model kmeans hierarchical

Cluster sizes:
 2 3 4 5

Validation Measures:
                                 2      3      4      5

model        Connectivity  10.739 28.612 39.069 67.840
             Dunn           0.152  0.063  0.022  0.026
             Silhouette     0.631  0.259  0.186  0.008
kmeans       Connectivity   8.446 10.896 16.188 28.744
             Dunn           0.173  0.258  0.256  0.109
             Silhouette     0.646  0.613  0.493  0.304
hierarchical Connectivity   6.087  6.954 16.188 18.677
             Dunn           0.364  0.437  0.256  0.284
             Silhouette     0.699  0.671  0.493  0.444

Optimal Scores:

             Score Method       Clusters
Connectivity 6.087 hierarchical 2
Dunn         0.437 hierarchical 3
Silhouette   0.699 hierarchical 2
```

From the calculations above, it looks like hierarchical clustering performed the best on all 3 dimensions -

Connectivity, Dunn, and Silhouette. The success of a clustering method is determined by high Silhouette width and Dunn index, and low Connectivity. The hierarchical clustering method fits in these categories well, and therefore it is chosen as the best performing clustering method.

9. (10 points) Discuss the validation output, e.g.,

- What can you take away from the fit?
- Which approach is optimal? And optimal at what value of k?
- What are reasons you could imagine selecting a technically "sub-optimal" clustering method, regardless of the validation statistics?

Looking at the fit, we can see that the gaussian model outputted some clusters that did not have as distinct differences as the ones we had with k-means and hierarchical clustering. This could be a concern if we want consistency and some clearer boundaries in our output. The Gaussian model seems to be performing significantly worse on the Connectivity aspect observing the high values of Connectivity it got for each value of k. The k-means and the hierarchical clustering is comparable on the most on the Silhouette aspect, however, the hierarchical clustering still performs better on all 3 aspects of comparison (Silhouette, Dunn, and Connectivity).

I would choose hierarchical clustering as the optimal fit for this data as observed from the validation output. The hierarchical clustering performs better than the rest of the clustering methods in all aspects - Dunn, Silhouette, and Connectivity. Even though k-means and hierarchical clustering seems to be doing comparatively pretty well, hierarchical clustering is able to produce more reliable results than the k-means clustering method. The validation output uses the values of $k = 2, 3, 4, 5$ so measure the success of the clustering methods given the respective sizes of k. Looking at the Connectivity and Silhouette aspects, we can say that the $k = 2$ performs the best for Hierarchical Clustering. If we want to optimize for the Dunn Index, then it would make sense to use $k = 3$ for our Hierarchical Clustering model.

I can imagine choosing a "sub-optimal" clustering method for the sake of consistency of results and the speed of the algorithm. Speed is something that we definitely want to keep in mind in situations such as an algorithm that needs to compute real time results in a production environment in a company. In such a case, an algorithm with loads of computations that slow down the system would be looked down upon since it would be hard to keep such an infrastructure up to speed, and it can be a potential a loophole that can create crashes in the production infrastructure of the company. Regarding the computation, we can say that the Gaussian model is less optimal, even though we know that the Gaussian method takes into account many more aspects such as the probability of belonging to one cluster. This is because the computation for this method takes far longer than the other clustering methods we compared it to (k-means and hierarchical clustering). In some large amounts of data, as mentioned before, the computational slowness might become an issue. Additionally, consistency is a problem for the Gaussian method.