Intro to Machine learning

Problem Set 4

Jingming Wei(Mark)

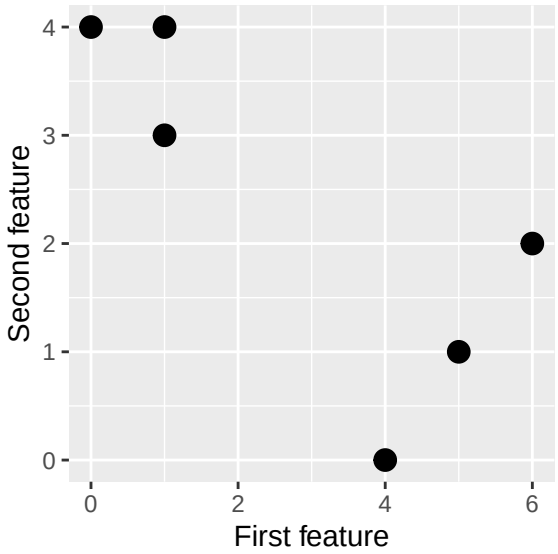
2/17/2020

Q1 Performing k-Means By Hand

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

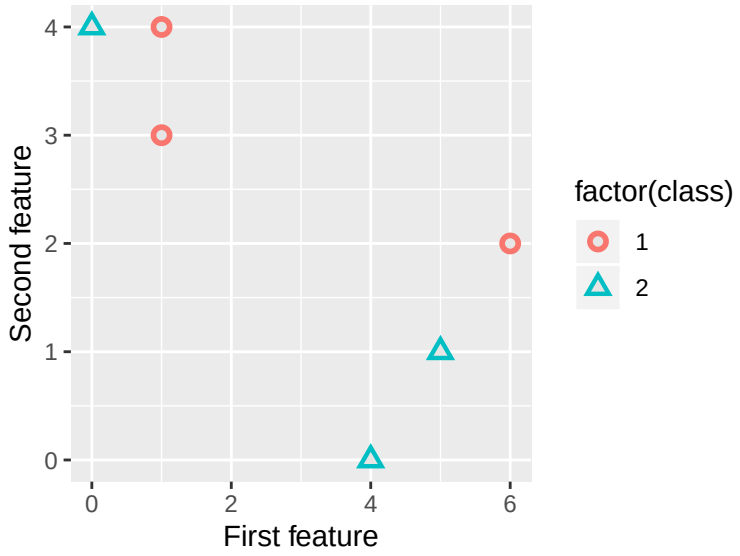
1. Plot the observations

```
scatter_plot <- ggplot(x, aes(x[,1], x[,2])) +
  geom_point(colour="black", size=3.5) +
  labs(x = "First feature", y = "Second feature")
scatter_plot
```



2. Randomly assign a cluster label to each observation

```
> set.seed(1)
> x$class <- floor(runif(nrow(x), min=1, max=3))
> pre_class_plot <- ggplot(x, aes(x[,1], x[,2], shape = factor(class))) +
+   geom_point(aes(colour = factor(class)), size = 3.5) +
+   geom_point(colour = "grey90", size = 1.5) +
+   labs(x = "First feature", y = "Second feature")
> data.frame(observation = c(1:6), class = x$class); pre_class_plot
  observation class
1           1     1
2           2     1
3           3     2
4           4     2
5           5     1
6           6     2
```



The class assignments are shown in the above table and graph.

3. Compute the centroid for each cluster

```
> #For the first class
> c1 <- c(mean(x[which(x$class==1),][,1]), mean(x[which(x$class==1),][,2]))
> c1
[1] 2.666667 3.000000
>
> #For the second class
> c2 <- c(mean(x[which(x$class==2),][,1]), mean(x[which(x$class==2),][,2]))
> c2
[1] 3.000000 1.666667
```

(2.666667, 3) is the first centroid, and (3, 1.666667) is the second centroid.

4. Assign each observation to the centroid to which it is closest

```
> x$new_class <- 0
> for (row in 1:nrow(x)){
+   pos <- c(x[,1][row], x[,2][row])
+   c1_dis <- (sum((pos - c1)^2))^(1/2)
+   c2_dis <- (sum((pos - c2)^2))^(1/2)
+   if (c1_dis > c2_dis){
+     x$new_class[row] <- 2
+   }
+   if (c1_dis < c2_dis){
+     x$new_class[row] <- 1
+   }
+   if (c1_dis == c2_dis){
+      x$new_class[row] <- sample(2,1)
+   }
+ }
> data.frame(observation = c(1:6), newclass = x$new_class)
  observation newclass
1           1        1
2           2        1
3           3        1
4           4        2
5           5        2
6           6        2
```

As we can see from the above, the first three observations are assigned to the first class, while the last three observations are assigned to the second class.

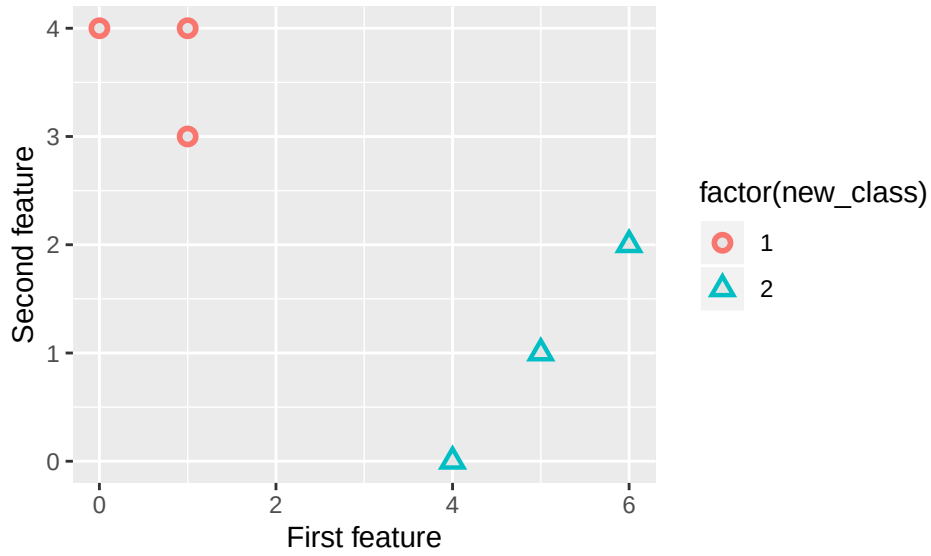
5. Repeat (3) and (4) until no changes

```
> i <- 1 #a counter of step
> while(identical(x$class, x$new_class)==FALSE){
+   if (i %% 2 != 0){
+     c1 <- c(mean(x[which(x$new_class==1),][,1]), mean(x[which(x$new_class==1),][,2]))
+     c2 <- c(mean(x[which(x$new_class==2),][,1]), mean(x[which(x$new_class==2),][,2]))
+     for (row in 1:nrow(x)){
+       pos <- c(x[,1][row], x[,2][row])
+       c1_dis <- (sum((pos - c1)^2))^(1/2)
+       c2_dis <- (sum((pos - c2)^2))^(1/2)
+       if (c1_dis > c2_dis){
+         x$class[row] <- 2
+         }
+       if (c1_dis < c2_dis){
+         x$class[row] <- 1
+         }
+       if (c1_dis == c2_dis){
+         x$class[row] <- sample(2,1)
+       }
+     }
+     i <- i + 1
+   }
+   if (i %% 2 == 1){
+     c1 <- c(mean(x[which(x$class==1),][,1]), mean(x[which(x$class==1),][,2]))
+     c2 <- c(mean(x[which(x$class==2),][,1]), mean(x[which(x$class==2),][,2]))
```

```
+     for (row in 1:nrow(x)){
+       pos <- c(x[,1][row], x[,2][row])
+       c1_dis <- (sum((pos - c1)^2))^(1/2)
+       c2_dis <- (sum((pos - c2)^2))^(1/2)
+       if (c1_dis > c2_dis){
+         x$new_class[row] <- 2
+         }
+       if (c1_dis < c2_dis){
+         x$new_class[row] <- 1
+         }
+       else{
+         x$new_class[row] <- sample(2,1)
+       }
+     }
+     i <- i + 1
+   }
+ }
> sprintf("Clustering converges after %d iteration(s)", i-1)
[1] "Clustering converges after 1 iteration(s)"
```

6. Reproduce the original plot by coloring

```
> #final clustering plot
> final_class_plot <- ggplot(x, aes(x[,1], x[,2], shape = factor(new_class))) +
+   geom_point(aes(colour = factor(new_class)), size = 3.5) +
+   geom_point(colour = "grey90", size = 1.5) +
+   labs(x = "First feature", y = "Second feature")
> final_class_plot
```



Q2 Clustering State Legislative Professionalism

1. Load the state legislative professionalism

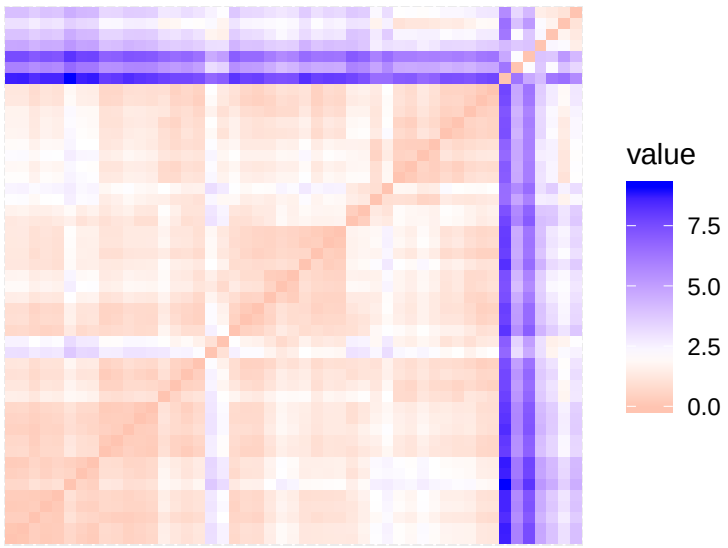
```
> leg <- get(load("/Users/Mark/Documents/GitHub/problem-set-4/Data and Codebook/legprof-components.v1.0
```

2. Munge the data

```
> #a. select only the continuous features (session length, salary, and expenditure)
> leg <- select(leg, c("stateabv", "sessid", "t_slength","slength","salary_real", "expend"))
>
> #b. restrict data to only include the 2009/10 legislative session
> leg <- filter(leg, sessid == "2009/10")
>
> #c. omit all missing values
> leg <- na.omit(leg)
>
> #d. standardize the input features
> features <- scale(leg[, c("t_slength","slength","salary_real", "expend")])
>
> #e. store the state variable
> states <- leg[, c("stateabv")]
>
> #additional. change the rownames to states
> row.names(features) <- states
```

3. Diagnose clusterability

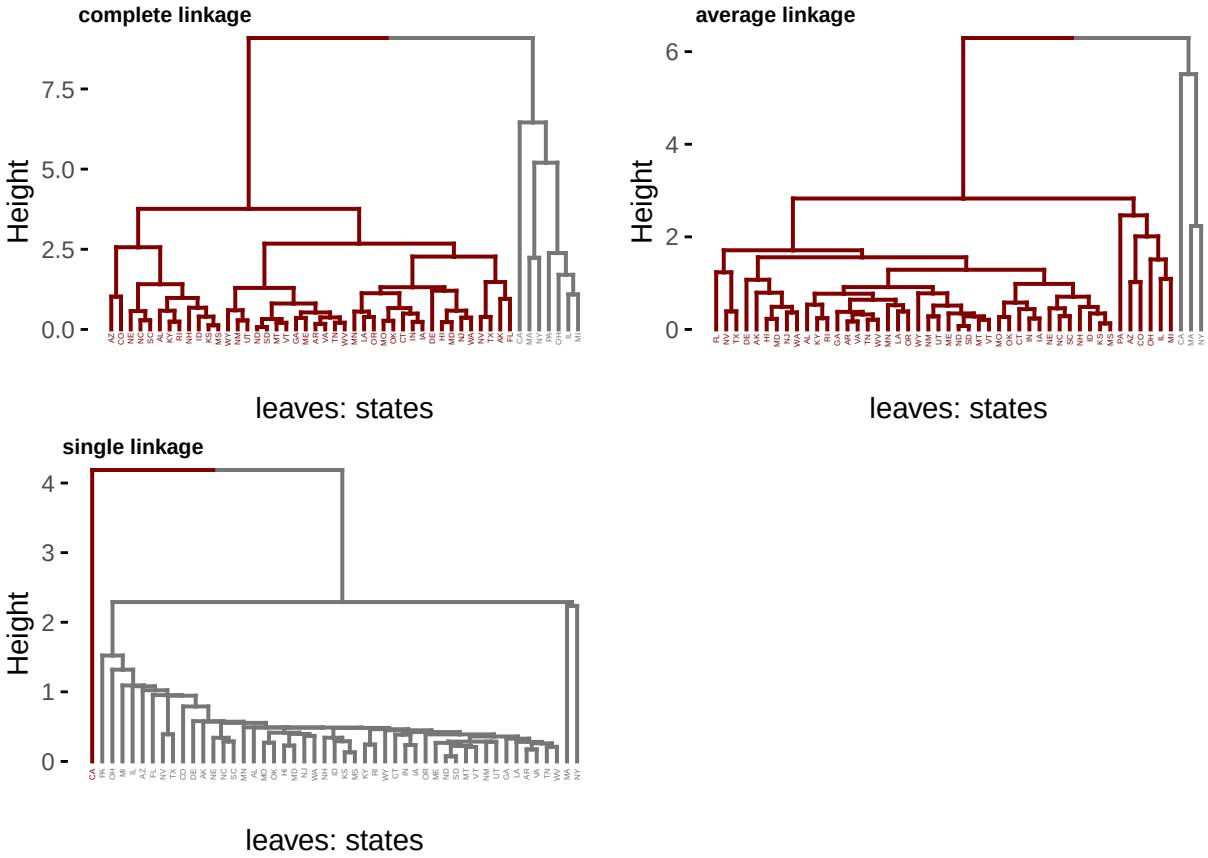
```
> cluster_diagnosis <- get_clust_tendency(features, nrow(features)-1, graph = TRUE,
+                                         gradient = list(low = "red", mid = "white",
+                                                         high = "blue"), seed = 1000)
> hopkins <- cluster_diagnosis$hopkins_stat
> plot <- cluster_diagnosis$plot
> hopkins;plot
[1] 0.8402627
```



As we know, a large value of hopkins statistics (greater than 0.5 and close to 1) indicate that the data is highly clustered. Since $0.8402627 > 0.5$, we can conlude that the data has a high cluster tendency. From the ODI graph, we can see that the right upper small red rectangle and the lower left big red rectangle suggest a low dissimilariry(high similararity) between objects "close" togther. This suggests that there might exist a non-random strcuture in the data. Therefore, we can perform clustering methods to this dataset.

4. Agglomerative hierchial clustering

```
> dsm <- dist(features)
>
> hc_complete <- hclust(dsm, method = "complete");
> complete_dendro <- fviz_dend(hc_complete, as.ggplot = TRUE, k = 2,
+                              k_colors = "uchicago", main = NULL,
+                              xlab = "leaves: states", cex = 0.2)
>
> hc_average <- hclust(dsm, method = "average");
> average_dendro <- fviz_dend(hc_average, as.ggplot = TRUE, k = 2,
+                             k_colors = "uchicago", main = NULL,
+                             xlab = "leaves: states", cex = 0.2)
>
> hc_single <- hclust(dsm, method = "single");
> single_dendro <- fviz_dend(hc_single, as.ggplot = TRUE, k = 2,
+                            k_colors = "uchicago", main = NULL,
+                            xlab = "leaves: states", cex = 0.2)
>
> hc_figure <- ggarrange(complete_dendro, average_dendro, single_dendro,
+                      labels = c("complete linkage", "average linkage", "single linkage"),
+                      font.label = list(size = 8), vjust = 1,
+                      ncol = 2, nrow = 2)
> hc_figure
```

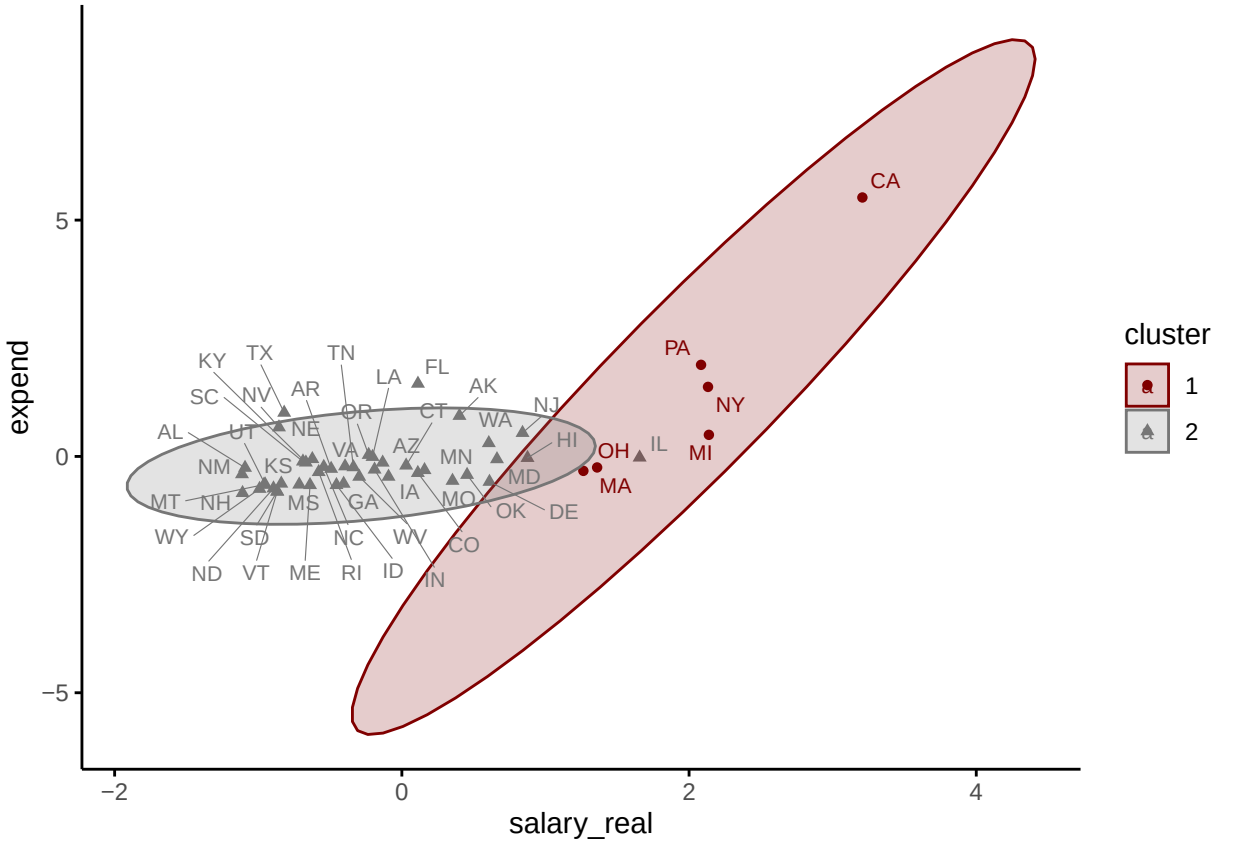


For ilustrative purposes, we cut the trees into two clusters (match the following two questions). Not surprisingly, the tree is the highest for the complete linkage, medium for the average linkage, and the lowest for the single linkage because of their definitions. As we can see from the above dendrograms, there is a general pattern that a large group of observations fuse at a relatively small hieght, implying close proximities (red for complete and average linkages, grey for single linkage) within the cluster. And the other group has fewer observations and they fuse at a relatively large height, implying that those observations are separated larger apart. In particular, for the average linkage, "CA", "MA", and "NY" form a cluster that is further away from the other cluster. It appears that "CA" is a point further away in all three methods. **I will use average linkage for the analysis in later questions.**

5. k-means

```
> set.seed(1000)
> km <- kmeans(features, 2)
> km
K-means clustering with 2 clusters of sizes 6, 43

Cluster means:
   t_slength    slength salary_real     expend
1  2.1000302  2.1014710   2.0307585  1.4677087
2 -0.2930275 -0.2932285  -0.2833616 -0.2047966

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 MS MO
 2  2  2  2  1  2  2  2  2  2  2  2  2  2  2  2  2  2  2  1  1  2  2  2
MT NE NV NH NJ NM NY NC ND OH OK OR PA RI SC SD TN TX UT VT VA WA WV WY
```

```
 2  2  2  2  2  2  1  2  2  1  2  2  1  2  2  2  2  2  2  2  2  2  2  2

Within cluster sum of squares by cluster:
[1] 40.36173 48.36594
 (between_SS / total_SS =  53.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
>
> km_figure <- fviz_cluster(list(data = features, cluster = km$cluster),
+                           ellipse.type = "norm",  palette = "uchicago", main = NULL,
+                           show.clust.cent = FALSE, choose.vars = c("salary_real","expend"),
+                           labelsize = 8, repel = TRUE, ggtheme = theme_classic())
> km_figure
```



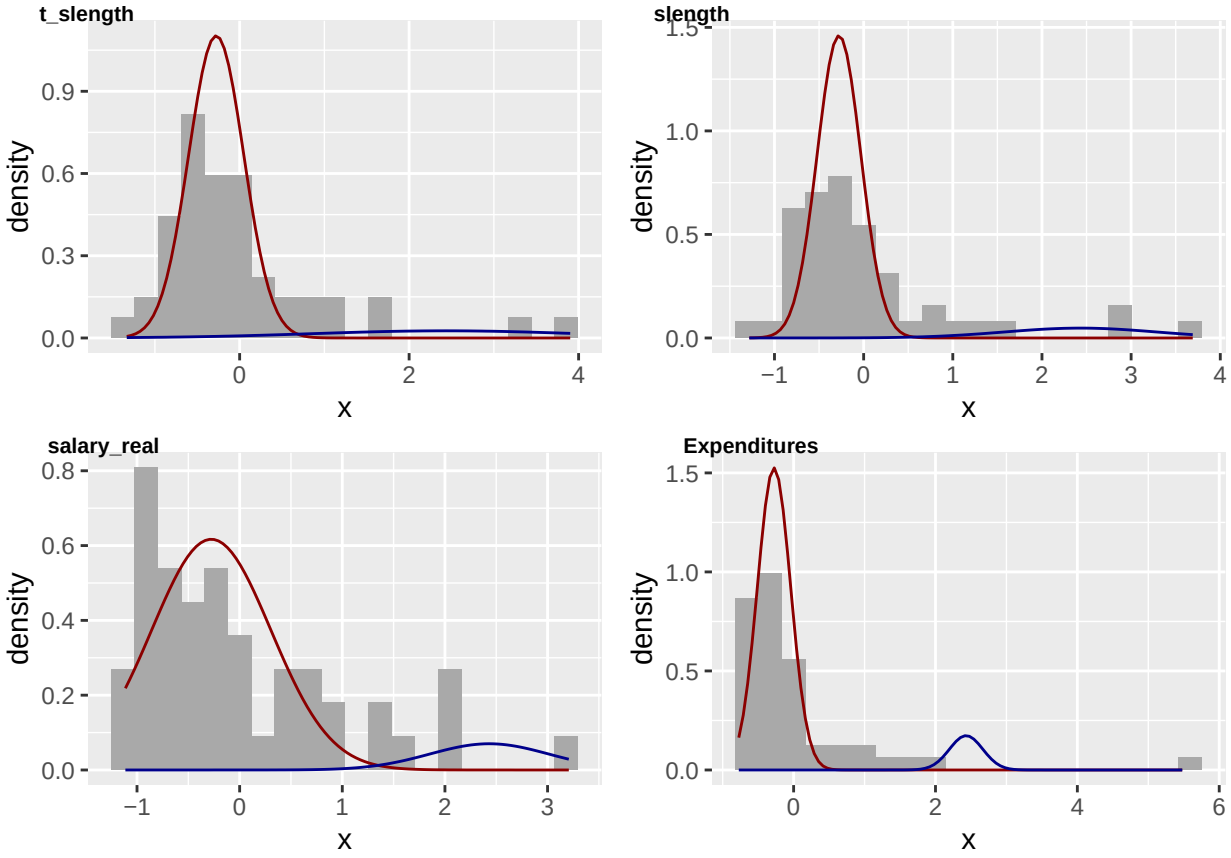
As we can see from the above (I plot salary_real vs expend here, other pairs of features would generate different looks), one of the cluster is pretty centered while the other contanins sparsely distributed points. This results suggests a similar pattern as shown in the previous question. Specifically, kmeans with $k = 2$ splits data into one cluster of size 6 and the other of size 43. The WSS of the first cluster is 40.36173, and the WSS of the second is 48.36594. The states in the sparse red region are "MA", "OH", "NY", "PA", "CA", and "MI".

6. Gaussian mixture model via the EM algorithm

```
> set.seed(123)
>
> gmm <- mvnormalmixEM(features, k = 2)
>
> gmm_figure_t_slength <- ggplot(data.frame(x = gmm$x[,1])) +
+   geom_histogram(aes(x, ..density..), fill = "darkgray", bins = 20) +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[1]][1], gmm$sigma[[1]][1,1], lam = gmm$lambda[1]),
+                 colour = "darkred") +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[2]][1], gmm$sigma[[2]][1,1], lam = gmm$lambda[2]),
+                 colour = "darkblue")
>
>   theme_classic()
>
> gmm_figure_slength <- ggplot(data.frame(x = gmm$x[,2])) +
+   geom_histogram(aes(x, ..density..), fill = "darkgray", bins = 20) +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[1]][1], gmm$sigma[[1]][2,2], lam = gmm$lambda[1]),
+                 colour = "darkred") +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[2]][1], gmm$sigma[[2]][2,2], lam = gmm$lambda[2]),
+                 colour = "darkblue")
>
>   theme_classic()
>
> gmm_figure_salary_real <- ggplot(data.frame(x = gmm$x[,3])) +
+   geom_histogram(aes(x, ..density..), fill = "darkgray", bins = 20) +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[1]][1], gmm$sigma[[1]][3,3], lam = gmm$lambda[1]),
+                 colour = "darkred") +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[2]][1], gmm$sigma[[1]][3,3], lam = gmm$lambda[2]),
+                 colour = "darkblue")
>
>   theme_classic()
>
> gmm_figure_expend <- ggplot(data.frame(x = gmm$x[,4])) +
+   geom_histogram(aes(x, ..density..), fill = "darkgray", bins = 20) +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[1]][1], gmm$sigma[[1]][4,4], lam = gmm$lambda[1]),
+                 colour = "darkred") +
+   stat_function(geom = "line", fun = plot_mix_comps,
+                 args = list(gmm$mu[[2]][1], gmm$sigma[[1]][4,4], lam = gmm$lambda[2]),
+                 colour = "darkblue")
>
>   theme_classic()
```

```
> gmm_figure_combine <- ggarrange(gmm_figure_slength, gmm_figure_t_slength,
+                                 gmm_figure_salary_real, gmm_figure_expend,
+                                 labels = c("t_slength", "slength", "salary_real",
+                                            "Expenditures"),
+                                 font.label = list(size = 8), vjust = 1,
```

```
+                                   ncol = 2, nrow = 2)
> gmm$lambda; gmm$mu; gmm$sigma; gmm_figure_combine
[1] 0.897959 0.102041
[[1]]
[1] -0.2763465 -0.2450724 -0.1943875 -0.1921007

[[2]]
[1] 2.431846 2.156634 1.710608 1.690484
[[1]]
           [,1]       [,2]      [,3]       [,4]
[1,] 0.24528126 0.27954563 0.2096713 0.03105197
[2,] 0.27954563 0.32491503 0.2375517 0.02479181
[3,] 0.20967129 0.23755167 0.5806866 0.13660923
[4,] 0.03105197 0.02479181 0.1366092 0.23481667

[[2]]
          [,1]       [,2]      [,3]       [,4]
[1,] 0.8556104  1.0197243 0.3979561  0.3509037
[2,] 1.0197243  1.5611386 0.3426861 -0.3956420
[3,] 0.3979561  0.3426861 1.2312537  1.9833537
[4,] 0.3509037 -0.3956420 1.9833537  4.3511252
```

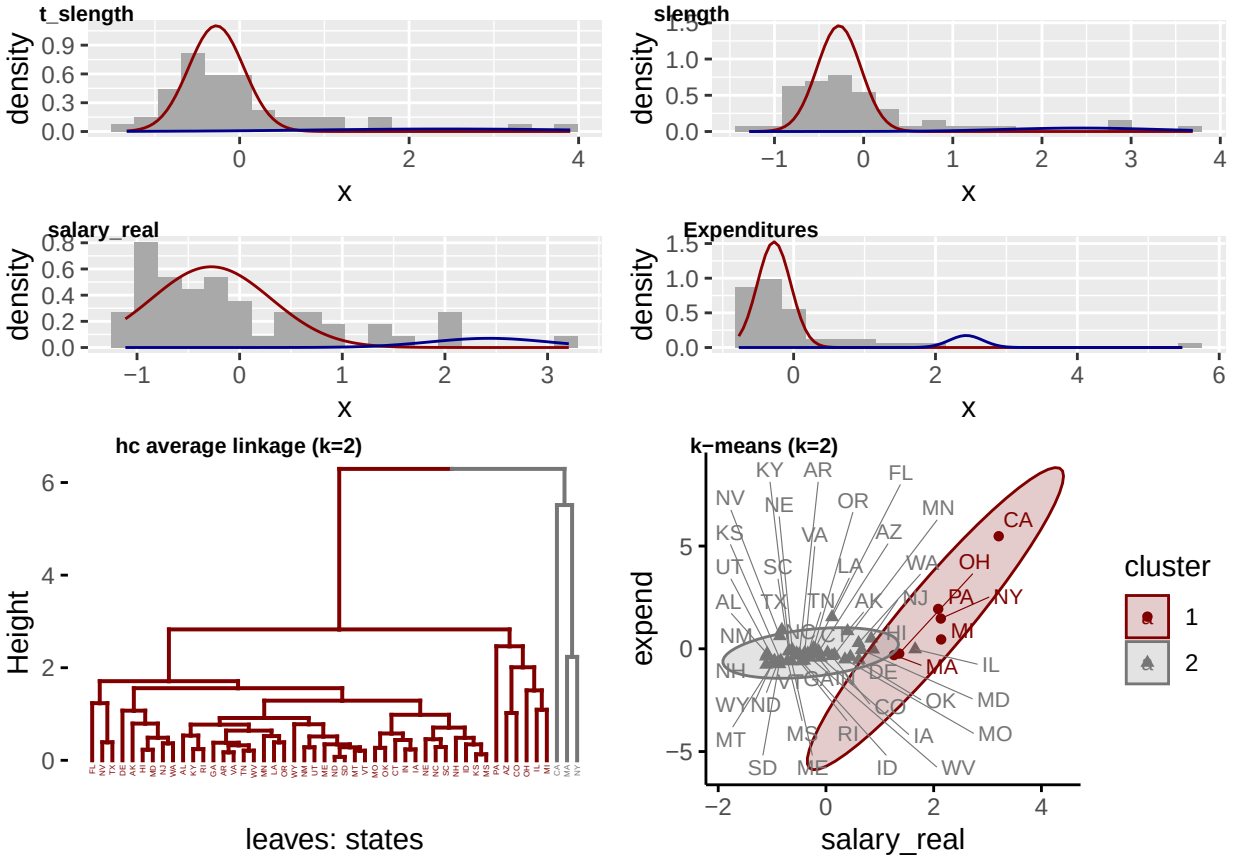


As we can see from the above results, 89.80% of the states fall into the first cluster and 10.20% of the states fall into the second cluster (each state has a "part" belonging to the first cluster and the other "part" belonging to the second clsuter. This is in some sense equivalent to the description above, but not necessarily each state is strictly classified into a certain cluster). The means of each of the features and

the covariance matrix are reported in the above charts. As we can see from the above graphs across four features (since this is a multivariate gaussian distribution, we can only visualize each variable separately), it's highly likely that the observations are drawn from different gaussian distributions.

7. Compare results visually

```
> Compare_fig <- ggarrange(average_dendro, km_figure,
+                     labels = c("hc average linkage (k=2)", "k-means (k=2)"),
+                     font.label = list(size = 8), vjust = 1,
+                     ncol = 2, nrow = 1)
> ggarrange(gmm_figure_combine, Compare_fig, ncol=1, nrow=2)
```



The above graphs suggest similar patterns using three different clustering methods. One cluster seems more centered with more observations while the other is more scattered with fewer observations.

8. Validation of the three methods

```
> summary(hc_valid)

Clustering Methods:
 hierarchical

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

Optimal Scores:

             Score  Method       Clusters
Connectivity 6.0869 hierarchical 2
Dunn         0.4371 hierarchical 3
Silhouette   0.6994 hierarchical 2
> summary(km_valid)

Clustering Methods:
 kmeans

Cluster sizes:
 2 3 4 5 6 7 8 9 10

Validation Measures:
                           2       3       4       5       6       7       8       9      10

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

Optimal Scores:

             Score  Method Clusters
Connectivity 8.4460 kmeans 2
Dunn         0.2581 kmeans 3
Silhouette   0.6458 kmeans 2
> summary(gmm_valid)

Clustering Methods:
 model

Cluster sizes:
 2 3 4 5 6 7 8 9 10

Validation Measures:
                           2       3       4       5       6       7       8       9      10

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 10.7393 model  2
```

```
Dunn          0.1810 model  9
Silhouette    0.6314 model  2
```

The above charts show the validation results of the three methods. Connectiviy scores, Dunn scores, and Silhouette scores are reported. By all scores, it seems hierarchical clustering is the best method (smallest Connectivity, largest Dunn and Silhouette).

9. Discuss the validation output

Let's focus on average silhouette width for the discussion. The Silhouette value measures the degree of confidence in a particular clustering assignment and lies in the interval [-1,1], with well-clustered observations having values near 1 and poorly clustered observations having values near -1.

- As we can see, the Silhouette scores suggest picking $k = 2$ is optimal for all of the three methods (Connectivity also suggests the same results). This confirms our choice of k in the previous prblems. The three methods yield similar partterns where most observations gather into a denser cluster while the rest of the observations are sparsely distributed in the other cluster.
- Since hiarchial clustering yields the best Silhouette score, we conclude that hierchial clustering with $k = 2$ is the optimal approach in this case.
- Both K-means and hierarchical clustering will assign each observation to a cluster. However, for instance, suppose that most of the observations truly belong to a small number of (unknown) subgroups, and a small subset of the observations are quite different from each other and from all other observations.(This is kind of like the case in this problem.) Then since K-means and hierarchical clustering force every observation into a cluster, the clusters found may be heavily distorted due to the presence of outliers that do not belong to any cluster. Therefore, in this case, even though the validation statistics might suggest the previous methods, we should probably still choose GMM.