Intro to Machine Learning - Problem Set 4: Clustering

Ipek Cinar

3/02/2020

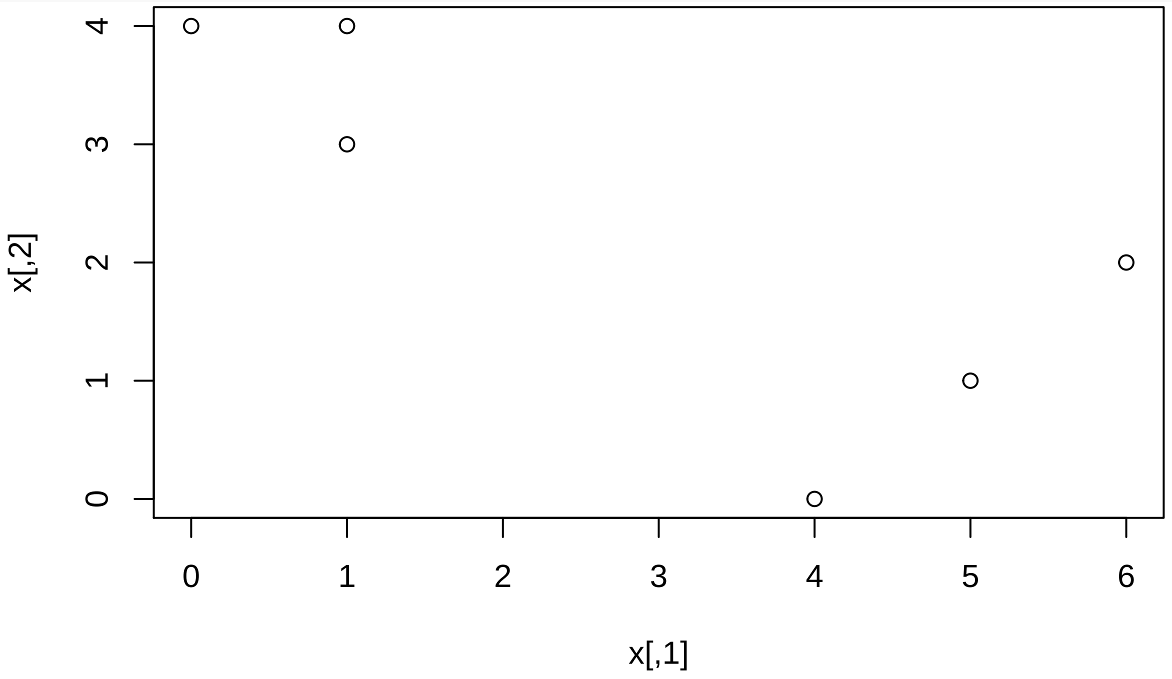
```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(tidyverse)
library(dplyr)
library(gridExtra)
```

Performing k-Means By Hand

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

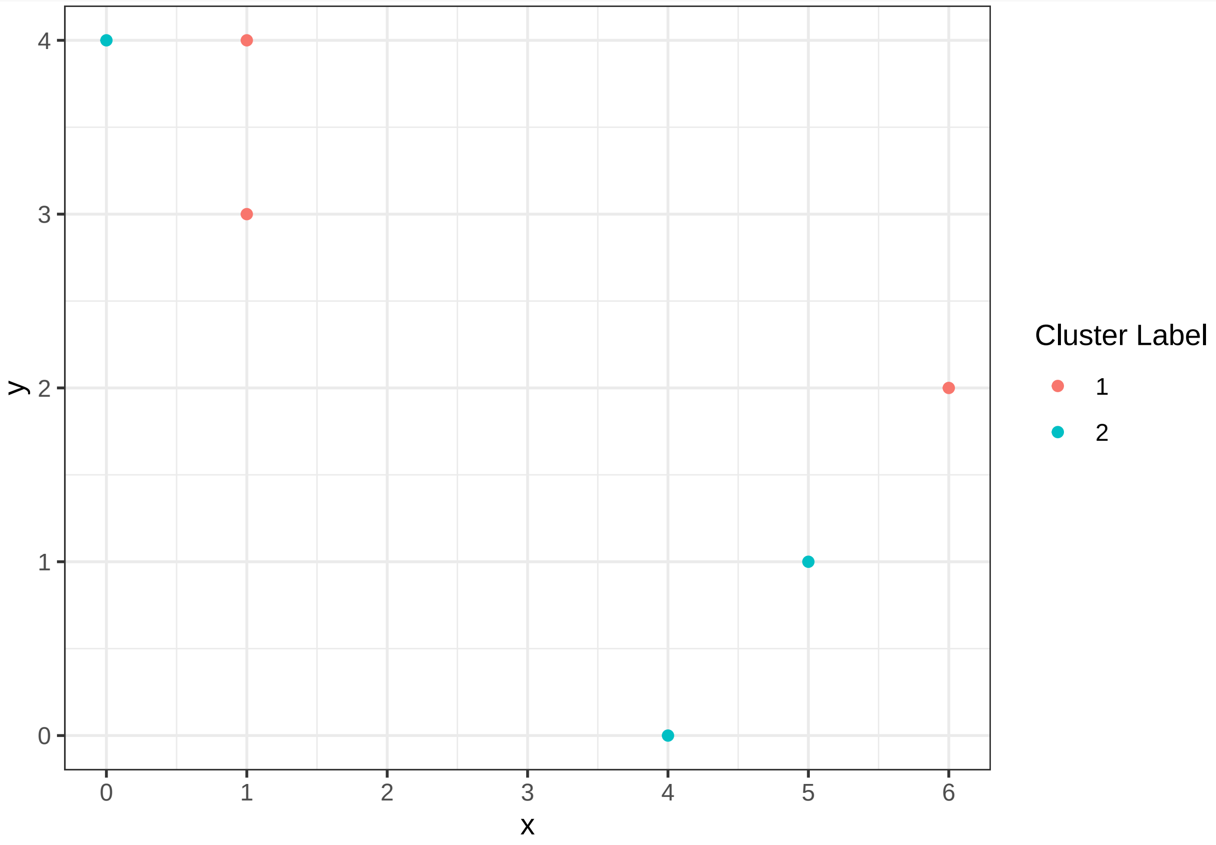
1. Plot the observations.

```
plot(x)
```



2. 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(1)
length = nrow(x)
colnames(x) <- c("x", "y")
x = as.data.frame(x)
data <- x %>% mutate(cluster_label = as.factor(sample(1:2, length, replace = TRUE)))
ggplot(data, aes(x, y, color = cluster_label)) + geom_point() +
  scale_x_continuous(breaks=seq(0,6,1)) +
  scale_y_continuous(breaks=seq(0,4,1)) +
  theme_bw() + labs(color= "Cluster Label")
```



3. Compute the centroid for each cluster.

```
cluster_centroids <- data %>% group_by(cluster_label) %>%
                     summarize_all(~mean(.))

cluster_centroids %>% knitr::kable()
```

cluster_label	x	y
1	2.666667	3.000000
2	3.000000	1.666667

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

```
change_cluster <- function(data){
  data_centroid <- data %>% group_by(cluster_label) %>% summarize_all(~mean(.))
  new_cluster <- data %>% mutate(cluster_label = ifelse(
    (((data$x - data_centroid$x[1])^2) + ((data$y - data_centroid$y[1])^2) <
     ((data$x - data_centroid$x[2])^2) + ((data$y - data_centroid$y[2])^2)), 1, 2))
  new_cluster %>% dplyr::select(x, y, cluster_label) %>% return()
}
```

```r
# Reporting
knitr::kable(change_cluster(data))
```

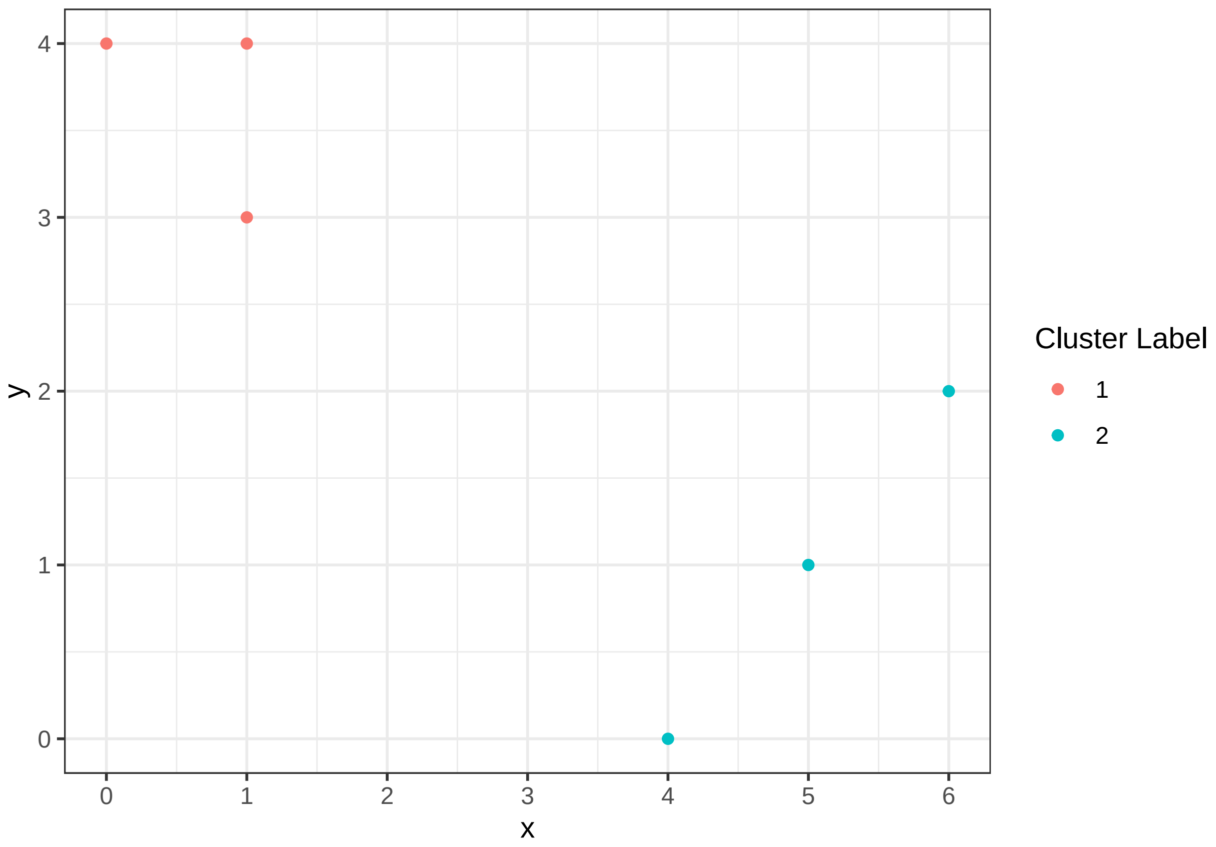
x	y	cluster_label
1	4	1
1	3	1
0	4	1
5	1	2
6	2	2
4	0	2

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

```r
end_same <- FALSE
while(!end_same){
  old_cluster_label <- data$cluster_label
  data <- change_cluster(data)
  end_same <- all(old_cluster_label == data$cluster_label)
}
```

6. 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.

```r
# Plotting
ggplot(data, aes(x, y, color = as.factor(cluster_label))) +
  geom_point() +
  scale_x_continuous(breaks=seq(0,6,1)) +
  scale_y_continuous(breaks=seq(0,4,1)) +
  theme_bw() + labs(color= "Cluster Label")
```



Clustering State Legislative Professionalism

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

```
load("~/Desktop/Intro to Machine Learning/Problem Sets/PSet 4/legprof-components.v1.0.RData")
slp_data <- x
rm(x)
```

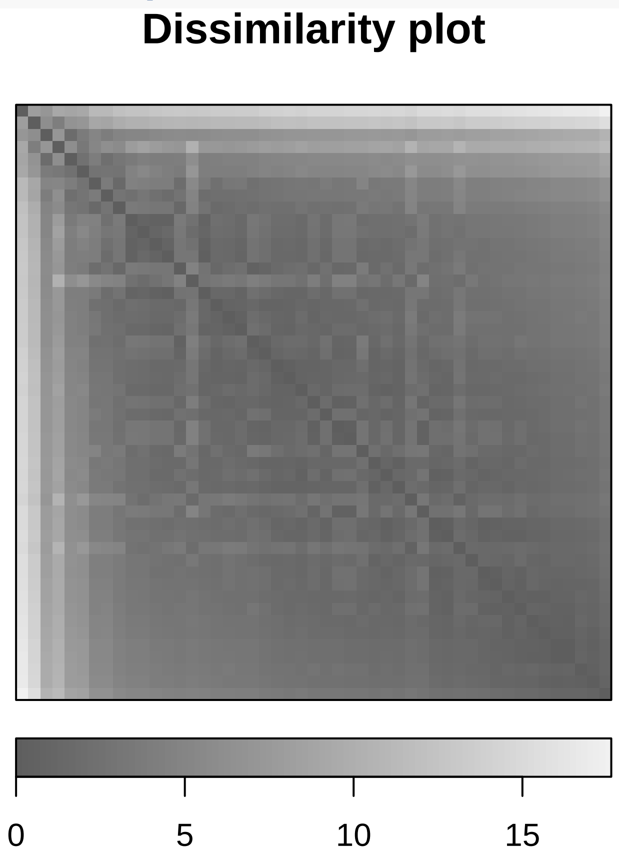
2. Munge the data: a. select only the continuous features that should capture a state legislature's level of "professionalism" (session length (total and regular), salary, and expenditures); b. restrict the data to only include the 2009/10 legislative session for consistency; c. omit all missing values; 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).

```
slp_data_munged <- slp_data %>%
                   filter(sessid == "2009/10") %>%
                   mutate(rowname = stateabv) %>%
                   column_to_rownames() %>%
                   dplyr::select(t_slength, slength, salary_real, expend) %>%
                   drop_na() %>%
                   scale()


slp_data_states <- slp_data %>%
                    filter(sessid == "2009/10") %>%
                    dplyr::select(stateabv, t_slength, slength, salary_real, expend) %>%
                    na.omit() %>% dplyr::select(stateabv)
```

3. 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.

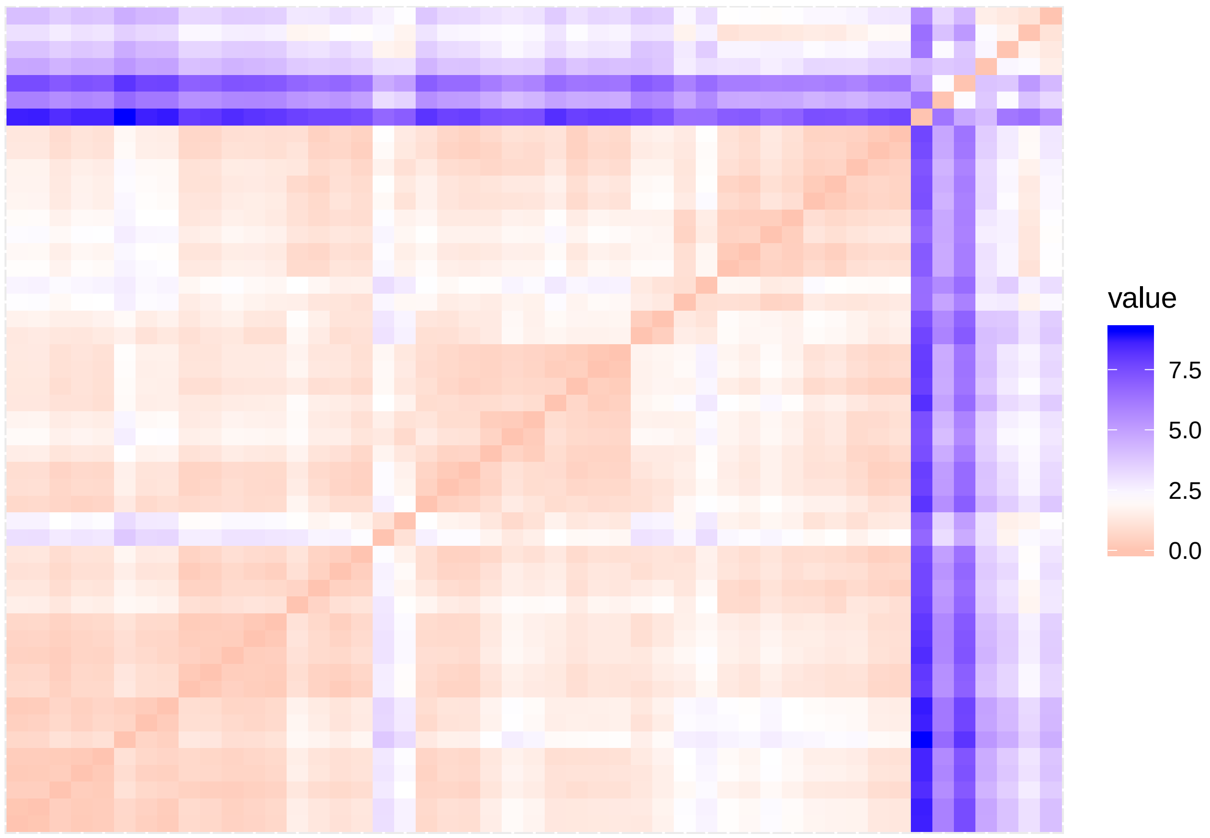
```r
# Dissimilarity Plot using seriation package:
distance_cluster <- dist(slp_data_munged, method = "manhattan")
seriation::dissplot(distance_cluster,
         options = list(main = "Dissimilarity plot"))
```



Using the dissimilarity matrix shading, we can see that there appears to be non-random structure existing in the data (looking along the diagonal), suggesting that the data is stratified and hence clusterable.

```r
# Assessing Clustering Tendency using factoextra package:
factoextra::get_clust_tendency(slp_data_munged, 40)
```

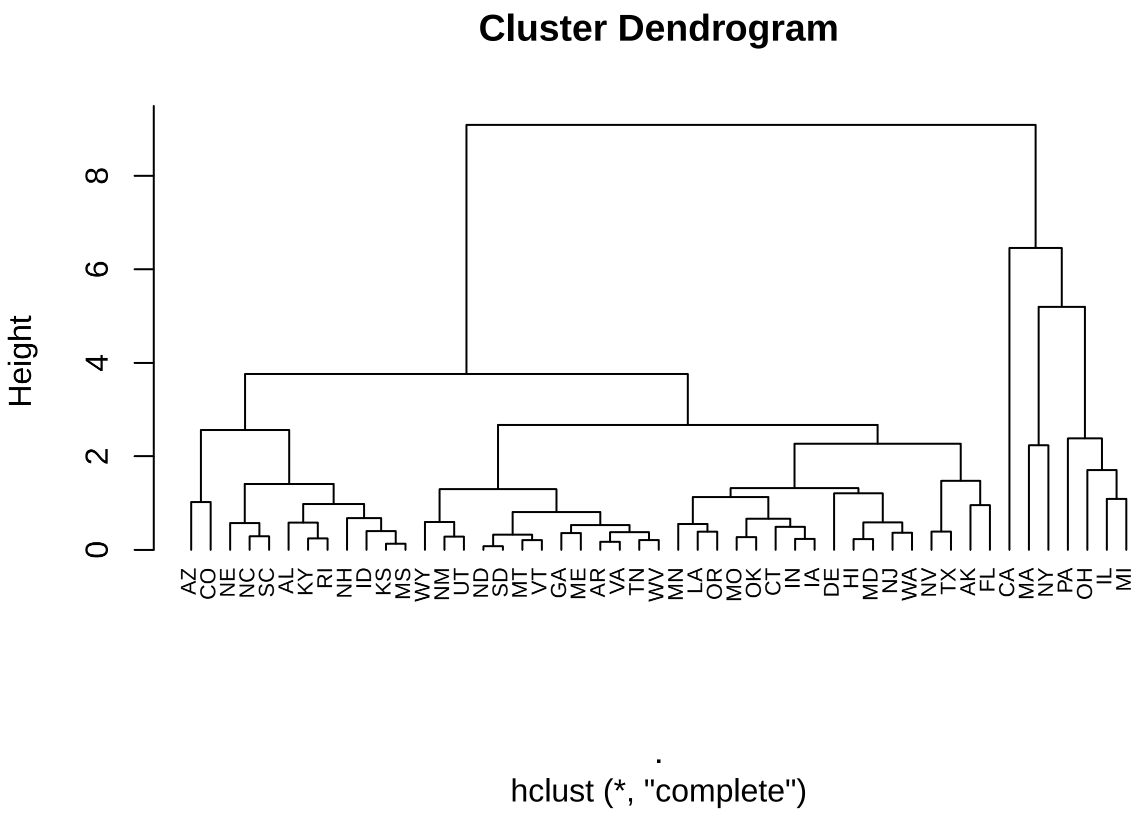
```
## $hopkins_stat
## [1] 0.8406165
##
## $plot
```



We can also assess clusterability using the Hopkins statistic. According to the Hopkins statistic we get - 0.841, the data is significantly clusterable (as suggested by the function's documentations: If the value of Hopkins statistic is close to 1 (far above 0.5), then we can conclude that the dataset is significantly clusterable.) We can also detect it by looking at the square shaped dark (or colored) blocks along the diagonal in the VAT image.

4. 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.

```r
hierarchical_clustering <- dist(slp_data_munged) %>% hclust(method = "complete")
plot(hierarchical_clustering, hang = -2, cex = 0.7)
```



The height (y-axis) of the Cluster Denongram shows the measure of closeness data points/clusters. Looking at the Cluster Denongram plot above, we can say that the more wealthy and/or populated states like California (CA), Massachusetts (MA), New York (NY), Pennsylvania (PA) seem to be more dissimilar from the rest of the states; while states pairs like North Dakota (ND) - South Dakota (SD) and Kansas (KS) - Mississipi (MS) are the most similar to one another. Hence, the similarity pairs signal to us that the geographically close /neigboring states are more similar to each other than the ones that are farther. However, we have to say that with caution since Hawaii (HI) seems to appear as most similar to, Maryland (MD) which does not fit into the geographical similarity hypothesis.

5. 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(123)
k_means <- kmeans(slp_data_munged, centers = 2)
#cluster_data_kmeans <- table(table(k_means$cluster))
cluster_data_kmeans <- as.data.frame(as.table(k_means$cluster))
colnames(cluster_data_kmeans) <- c("State Abbrev", "ClusterAssignment")
cluster_data_kmeans %>% count(ClusterAssignment) %>% knitr::kable()
```

ClusterAssignment	n
1	43
2	6

As seen from the table above, our k-means algorithm using two clusters, classified 6 observations/states in Cluster 2 while the rest of 43 observations/states was classified in Cluster 1. Even though it is worth to see how the clustering changes when we increase k = 2 to see how the classifications into clusters differ, the k-means algorithm returned a failt similar result that is in line with our Cluster Denongram in q4 where we used agglomerative hierarchical clustering algorithm.

6. 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.

```r
set.seed(123)
gaussian_mixture_model <- mixtools::mvnormalmixEM(slp_data_munged, k = 2)
```

```
## number of iterations= 25
```

```r
gaussian_mixture_model_results <- as.data.frame(gaussian_mixture_model$posterior) %>%
  cbind(slp_data_states) %>% mutate(ClusterAssignment = ifelse(comp.1 < comp.2, 2, 1)) %>%
  dplyr::select(stateabv, ClusterAssignment) %>% count(ClusterAssignment)

gaussian_mixture_model_results %>% knitr::kable()
```

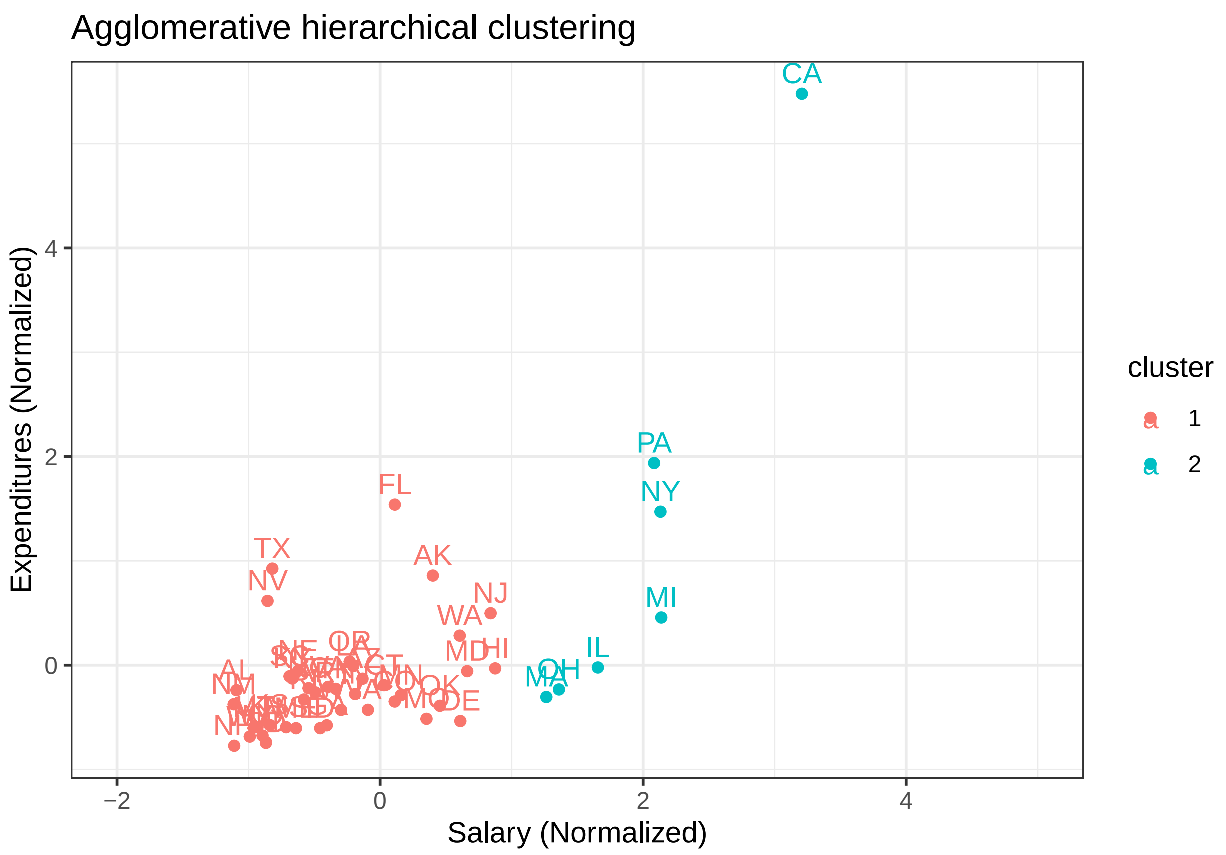
ClusterAssignment	n
1	44
2	5

Similar to our k-means algorithm, gaussian mixture model classified 5 observations/states in Cluster 2 while the rest of 44 observations/states was classified in Cluster 1. This also signals to me that the models are picking up the non-randomness present in the data.

7. 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.

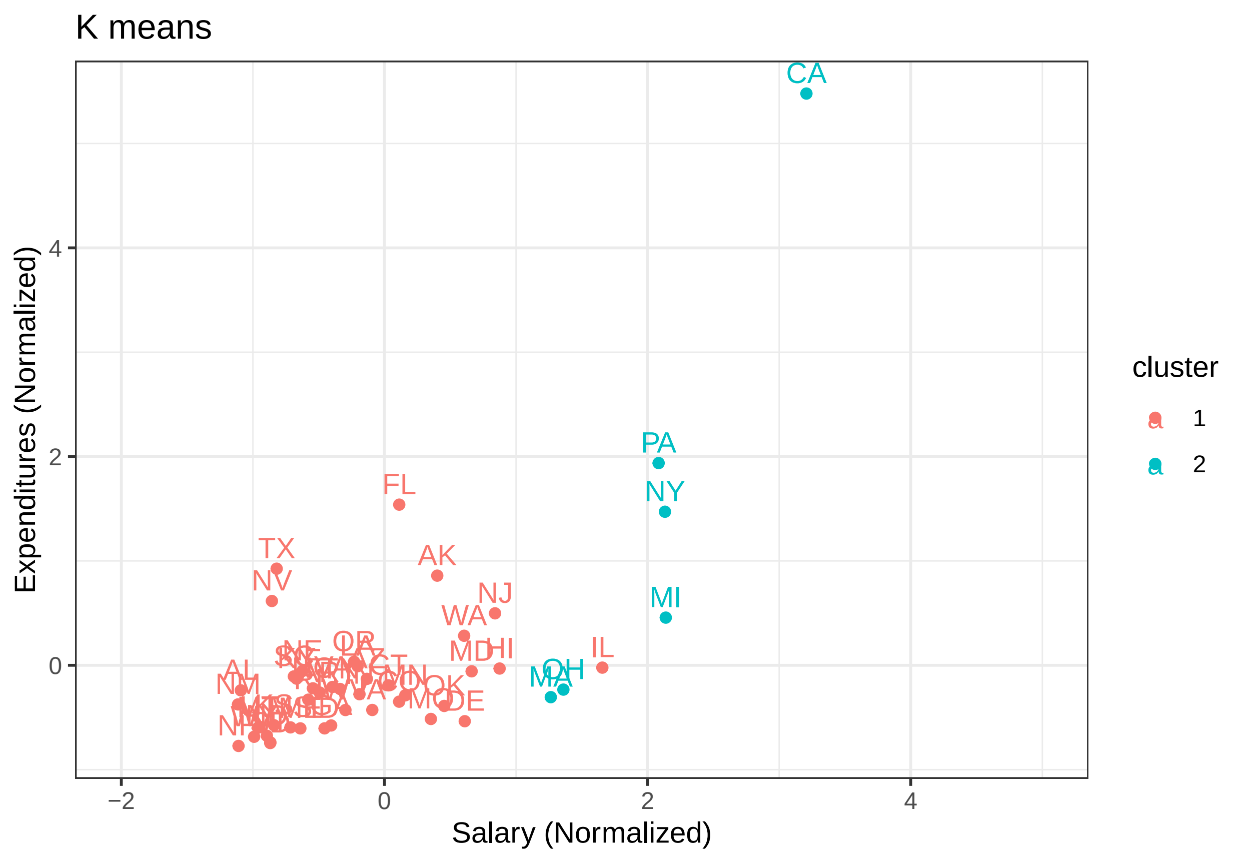
A) Salary and Expenditures:
Please see the plot below for a visualization of the clustering by states across salary and expenditures using agglomerative hierarchical clustering algorithm.

```r
#Visualizing clusters with salary and expenditures:
p1 <- slp_data_munged %>% as_tibble() %>% mutate(cluster = as.factor(
  cutree(hierarchical_clustering, k=2))) %>%
  ggplot(aes(salary_real, expend, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (y = "Expenditures (Normalized)", x= "Salary (Normalized)",
        title = "Agglomerative hierarchical clustering") +
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p1
```



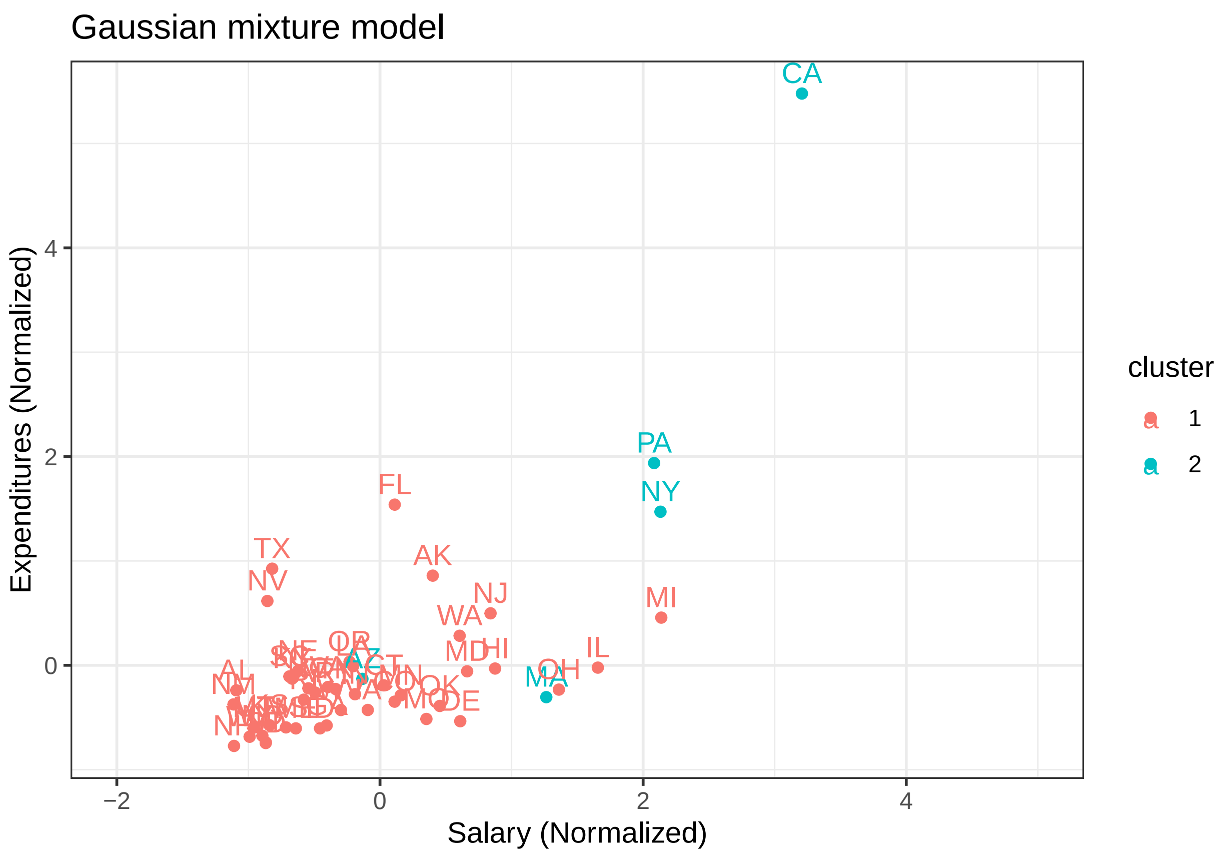
Please see the plot below for a visualization of the clustering by states across salary and expenditures using k means algorithm.

```r
#Visualizing clusters with salary and expenditures:
p2 <- slp_data_munged %>% as_tibble() %>% mutate(cluster = as.factor(
  k_means$cluster)) %>%
  ggplot(aes(salary_real, expend, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (y = "Expenditures (Normalized)", x= "Salary (Normalized)",
        title = "K means") +
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p2
```



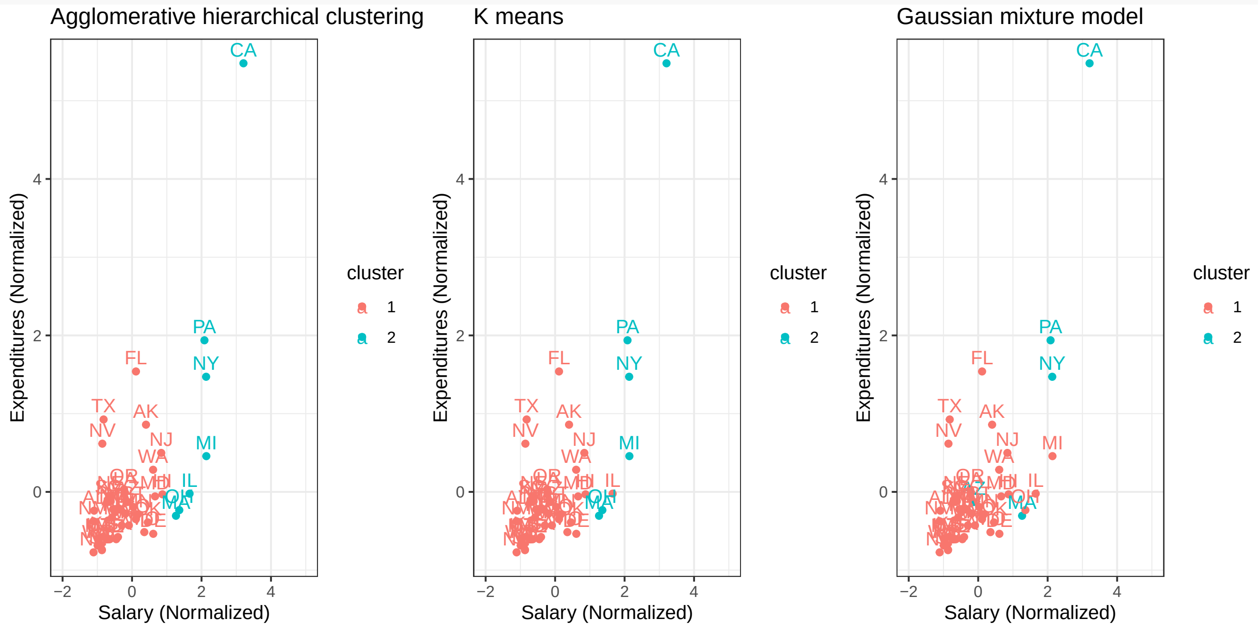
Please see the plot below for a visualization of the clustering by states across salary and expenditures using gaussian mixture model.

```r
#Visualizing clusters with salary and expenditures:
p3 <- slp_data_munged %>% as_tibble() %>% mutate(cluster = as.factor(
  apply(gaussian_mixture_model$posterior, 1, which.max))) %>%
  ggplot(aes(salary_real, expend, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (y = "Expenditures (Normalized)", x= "Salary (Normalized)",
        title = "Gaussian mixture model") +
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p3
```



Placing agglomerative hierarchical clustering , k-means and gaussian mixture models together:

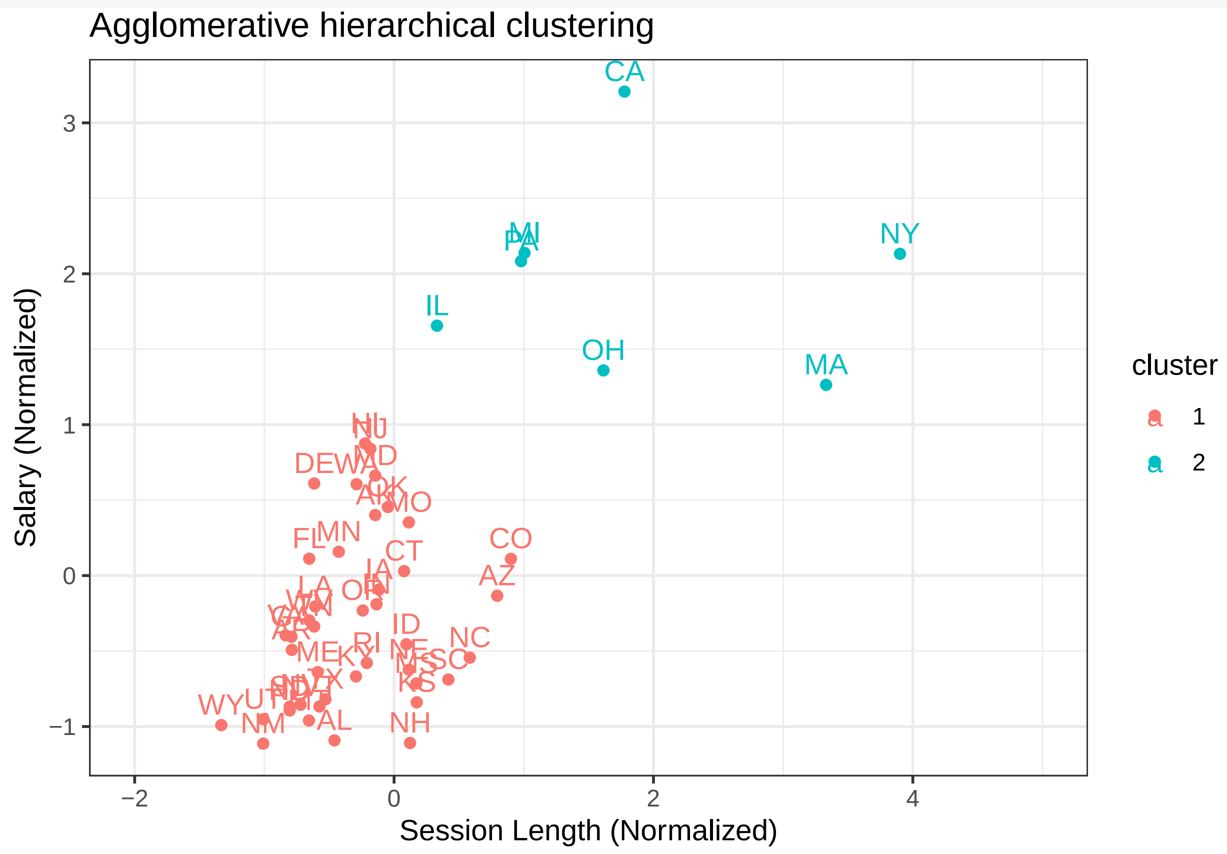
```r
grid.arrange(p1, p2, p3, nrow=1)
```



B) Session Length and Salary:

Please see the plot below for a visualization of the clustering by states across session length and salary using agglomerative hierarchical clustering algorithm.

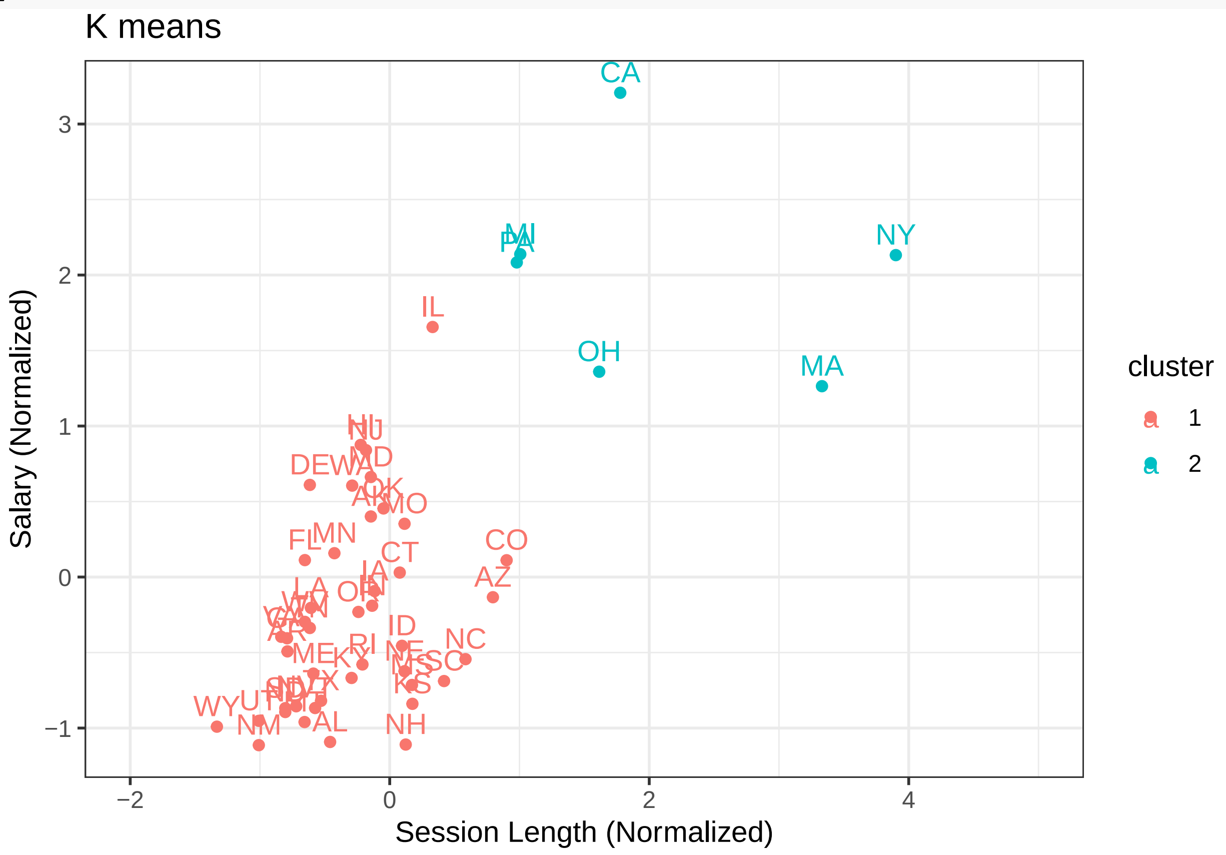
```r
#Visualizing clusters with session length and salary:
p4 <- slp_data_munged %>% as_tibble() %>% mutate(cluster = as.factor(
  cutree(hierarchical_clustering, k=2))) %>%
  ggplot(aes(slength, salary_real, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (x = "Session Length (Normalized)", y= "Salary (Normalized)",
        title = "Agglomerative hierarchical clustering") +
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p4
```



Please see the plot below for a visualization of the clustering by states across session length and salary using k means algorithm.

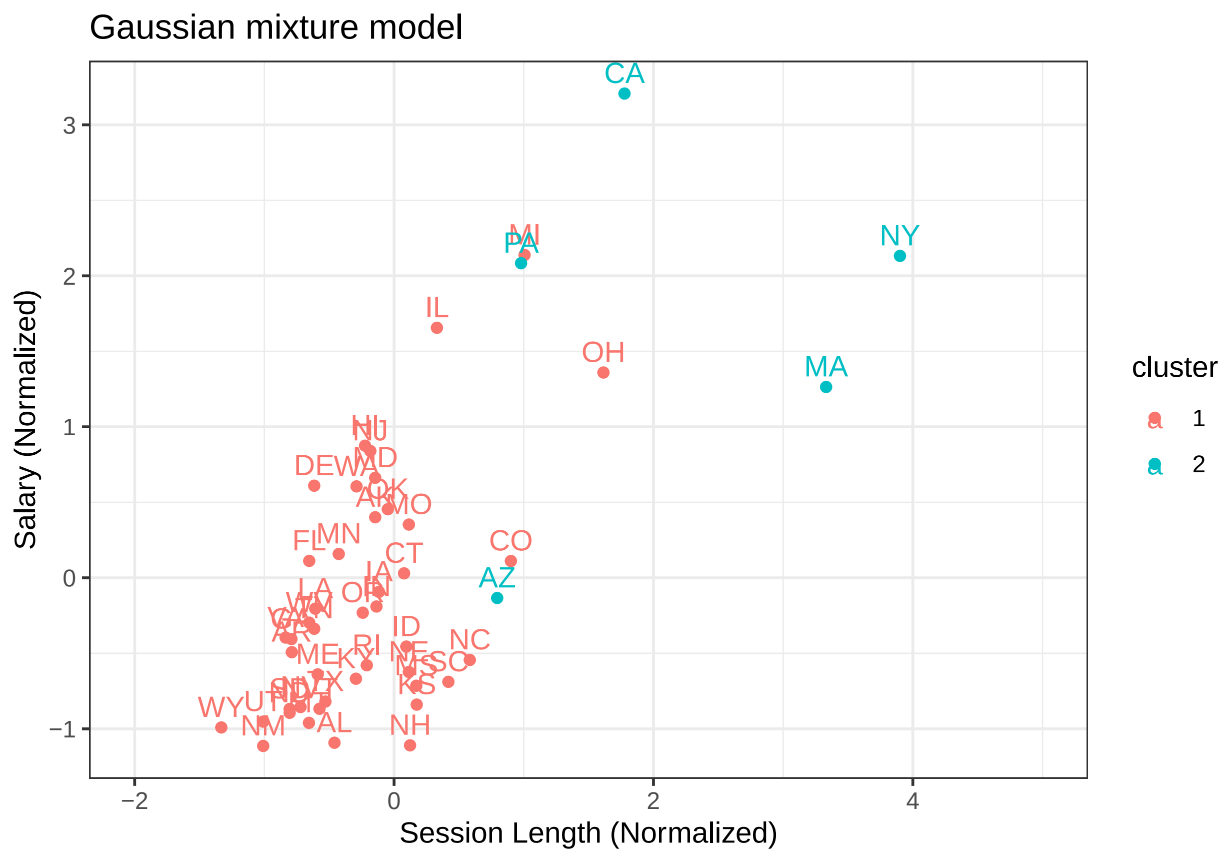
```r
#Visualizing clusters with session length and salary:
col_1 <- slp_data_states
col_2 <- as.data.frame(k_means$cluster)
df <- cbind(col_1, col_2)

#Visualizing clusters with session length and salary:
p5 <- slp_data_munged %>% as_tibble() %>% mutate(cluster =
                                                  as.factor(k_means$cluster)) %>%
  ggplot(aes(slength, salary_real, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (x = "Session Length (Normalized)", y= "Salary (Normalized)",
        title = "K means")+
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p5
```



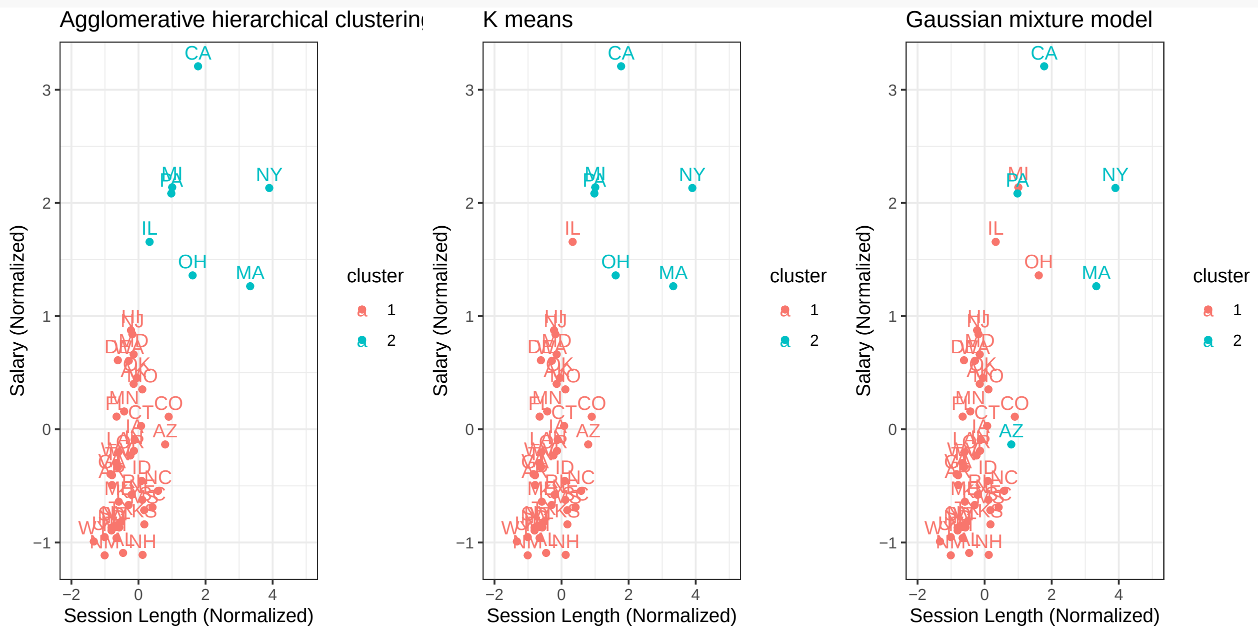
Please see the plot below for a visualization of the clustering by states across session length and salary using gaussian mixture model.

```r
#Visualizing clusters with session length and salary:
p6 <- slp_data_munged %>% as_tibble() %>% mutate(cluster = as.factor(
  apply(gaussian_mixture_model$posterior, 1, which.max))) %>%
  ggplot(aes(slength, salary_real, colour = cluster)) +
  geom_point() +
  geom_text(aes(label = rownames(slp_data_munged)), vjust = -0.5 ) +
  labs (x = "Session Length (Normalized)", y= "Salary (Normalized)",
        title = "Gaussian mixture model") +
  theme_bw() + coord_cartesian(xlim =c(-2,5))
p6
```



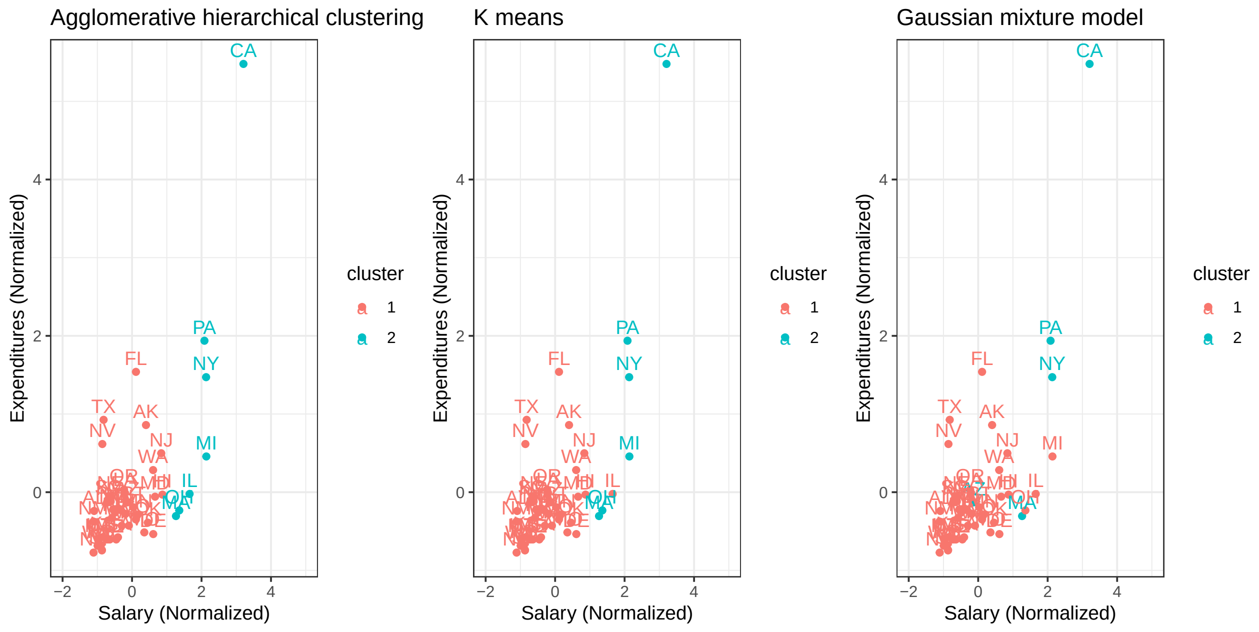
Placing agglomerative hierarchical clustering , k-means and gaussian mixture models together:

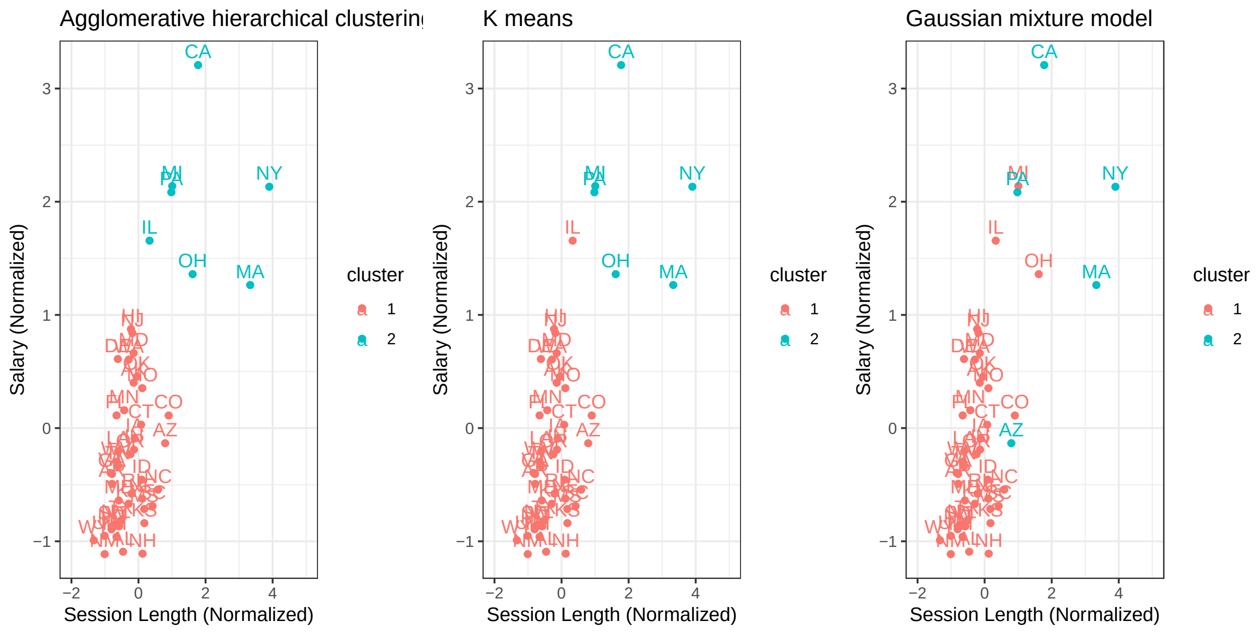
```r
grid.arrange(p4, p5, p6, nrow = 1, ncol=3)
```



Placing agglomerative hierarchical clustering , k-means and gaussian mixture models together for expenditures-salary and salary-sesison length:

```r
grid.arrange(p1, p2, p3, p4, p5, p6, nrow=2)
```





As we can see, there aren't drastic differences between the three different clustering methods - hierarchical, k-means and gaussian mixture on average as they are differentiating between the clusters in line with our expectations.

8. 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.

```r
library(clValid)
library(mclust)
validation <- clValid(as.matrix(slp_data_munged),2:5,validation ="internal",
                      clMethods = c("model","kmeans","hierarchical"))
```

```
summary(validation)
```

```
##
## Clustering Methods:
##  model kmeans hierarchical
##
## Cluster sizes:
##  2 3 4 5
##
## Validation Measures:
##                                  2       3       4       5
##
## model        Connectivity  10.7393 28.6119 39.0687 67.8401
##              Dunn           0.1522  0.0633  0.0225  0.0258
##              Silhouette     0.6314  0.2588  0.1861  0.0085
## kmeans       Connectivity   8.4460 10.8960 16.1885 28.7437
##              Dunn           0.1735  0.2581  0.2562  0.1090
##              Silhouette     0.6458  0.6131  0.4932  0.3042
## hierarchical Connectivity   6.0869  6.9536 16.1885 18.6774
##              Dunn           0.3637  0.4371  0.2562  0.2836
##              Silhouette     0.6994  0.6711  0.4932  0.4440
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 6.0869 hierarchical 2
## Dunn         0.4371 hierarchical 3
## Silhouette   0.6994 hierarchical 2
```

Looking at the output above, we can say that hierarchical clustering performed best on all 3 internal validation measures based on clustering results: Connectivitiy, Dunn and Silhouette. In more detail:

- While the connectivity "indicates the degree of connectedness of the clusters, as determined by the k-nearest neighbors" and has a value between 0 and infinity that should be minimized; we can see that hierarchical clustering has the lowest connectivity value.
- While 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"; we can see that hierarchical clustering has the highest value near 1.
- While the Dunn Index represents the "ratio between the smallest distance between observations not in the same cluster to the largest intra-cluster distance" and has a value between 0 and infinity that should be maximized; we can see that hierarchical clustering has the maximum Dunn Index.

9. Discuss the validation output.

If we were to compare all three models, in contrast to k-means and hierarchical clustering, gaussian model seems to have the worse performance overall as it performs significanly bad on Connectivity measure at all levels of $k, k = 2, 3, 4, 5$ while k-means and hierarchical clustering have comparable internal validation measure values. However, in between the two, in a similar vein with q8, I would choose hierarchical clustering as the optimal approach as it has better internal validation measures across all three measures - Connectivity, Silhouette and Dunn for the reasons explained above (in q8). If we have to choose a single measure to choose the optimal k, we would choose $k = 3$ clustering if we use Dunn but we would choose $k = 2$ clustering if we optimize by using Connectivity or Silhouette measure.

However, it is not always the case that we choose the most optimal clustering method. If it were the case that gaussian method had a comparatively worse performance (not too bad to the point that we get bad results), it could still have been selected as technically a "sub-optimal" clustering method if it were to perform with better speed. This is not to say that gaussian model is faster, in fact it is actually slower than others. Given that we live in the world of “big data”, speed in computations is increasingly becoming important for scientists. While gaussian model is indeed computational slower than the other two models we have - k means and hierarchical clustering -, if it were the case that it was indeed faster, it could have been chosen as the “sub-optimal” clustering method.

Another reason for choosing a “sub-optimal” cluster could be the issue of consistencty. More specifically, if it were the case that a clustering method was producing very different results for different seeds, we would prefer to choose “sub-optimal” clustering method that has worse performance measures but better consistency when used with different seeds.