Problem Set 3

Di Tong

10/22/2019

```
knitr::opts_chunk$set(echo = TRUE)
```

1. Load the state legislative professionalism data from the folder.

```
load("/Users/ditong/Documents/uml/Problem-Set-3/State Leg Prof Data & Codebook/legprof-components.v1.0.
```

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)

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.1.0       v purrr   0.3.0
## v tibble  2.0.1       v dplyr   0.8.0.1
## v tidyr   0.8.2       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

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

```
lf <- x %>%
  filter(sessid == '2009/10') %>%
  drop_na() %>%
  select(state, t_slength, slength, salary_real, expend)

lf_std <- as.data.frame(scale(lf[2:5]))
state <- lf$state
```

3. Perform quick EDA visually or numerically and discuss the patterns you see.

```
library(skimr)
```

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

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(ggplot2)
library(GGally)
```

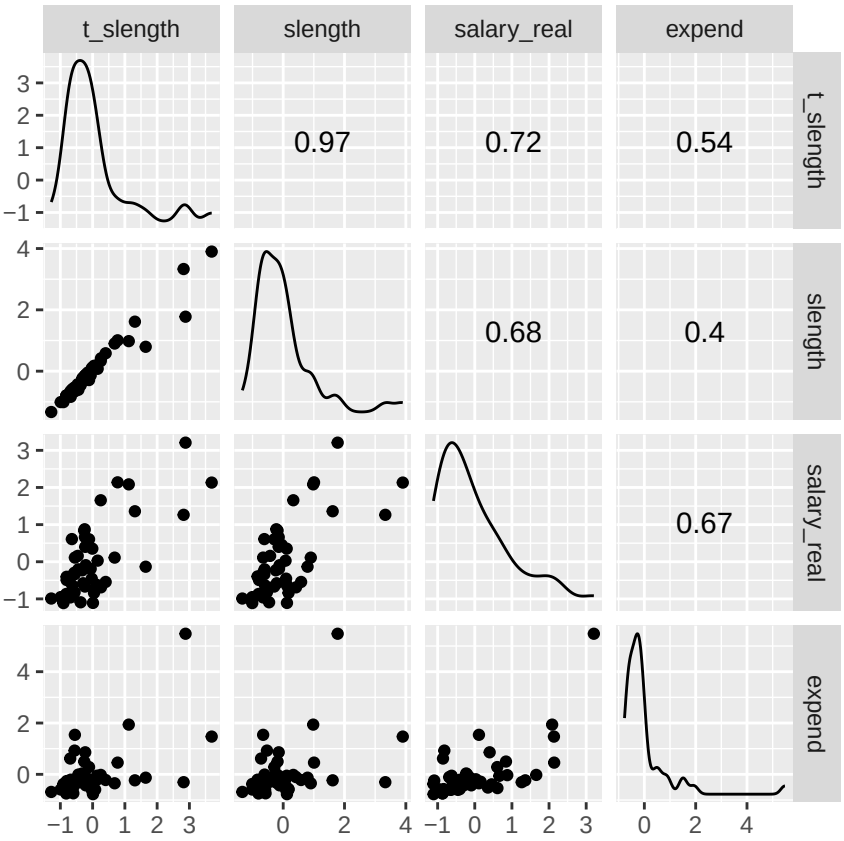
```
##
## Attaching package: 'GGally'
```

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

```
library(grid)

#summarize data
skim(lf_std)
```

```
## Skim summary statistics
##  n obs: 49
##  n variables: 4
##
## -- Variable type:numeric ---------------------------------------------------------------------------------------
##     variable missing complete  n     mean sd    p0   p25   p50    p75 p100
##       expend       0       49 49 -3.7e-17  1 -0.77 -0.54 -0.24 -0.022 5.48
##  salary_real       0       49 49  2.2e-17  1 -1.11 -0.71 -0.3   0.45  3.21
##      slength       0       49 49 -1.6e-16  1 -1.33 -0.62 -0.21  0.17  3.9
##    t_slength       0       49 49 -7.8e-17  1 -1.28 -0.6  -0.24  0.13  3.69
##      hist
##
##
##
##
```

```
# scatterplot matrix
ggscatmat(lf_std)
```

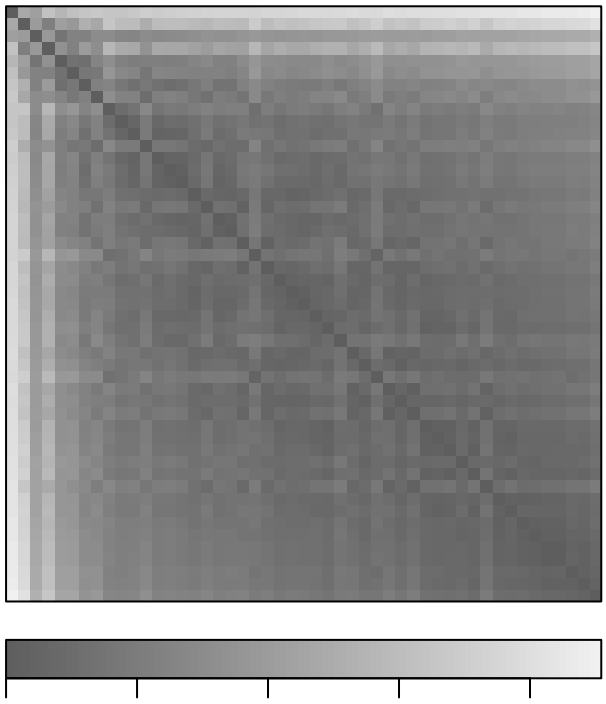


From the scatterplot matrix above, it seems that judging from all pairs of features,data points seems to gather closely in one cluster with relatively

smaller values for both features, yet there seems to always be 6-9 outlier data points that distribute further away from the cluster. The clustering pattern might be very different if these outliers are removed from the data.

4. 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.

```
library(seriation)

# calculate (and store) distance matrix
dist <- dist(lf_std, method = "euclidean")
# generate ODI
dissplot(dist)
```



The ODI does not exhibit a salient and clear pattern of clusterability, though there seems to be a large cluster at the lower right and a vaguely small one at the upper left.

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.

```
library(factoextra)
```

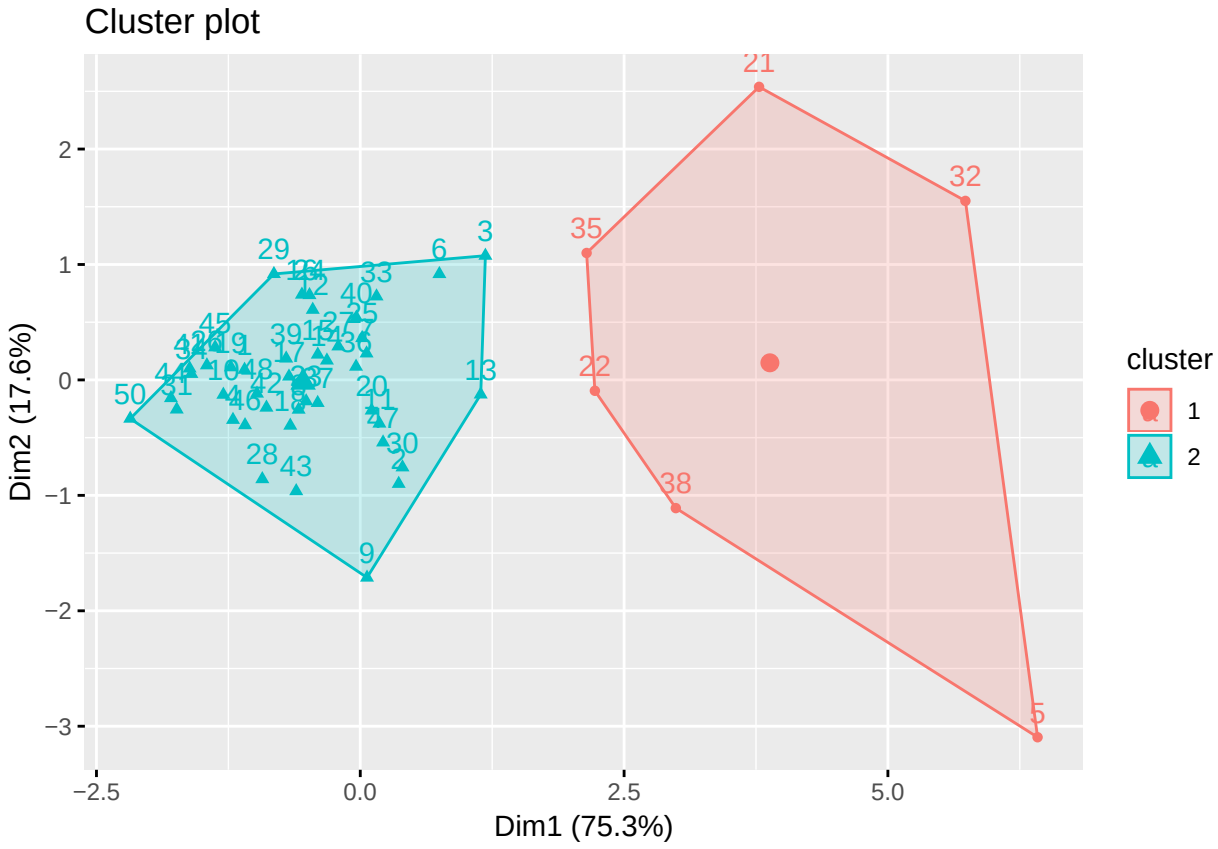
```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

```
library(ggrepel)

# fit the algorithm
set.seed(666)
kmeans <- kmeans(lf_std,
                 centers  = 2,
                 nstart = 15)
# show the summary
```

```
str(kmeans)
```

```
## List of 9
##  $ cluster     : Named int [1:49] 2 2 2 2 1 2 2 2 2 2 ...
##   ..- attr(*, "names")= chr [1:49] "1" "2" "3" "4" ...
##  $ centers     : num [1:2, 1:4] 2.1 -0.293 2.101 -0.293 2.031 ...
##   ..- 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] 40.4 48.4
##  $ tot.withinss: num 88.7
##  $ betweenss   : num 103
##  $ size        : int [1:2] 6 43
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
# visualize the cluster
fviz_cluster(kmeans, lf_std)
```




```
# save clusters in df
state$km_cluster <- as.factor(kmeans$cluster)
```

```
## Warning in state$km_cluster <- as.factor(kmeans$cluster): Coercing LHS to a
## list
```

```r
# print states in cluster 1
state[which(state$km_cluster ==1)]
```

```
## [[1]]
## [1] "California"
##
## [[2]]
## [1] "Massachusetts"
##
## [[3]]
## [1] "Michigan"
##
## [[4]]
## [1] "New York"
##
## [[5]]
## [1] "Ohio"
##
## [[6]]
## [1] "Pennsylvania"
```

The data points are divided into two clusters with size 43 and 6, each. This pattern corresponds to the initial pattern emerged in EDI that there is a large cluster that contain most of the data points and a small group of outliers that distribute away from the majority.

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
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
##
## Attaching package: 'mixtools'
## The following object is masked from 'package:grid':
##
##     depth
```

```r
library(plotGMM)

# fit the algorithm
set.seed(666)
gmm <- normalmixEM(as.matrix(lf_std[,2,4]), k = 2)
```

```
## number of iterations= 43
```

```r
# show the summary
summary(gmm)
```

```
## summary of normalmixEM object:
##           comp 1   comp 2
## lambda  0.797939 0.202061
## mu     -0.339554 1.340877
## sigma   0.447615 1.342940
## loglik at estimate:  -54.59596
```

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

```
##
##  1  2
## 41  8
```

```r
# save clusters in df
state$gmm_cluster <- as.factor(posterior$component)

# print states in cluster 2
state[which(state$gmm_cluster ==2)]
```

```
## [[1]]
## [1] "Arizona"
##
## [[2]]
## [1] "California"
##
## [[3]]
## [1] "Colorado"
##
## [[4]]
## [1] "Massachusetts"
##
## [[5]]
## [1] "Michigan"
##
## [[6]]
## [1] "New York"
##
## [[7]]
## [1] "Ohio"
##
## [[8]]
## [1] "Pennsylvania"
```

As I encountered error when initially using all features to fit the model, which I suspect might be associated with the high correlation between t_slength and slength. Therefore I fit a model that exclude the t_slength. The resulting pattern seems pretty similar to that of the k-means algorithm, though GMM, as a soft partitioning method, allows more flexibility for one data point to be assigned to more than one groups. Cluster 1 includes 41 states and cluster 2 includes 8 states.

7. Fit one additional partitioning technique of your choice (e.g., PAM, CLARA, fuzzy C- means, DBSCAN, etc.), and present and discuss results. Here again initialize at k=2.

```r
library(cluster)

# fit PAM algorithm
set.seed(123)
pam <- pam(lf_std, k = 2)
# summarize clusters
table(pam$cluster)
```

```
##
```

```
##  1  2
## 42  7
```

```
# save clusters in df
state$pam_cluster <- as.factor(pam$cluster)

# print states in cluster 2
state[which(state$pam_cluster ==2)]
```

```
## [[1]]
## [1] "California"
##
## [[2]]
## [1] "Illinois"
##
## [[3]]
## [1] "Massachusetts"
##
## [[4]]
## [1] "Michigan"
##
## [[5]]
## [1] "New York"
##
## [[6]]
## [1] "Ohio"
##
## [[7]]
## [1] "Pennsylvania"
```

The resulting pattern still does not vary much from the patterns discover from the previous two methods, especially the k means. Cluster 1 includes 42 states and cluster 2 includes 7 states.

8. Compare output of all in a visually useful, simple way (e.g., plotting by state cluster assignment across two features like salary and expenditures).

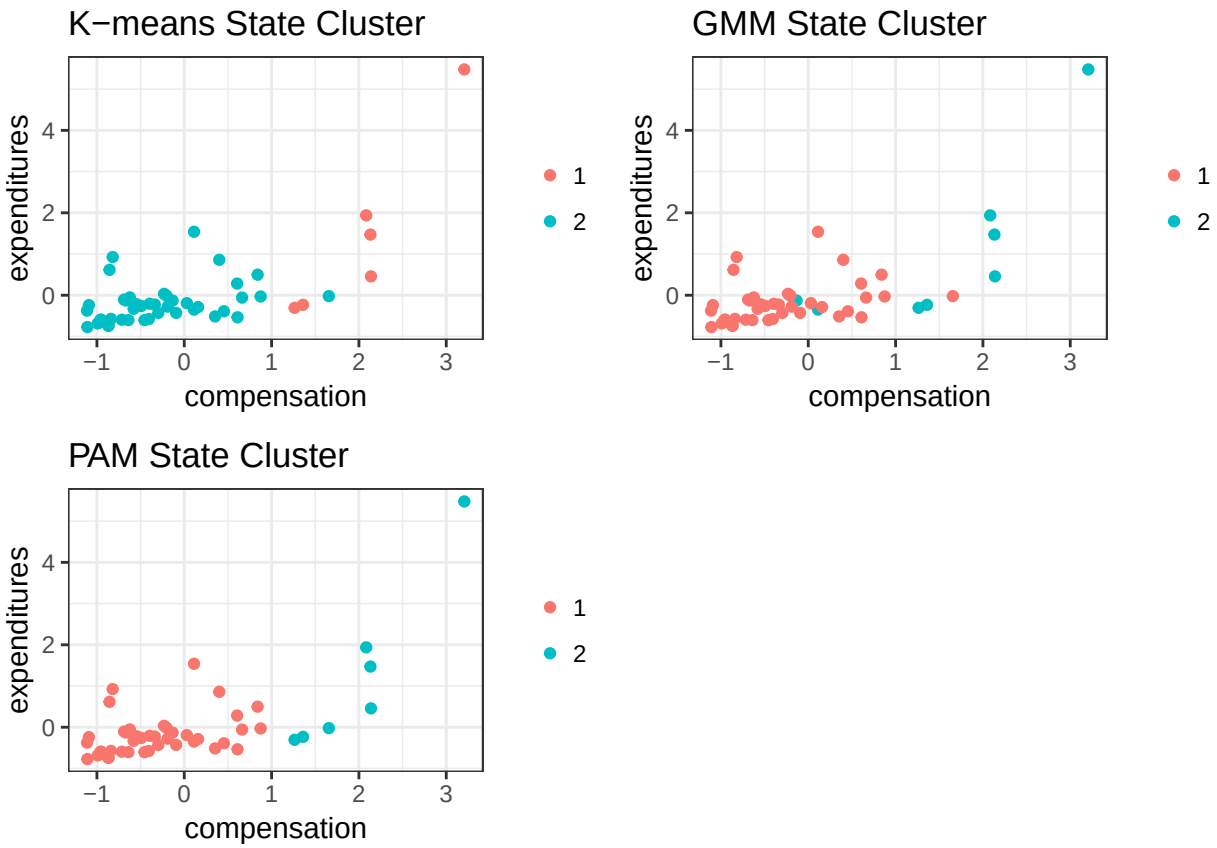
```
library(gridExtra)
```

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

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

```
# combine cluster labels
combine_data = cbind(state, lf_std)


# scatterplot
kmeans_plot <- combine_data %>%
  ggplot(aes(x = salary_real, y = expend, color = km_cluster)) + geom_point() +
  labs(x = 'compensation', y = 'expenditures', title = 'K-means State Cluster') +
  theme_bw() + theme(legend.title = element_blank())

gmm_plot <- combine_data %>%
  ggplot(aes(x = salary_real, y = expend, color = gmm_cluster)) + geom_point() +
```

```
  labs(x = 'compensation', y = 'expenditures', title = 'GMM State Cluster') +
  theme_bw() +  theme(legend.title = element_blank())

pam_plot <- combine_data %>%
  ggplot(aes(x = salary_real, y = expend, color = pam_cluster)) + geom_point() +
  labs(x = 'compensation', y = 'expenditures', title = 'PAM State Cluster') +
  theme_bw() + theme(legend.title = element_blank())

grid.arrange(kmeans_plot, gmm_plot, pam_plot, nrow=2)
```



9. 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 (k-means, GMM, X).

```
library(clValid)

internal <- clValid(lf_std, 2:10,
                    clMethods = c("kmeans","model","pam"),
                    validation = "internal")
```

```
## 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
```

```
summary(internal)
```

```
##
## Clustering Methods:
##  kmeans model pam
##
## 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
## 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
## pam    Connectivity   7.9071 21.2952 25.4798 26.3464 35.7008 42.7266 49.2762 51.2762 53.8071
##        Dunn           0.1673  0.0324  0.0332  0.0670  0.0731  0.0991  0.1019  0.1019  0.1130
##        Silhouette     0.6204  0.2530  0.2673  0.2841  0.2993  0.2682  0.2564  0.2385  0.2415
##
## Optimal Scores:
##
##              Score  Method Clusters
## Connectivity 7.9071 pam    2
## Dunn         0.2581 kmeans 3
## Silhouette   0.6458 kmeans 2
```

10. Discuss the validation output.

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

I would say, in this case, due to the special distribution pattern of the data (two quite clearly distinct groups:a small group of outliers distributed far away from the rest majority), different methods with the same number of clusters does not exhibit much difference. Moreover, owing to the low clusterability of the data, by incrementing the number of k would not end up with better clusters. From the validation results, we can see that judging from connectivity, pam performs the best at the value of 2; judeging from Dunn, k-means is the optimal with 3 clusters; judging from sihouette, k-means also performs the best at the value of 2. When method A divide the data points in a manner that looks more reasonable and meaningful according to human judgement and contain more theoretically intriguing elements, it would be a good decision to choose method A for further analysis even if it is not the optimal method judging from validation statistics.