NBA 4920/6921 Lecture 23

Unsupervised Learning: K-means Clustering

Murat Unal

11/18/2021

```r
rm(list=ls())
options(digits = 3, scipen = 999)
library(dplyr)
library(tidyverse)
library(ggplot2)
library(cluster)
library(factoextra)
library(gridExtra)
set.seed(1)
```

K-means Clustering

- Clustering is a method for finding subgroups of observations within a data set.
- When we cluster observations, we want observations in the same group to be similar and observations in different groups to be dissimilar.
- Clustering seeks to find relationships between observations without being trained by a response variable.
- K-means clustering is the simplest and the most commonly used clustering method for splitting a dataset into a set of k groups, where k represents the number of groups pre-specified by the analyst.

Clustering distance measures

- The classification of observations into groups requires some methods for computing the distance or the (dis)similarity between each pair of observations.
- The result of this computation is known as a dissimilarity or distance matrix.
- The classical methods for distance measures are **Euclidean** and **Manhattan** distances, which are defined as follow:

$$d_e(x, y) = \sqrt{(\sum_{i=1}^{n}(x_i - y_i)^2)}$$

$$d_m(x, y) = \sum_{i=1}^{n} |(x_i - y_i)|$$

Clustering algorithm

- In *k*-means clustering, each cluster is represented by its center (i.e, centroid) which corresponds to the mean of points assigned to the cluster.
- The basic idea behind *k*-means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized.

$$W(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2$$

- x_i is a data point belonging to the cluster C_k
- μ_k is the mean value of the points assigned to the cluster C_k
- Each observation x_i is assigned to a given cluster such that the sum of squares (SS) distance of the observation to their assigned cluster centers μ_k is minimized.
- We define the total within-cluster variation as follows

$$\sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2$$

K-means algorithm can be summarized as follows:

1. Specify the number of clusters k to be created.
2. Randomly assign a number, from 1 to k, to each of the observations.

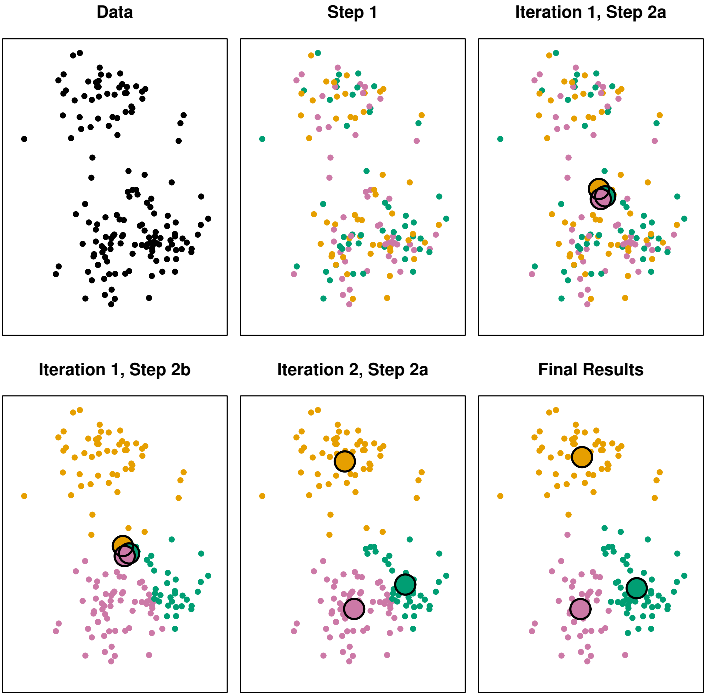
- These serve as initial cluster assignments for the observations.

3. For each of the k clusters update the cluster centroid by calculating the new mean values of all the data points in the cluster.

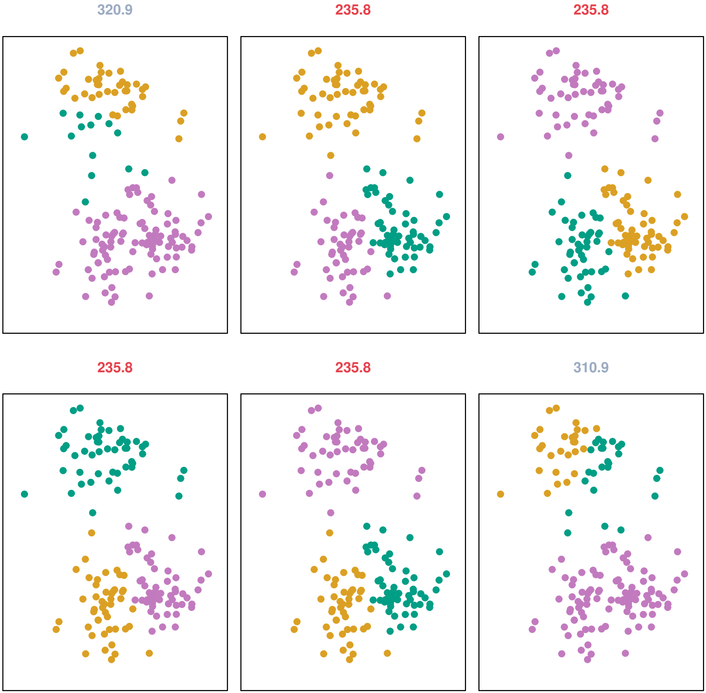
- The centroid of a kth cluster is a vector of length p containing the means of all variables for the observations in the kth cluster; p is the number of variables.

4. Iteratively minimize the total within sum of square. That is, iterate until the cluster assignments stop changing or the maximum number of iterations is reached.

- By default, R suses 10 as the default value for the maximum number of iterations.


Data
Step 1
Iteration 1, Step 2a
Iteration 1, Step 2b
Iteration 2, Step 2a
Final Results

- Because the K-means algorithm finds a local rather than a global optimum, the results obtained will depend on the initial (random) cluster assignment of each observation in Step 2 of the algorithm.
- For this reason, it is important to run the algorithm multiple times from different random initial configurations it is important to run the algorithm multiple times from different random
- In the next figure, the local optima was obtained obtained by running K-means clustering six times using six different initial cluster assignments



320.9

235.8

235.8

235.8

235.8

310.9

K-means clustering in R

- We'll use the built-in R data set `USArrests`, which contains statistics in arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973.
- It includes also the percent of the population living in urban areas

```
df <- USArrests
head(df)
```

	Murder	Assault	UrbanPop	Rape
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6
Colorado	7.9	204	78	38.7

Data preparation

To perform a cluster analysis in R, generally, the data should be prepared as follows:

1. Rows are observations (individuals) and columns are variables
2. Any missing value in the data must be removed or estimated.
3. The data must be standardized (i.e., scaled) to make variables comparable.

- Recall that, standardization consists of transforming the variables such that they have mean zero and standard deviation one.

```r
df <- na.omit(df)
df <- scale(df)
```

- We can compute k-means in R with the `kmeans` function. Here will group the data into two clusters (`centers = 2`).
- The `kmeans` function also has an `nstart` option that attempts multiple initial configurations and reports on the best one. For example, adding `nstart = 25` will generate 25 initial configurations.

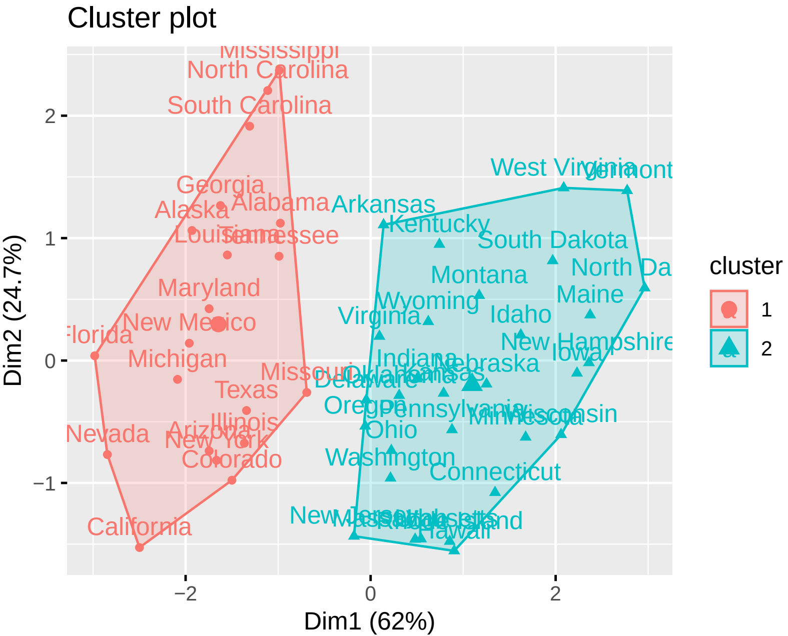
```r
k2 <- kmeans(df, centers = 2, nstart = 25)
str(k2)
```

```
List of 9
 $ cluster     : Named int [1:50] 1 1 1 2 1 1 2 2 1 1 ...
  ..- attr(*, "names")= chr [1:50] "Alabama" "Alaska" "Ariz
 $ centers     : num [1:2, 1:4] 1.005 -0.67 1.014 -0.676 0.
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:4] "Murder" "Assault" "UrbanPop" "Rape"
 $ totss       : num 196
 $ withinss    : num [1:2] 46.7 56.1
 $ tot.withinss: num 103
```

- The output of `kmeans` is a list with several bits of information. The most important being:
- `cluster`: A vector of integers (from `1:k`) indicating the cluster to which each point is allocated.
- `centers`: A matrix of cluster centers.
- `totss`: The total sum of squares.
- `withinss`: Vector of within-cluster sum of squares, one component per cluster.
- `tot.withinss`: Total within-cluster sum of squares, i.e. `sum(withinss)`
- `betweenss`: The between-cluster sum of squares, i.e. $totss - tot.withinss$.
- `size`: The number of points in each cluster.

- We can also view our results by using `fviz_cluster`.
- This provides a nice illustration of the clusters. If there are more than two dimensions (variables) `fviz_cluster` will perform principal component analysis (PCA) and plot the data points according to the first two principal components that explain the majority of the variance.

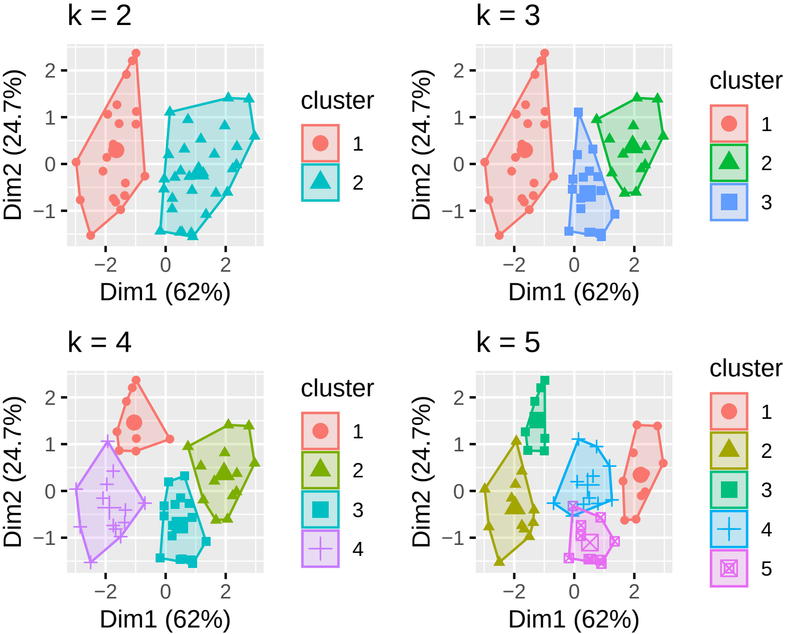
```
fviz_cluster(k2, data = df)
```



- Because the number of clusters (k) must be set before we start the algorithm, it is often advantageous to use several different values of k and examine the differences in the results.

- We can execute the same process for 3, 4, and 5 clusters

```r
k3 <- kmeans(df, centers = 3, nstart = 25)
k4 <- kmeans(df, centers = 4, nstart = 25)
k5 <- kmeans(df, centers = 5, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = df) + ggtitle
p2 <- fviz_cluster(k3, geom = "point",  data = df) + ggtitl
p3 <- fviz_cluster(k4, geom = "point",  data = df) + ggtitl
p4 <- fviz_cluster(k5, geom = "point",  data = df) + ggtitl
```



k = 2
k = 3
k = 4
k = 5
Dim2 (24.7%)
Dim1 (62%)
−1
0
1
2
−2
0
2
cluster
1
2
3
4
5

Determining optimal clusters

- Recall we need to specify the number of clusters to use; ideally we would like to use the optimal number of clusters
- The **elbow method** can help us determine this
- Recall our goal is to define clusters such that the total intra-cluster variation is minimized:

$$min(\sum_{k=1}^{k} W(C_k))$$

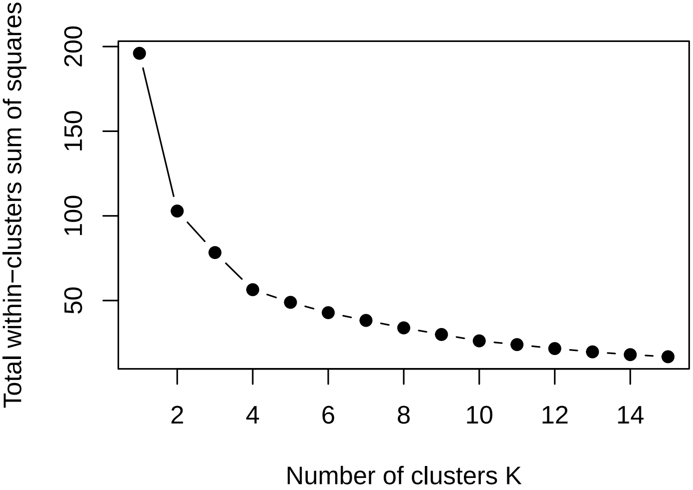
- where C_k is the kth cluster and $W(C_k)$ is the within-cluster variation

- We can use the following algorithm to define the optimal clusters:

1. Compute clustering algorithm for different values of k. For instance, by varying k from 1 to 10 clusters.
2. For each k, calculate the total within-cluster sum of square
3. Plot the curve of wss according to the number of clusters k.
4. The location of a bend in the plot is generally considered as an indicator of the appropriate number of clusters.

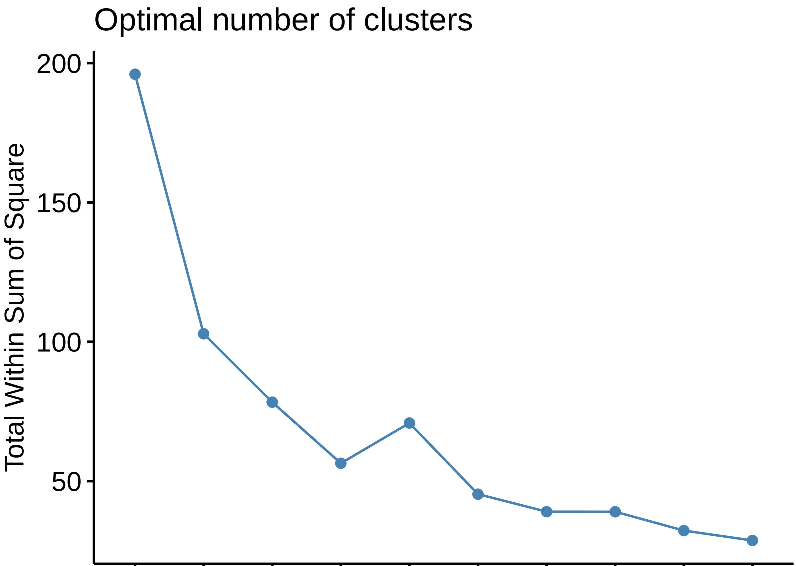
```
# function to compute total within-cluster sum of square
wss <- function(k) {
  kmeans(df, k, nstart = 25 )$tot.withinss
}

# Compute and plot wss for k = 1 to k = 15
k.values <- 1:15

# extract wss for 2-15 clusters
wss_values <- map_dbl(k.values, wss)
```

- The results suggest that 4 is the optimal number of clusters as it appears to be the bend in the elbow.



- This process to compute the **elbow method** has been wrapped up in a single function `fviz_nbclust`

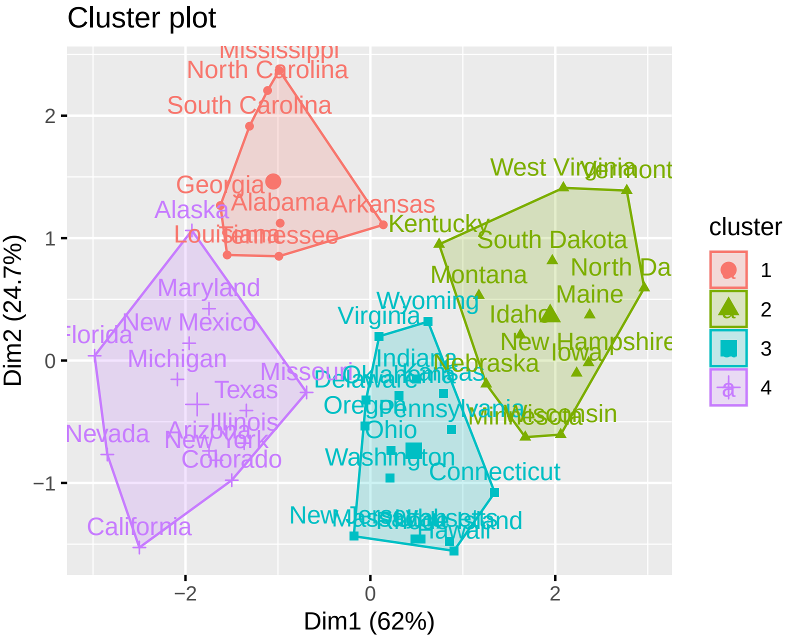
```
fviz_nbclust(df, kmeans, method = "wss")
```



- As the number of optimal clusters is suggested to be 4, we can perform the final analysis and extract the results using 4 clusters.

```r
final <- kmeans(df, 4, nstart = 25)
```

```
fviz_cluster(final, data = df)
```



- We can extract the clusters and add to our initial data to do some descriptive statistics at the cluster level

```
USArrests %>%
  mutate(Cluster = final$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
```

Cluster	Murder	Assault	UrbanPop	Rape
1	13.94	243.6	53.8	21.4
2	3.60	78.5	52.1	12.2
3	5.66	138.9	73.9	18.8
4	10.81	257.4	76.0	33.2