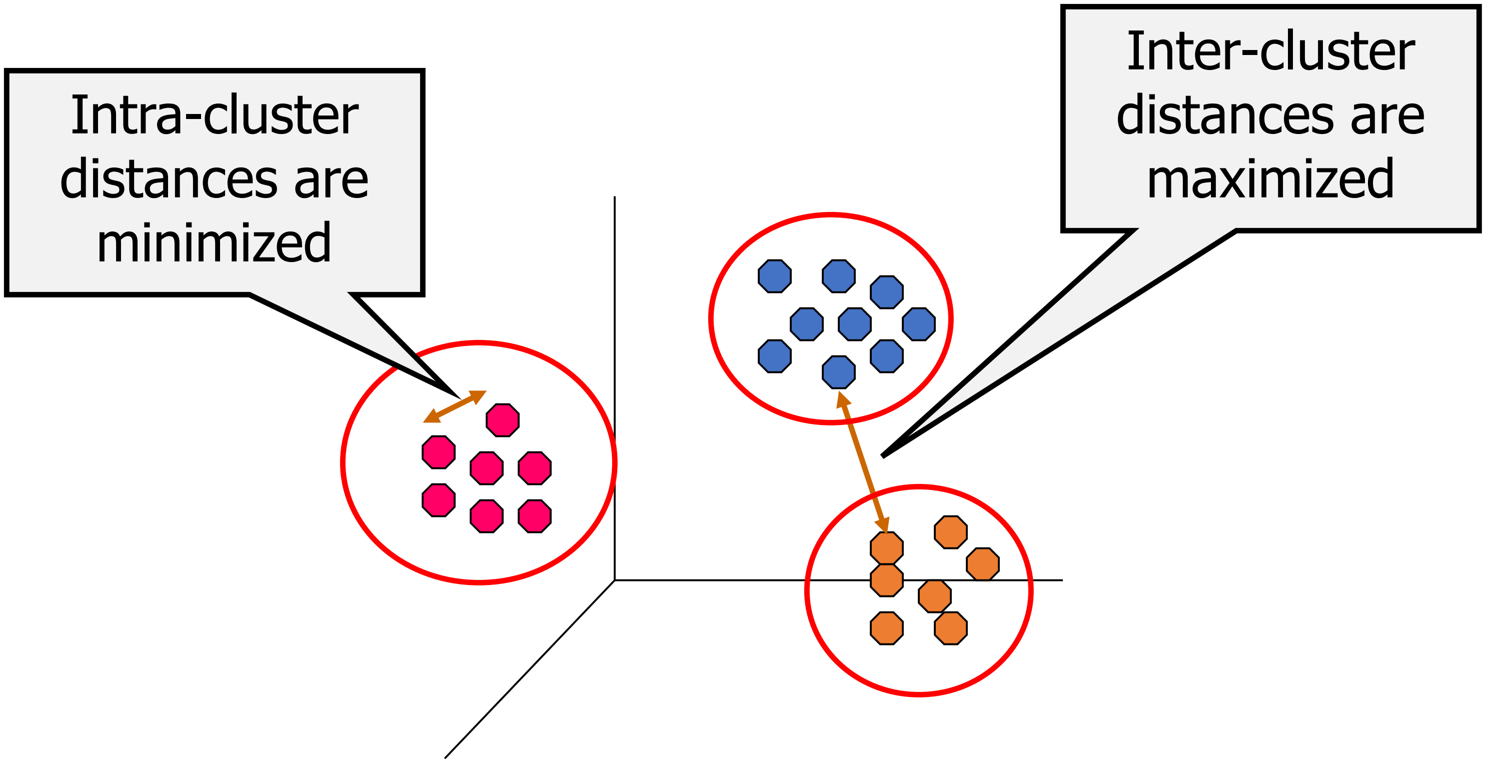
CSE 241 Data Mining

Cluster Analysis

Creating clusters

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Clusters: Definition

- Cluster is a group of similar objects (cases, points, observations, examples, members, customers, patients, locations, etc)
- Finding the groups of cases/observations/ objects in the population such that the objects are
  - Homogeneous within the group (high intra-class similarity)
  - Heterogeneous between the groups(low inter-class similarity)

Unsupervised learning technique

Goal: Form groups (clusters) of similar records

Used for **segmenting markets** into groups of similar customers

Example: Claritas segmented US neighborhoods based on demographics & income: “Furs & station wagons,” “Money & Brains”, …

Clustering methods

- Hierarchical clustering
- Partitioning based clustering

Agglomerative Methods

- Begin with n-clusters (each record its own cluster)
- Keep joining records into clusters until one cluster is left (the entire data set)
- Most popular

Divisive Methods

- Start with one all-inclusive cluster
- Repeatedly divide into smaller clusters

- To combine similar records together we need to measure the distance, similarity/dissimilarity between records and clusters

Two groups of distance measures:

- Distance between records
- Distance between clusters

Distance between numeric variables

Minkowski distance

$$d(i,j) = \sqrt[p]{|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{il} - x_{jl}|^p}$$

$$p \geq 1$$

- Properties
  - d(i, j) > 0 if i ≠ j, and d(i, i) = 0 (Positivity)
  - d(i, j) = d(j, i) (Symmetry)
  - d(i, j) ≤ d(i, k) + d(k, j) (Triangle Inequality)

Distance between scaled variables

Special cases of Minkowski distance

Manhattan distance $p = 1$

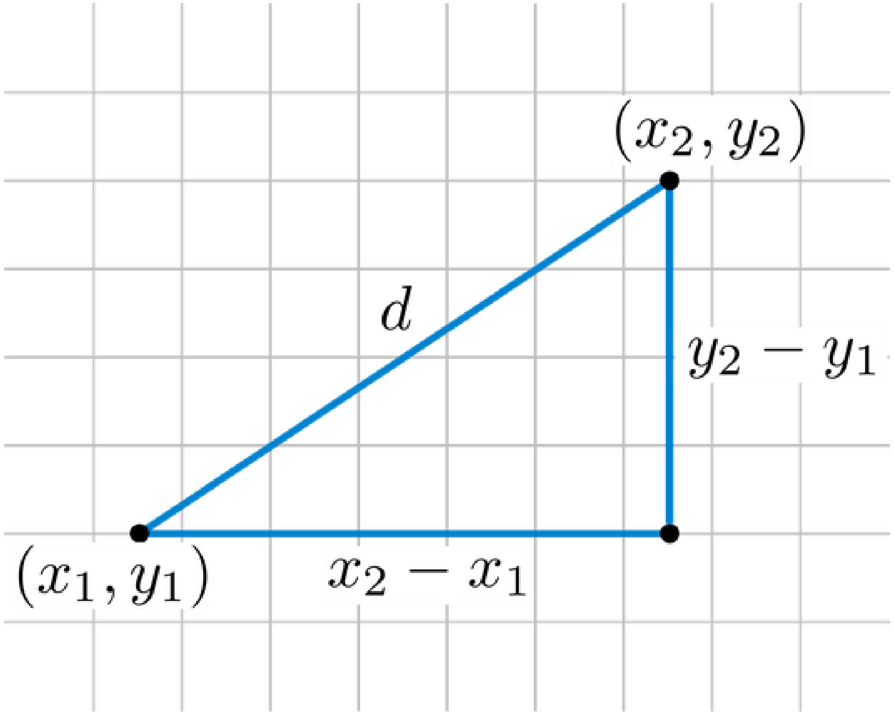
$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{il} - x_{jl}|$$

Euclidian distance $p = 2$

$$d(i,j) = \sqrt[2]{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \cdots + |x_{il} - x_{jl}|^2}$$

Euclidian distance

$$d(i,j) = \sqrt[2]{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \cdots + |x_{il} - x_{jl}|^2}$$



Between clusters distance measures

Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = min(t_{ip}, t_{jq})$

Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = max(t_{ip}, t_{jq})$

Average: Average distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = avg(t_{ip}, t_{jq})$

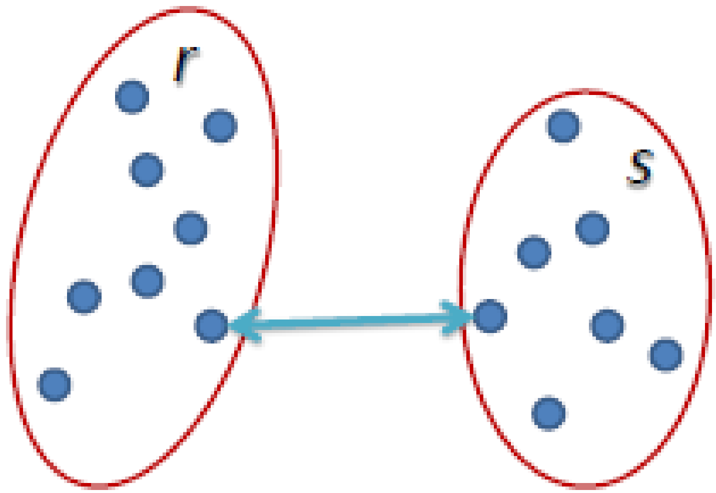
Centroid: distance between the centroids of two clusters, i.e., $dis(K_i, K_j) = dis(C_i, C_j)$

Medoid: distance between the medoids of two clusters, i.e., $dis(K_i, K_j) = dis(M_i, M_j)$

Medoid: one chosen, centrally located object in the cluster

Single Linkage

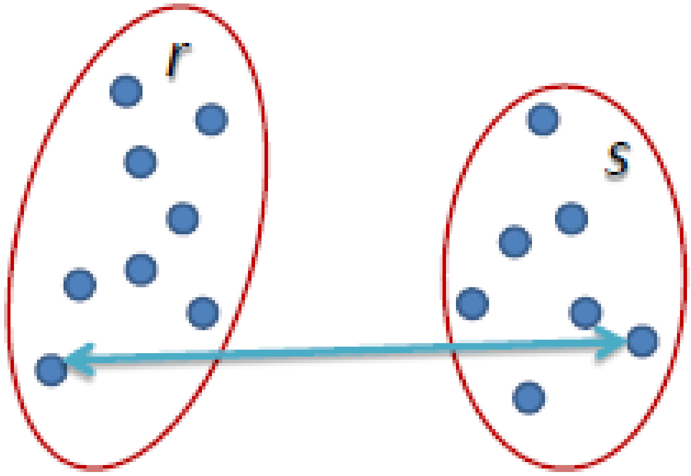
Single link: smallest distance between an element in one cluster and an element in the other, i.e.,



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Complete Linkage

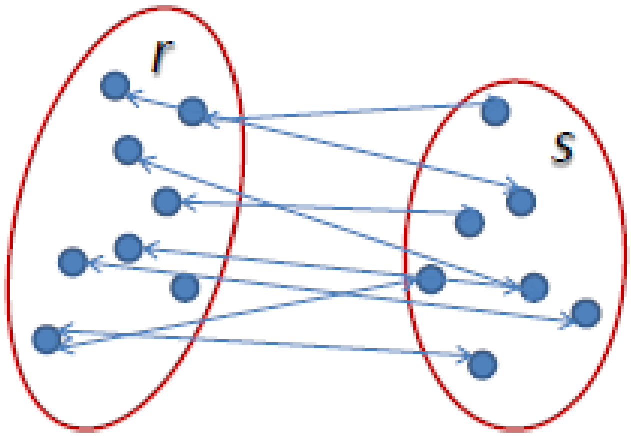
largest distance between an element in one cluster and an element in the other



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

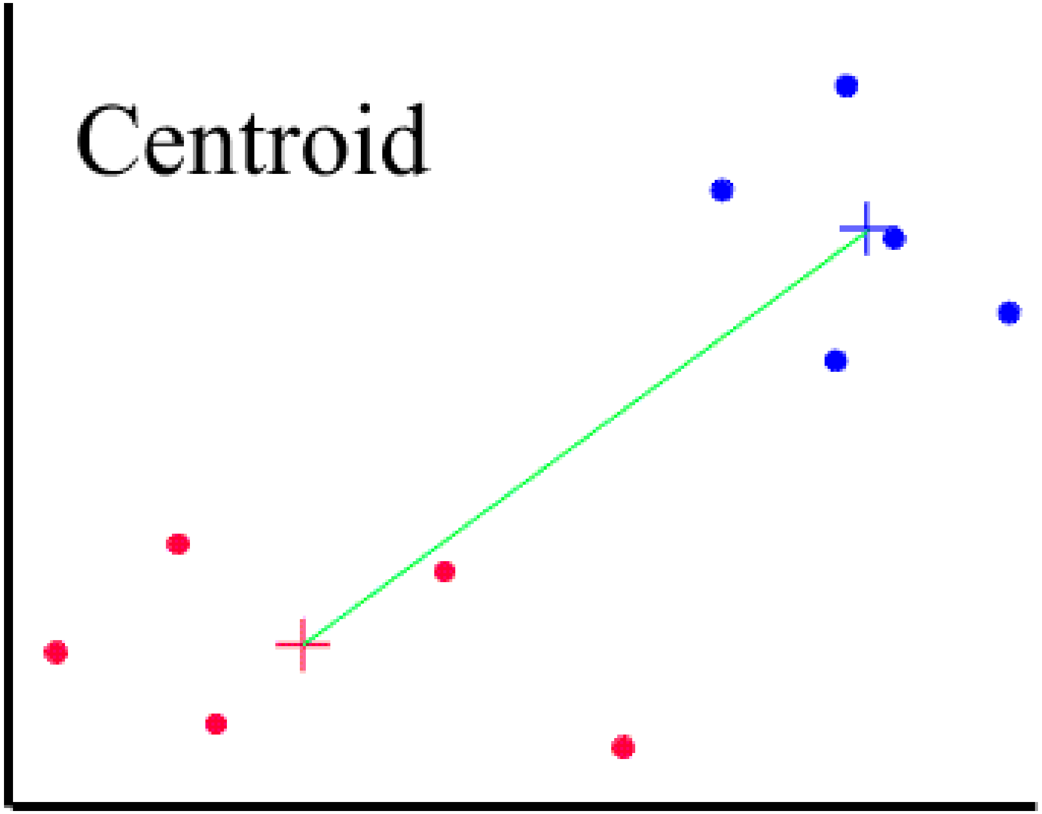
Average linkage

Average: Average distance between an element in one cluster and an element in the other, i.e.



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

distance between the centroids of two clusters, i.e., $dis(K_i, K_j) = dis(C_i, C_j)$



Doing in R: Hierarchical clustering

Taking a small sample from index dataset and calculating the distance matrix

```
index<-read.csv("index2017.csv")
rownames(index)<-index$Abbr
index1<-index[1:7,c("Unemployment", "GDP.per.Capita.PPP")]
```

Calculating the distance

```
d<-dist(index1, method="euclidian")
d
```

```
##           AFG       ALB       DZA       AGO       ARG       ARM
## ALB  9354.003
## DZA 12557.000  3203.007
## AGO  5397.000  3957.012  7160.001
## ARG 20607.000 11253.005  8050.001 15210.000
## ARM  6521.003  2833.000  6036.003  1124.034 14086.003
## AUS 45442.000 36088.002 32885.000 40045.000 24835.000 38921.001
```

The hclust function takes distance matrix as a first argument

- complete link is the default for calculating the distance between clusters

Running hierarchical cluster analysis

```
cl<-hclust(d, method="complete")
cl
```

```
##
## Call:
## hclust(d = d, method = "complete")
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 7
```

Agglomerative method

At the first step singleton clusters (clusters with only one element) 4 and 6 are merged together, Those are Armenia and Angola

```
cl$merge
```

```
##      [,1] [,2]
## [1,]   -4   -6
## [2,]   -2   -3
## [3,]   -1    1
## [4,]   -5    2
## [5,]    3    4
## [6,]   -7    5
```

```
index1
```

```
##     Unemployment GDP.per.Capita.PPP
## AFG          9.6               1947
## ALB         17.3              11301
## DZA         10.5              14504
## AGO          7.6               7344
## ARG          6.7              22554
## ARM         16.3               8468
## AUS          6.3              47389
```

Agglomerative method

Note that Armenia and Angola has the smallest distance over all the records (1124)

```
cl$merge
```

```
##      [,1] [,2]
## [1,]   -4   -6
## [2,]   -2   -3
## [3,]   -1    1
## [4,]   -5    2
## [5,]    3    4
## [6,]   -7    5
```

```
d<-dist(index1, method="euclidian")
d
```

```
##           AFG       ALB       DZA       AGO       ARG       ARM
## ALB  9354.003
## DZA 12557.000  3203.007
## AGO  5397.000  3957.012  7160.001
## ARG 20607.000 11253.005  8050.001 15210.000
## ARM  6521.003  2833.000  6036.003  1124.034 14086.003
## AUS 45442.000 36088.002 32885.000 40045.000 24835.000 38921.001
```

Agglomerative method

At the second step ALB and DZA (Albania and Algeria) are merged
At the third step AFG is merged with the cluster from the first step (ARM and Angola)
The element has positive sign when cluster is merged with the single record

```
cl$merge
```

```
##      [,1] [,2]
## [1,]   -4   -6
## [2,]   -2   -3
## [3,]   -1    1
## [4,]   -5    2
## [5,]    3    4
## [6,]   -7    5
```

```
index1
```

```
##     Unemployment GDP.per.Capita.PPP
## AFG          9.6               1947
## ALB         17.3              11301
## DZA         10.5              14504
## AGO          7.6               7344
## ARG          6.7              22554
## ARM         16.3               8468
## AUS          6.3              47389
```

Agglomerative method

The height is going to show the distance between clusters that are merged

```
cl$height
```

```
## [1]  1124.034  3203.007  6521.003 11253.005 20607.000 45442.000
```

```
cl$merge
```

```
##      [,1] [,2]
## [1,]   -4   -6
## [2,]   -2   -3
## [3,]   -1    1
## [4,]   -5    2
## [5,]    3    4
## [6,]   -7    5
```

```
d<-dist(index1, method="euclidian")
d
```

```
##           AFG       ALB       DZA       AGO       ARG       ARM
## ALB  9354.003
## DZA 12557.000  3203.007
## AGO  5397.000  3957.012  7160.001
## ARG 20607.000 11253.005  8050.001 15210.000
## ARM  6521.003  2833.000  6036.003  1124.034 14086.003
## AUS 45442.000 36088.002 32885.000 40045.000 24835.000 38921.001
```

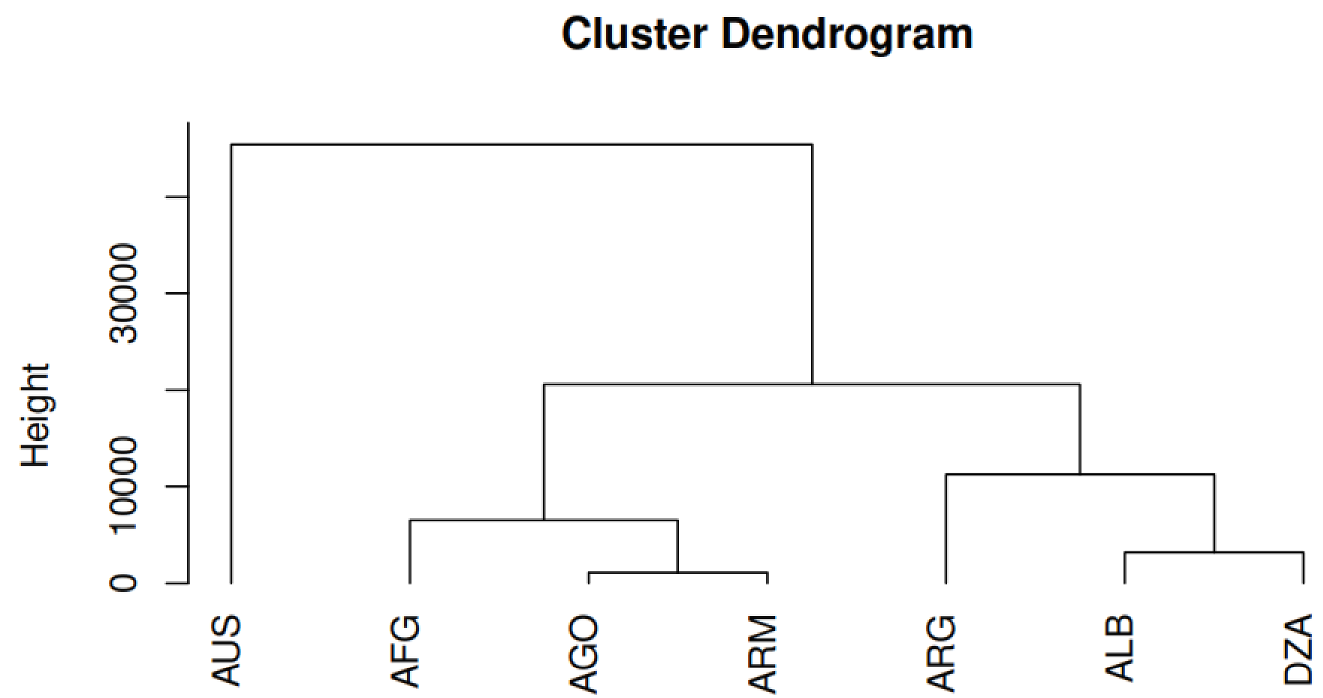
Dendrogram

Dendrogram represents how the clusters are created

```
cl$height
```

```
## [1]  1124.034  3203.007  6521.003 11253.005 20607.000 45442.000
```

```
plot(cl, hang=-1)
```



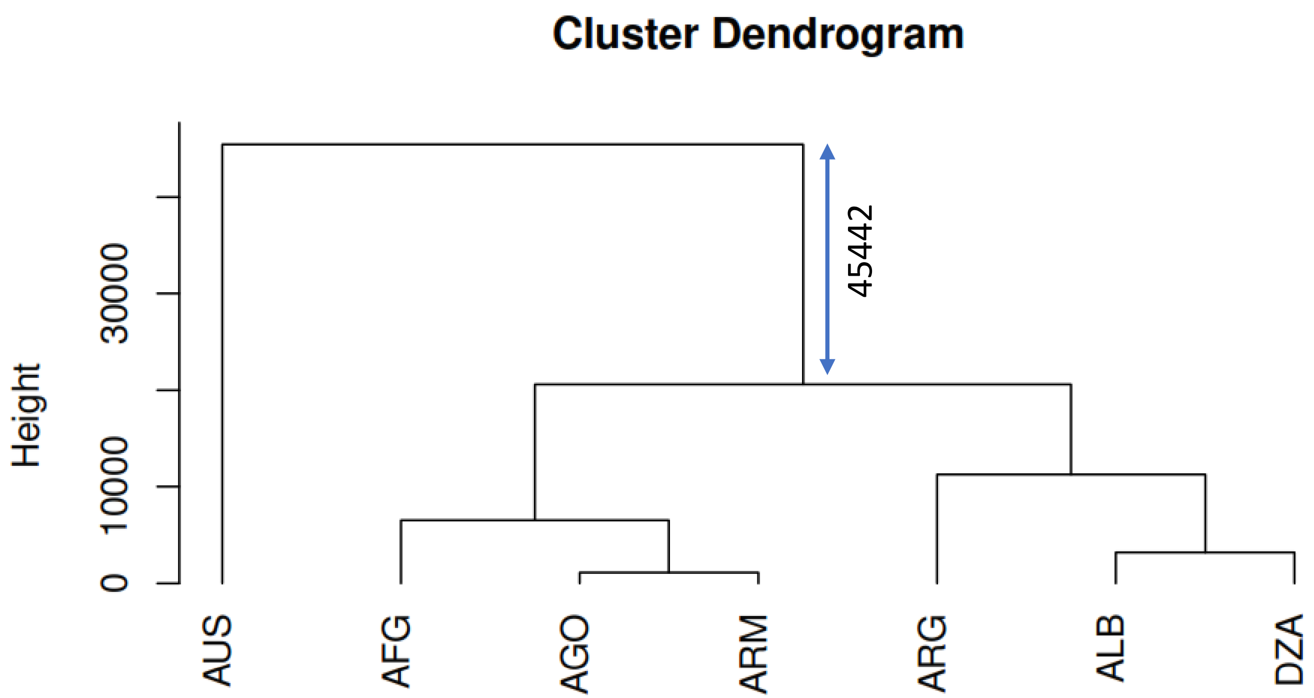
Dendrogram

Dendrogram represents how the clusters are created

```r
cl$height
```

```
## [1]  1124.034  3203.007  6521.003 11253.005 20607.000 45442.000
```

```r
plot(cl, hang=-1)
```



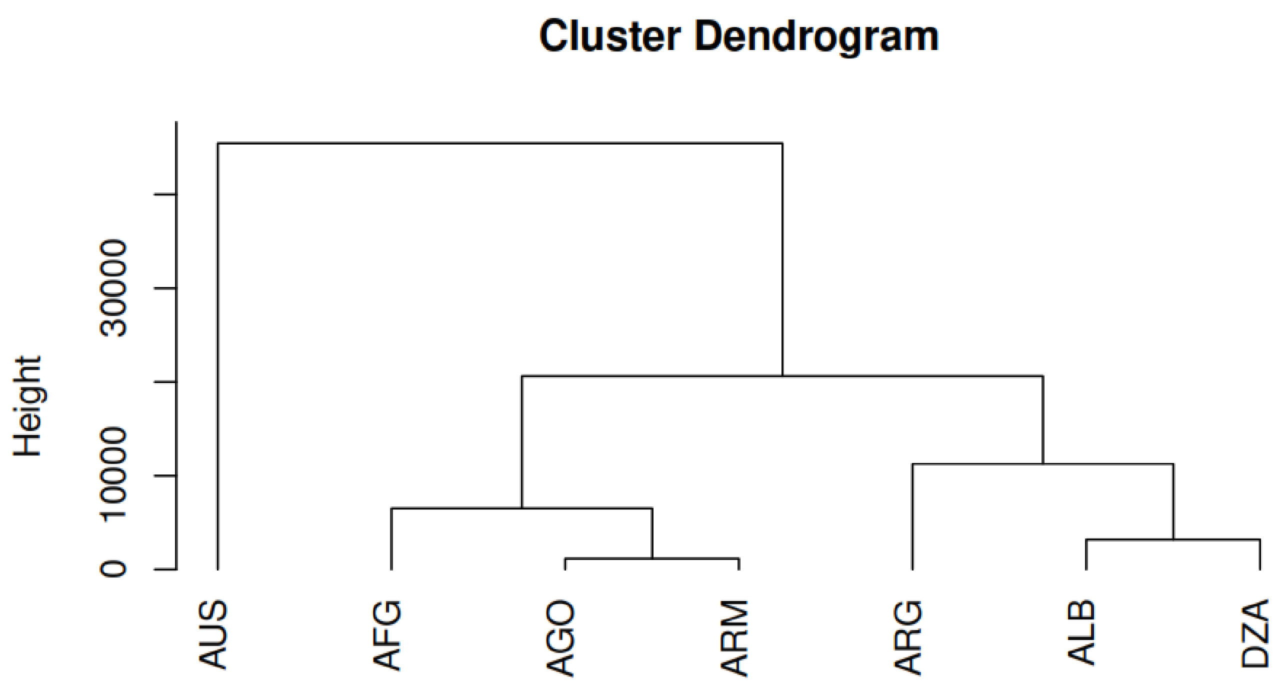
Dendrogram

How many clusters to chose ?

```r
cl$height
```

```
## [1]  1124.034  3203.007  6521.003 11253.005 20607.000 45442.000
```

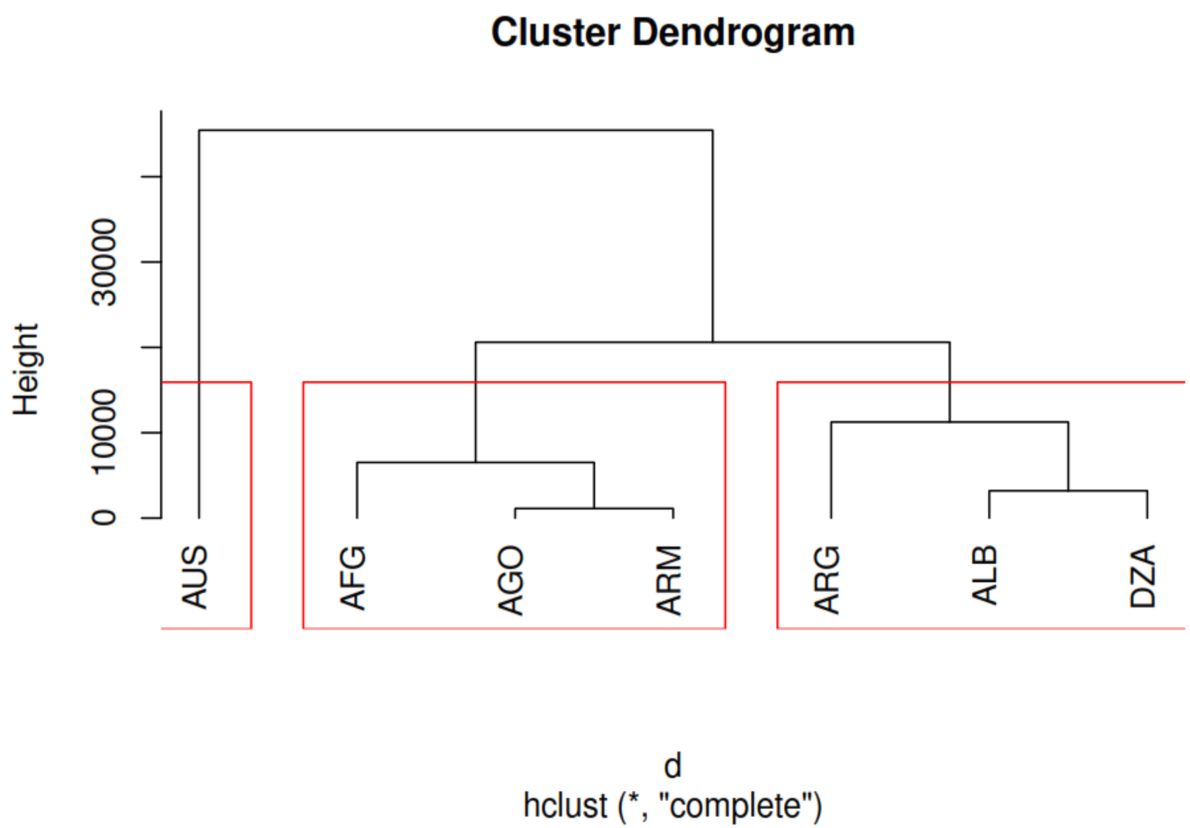
```r
plot(cl, hang=-1)
```



Dendrogram and cluster analysis

if we chose 3 clusters

```
plot(cl, hang=-1)
rect.hclust(cl, 3)
```



Cluster membership

We can create a new variable with cluster membership

Cluster membership

```
index1$cl_membership<-cutree(cl,k=3)
index1
```

```
##     Unemployment GDP.per.Capita.PPP cl_membership
## AFG          9.6               1947             1
## ALB         17.3              11301             2
## DZA         10.5              14504             2
## AGO          7.6               7344             1
## ARG          6.7              22554             2
## ARM         16.3               8468             1
## AUS          6.3              47389             3
```

Standardization

Unemployment and GDP per capita has different measurement scales.
because the scale of the per Capita GDP is larger than for the Unemployment, the distance measure is going to be strongly affected by this variable only
Solution: Standardize the variables with different measurement scales before running cluster analysis

```
index1
```

```
##     Unemployment GDP.per.Capita.PPP
## AFG          9.6               1947
## ALB         17.3              11301
## DZA         10.5              14504
## AGO          7.6               7344
## ARG          6.7              22554
## ARM         16.3               8468
## AUS          6.3              47389
```

Standardization

Use scale function to create dataframe with the z-scores

Standardizing, Z scores

```
scale(index1, center=TRUE, scale=TRUE)
```

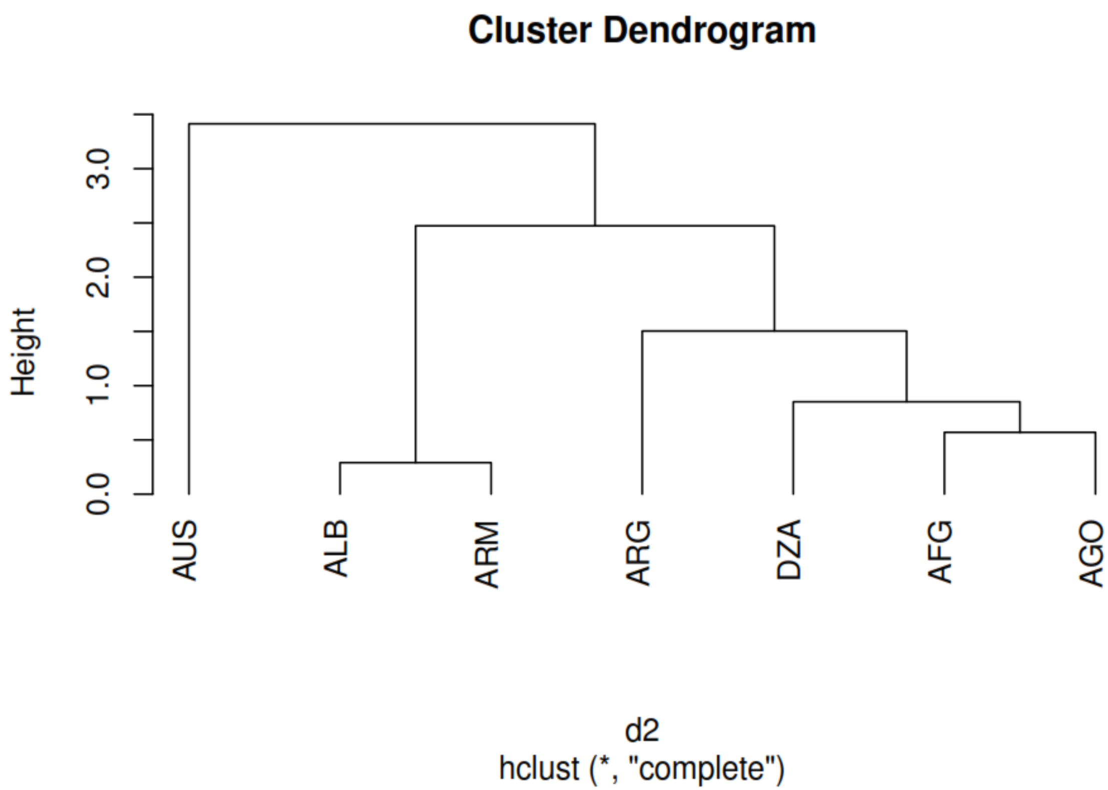
```
##      Unemployment GDP.per.Capita.PPP
## AFG  -0.22577942         -0.9403588
## ALB   1.48823619         -0.3238785
## DZA  -0.02543993         -0.1127832
## AGO  -0.67097828         -0.5846667
## ARG  -0.87131777          0.4177563
## ARM   1.26563676         -0.5105889
## AUS  -0.96035754          2.0545198
## attr(,"scaled:center")
##        Unemployment GDP.per.Capita.PPP
##            10.61429        16215.28571
## attr(,"scaled:scale")
##        Unemployment GDP.per.Capita.PPP
##            4.492374       15173.236237
```

```
index2<-as.data.frame(scale(index1, center=TRUE, scale=TRUE))
```

New clusters

```
index2<-as.data.frame(scale(index1, center=TRUE, scale=TRUE))
```

```
d2<-dist(index2, method="euclidian")
cl3<-hclust(d2, method="complete")
plot(cl3, hang=-1)
```



k-means clustering

K-Means

- Is used when the number of the clusters we are looking for is known beforehand

- We assume that the data is divided equally among the clusters

K-means algorithm

The number of clusters is defined by the user

Given *k*, the *k-means* algorithm consists of four steps:

- Select initial centroids at random.
- Assign each object to the cluster with the nearest centroid.
- Compute each centroid as the mean of the objects assigned to it.
- Repeat previous 2 steps until no change.

With this steps K-means algorithm is maximizes Between group Sum of Squares and minimizes within group Sum of Squares

Total Sum of Squares

$$SST = \sum_{j=1}^{m} \sum_{i=1}^{n} (y_{ij} - \bar{y}_m)^2$$

Where I is the number of the row/case, j number of the variables, k is the number of the groups/clusters

Between Groups Sum of Squares

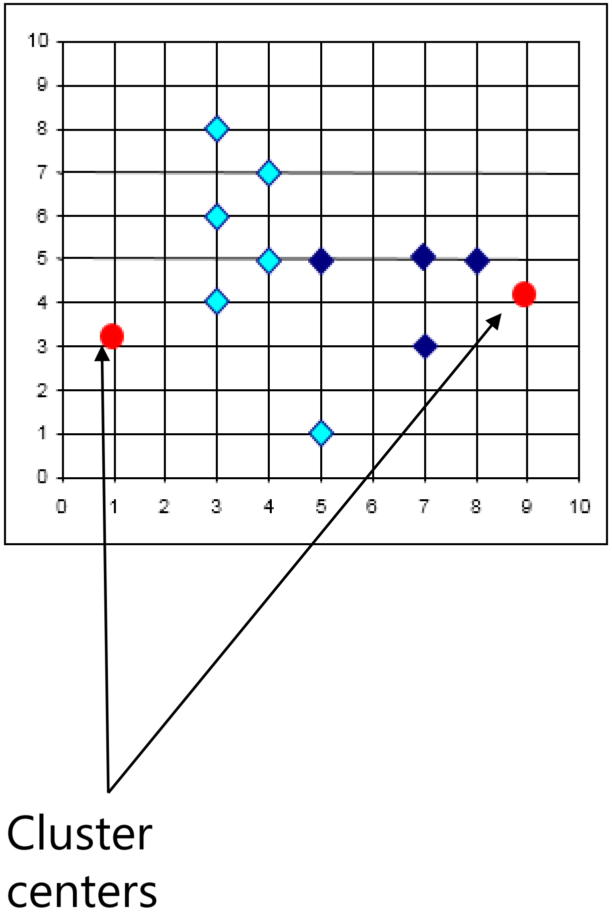
$$SSB = \sum_{l=1}^{k} \sum_{j=1}^{m} (\bar{y}_{km} - \bar{y}_m)^2$$

$$SST = SSB + SSW$$

Within group sum of squares

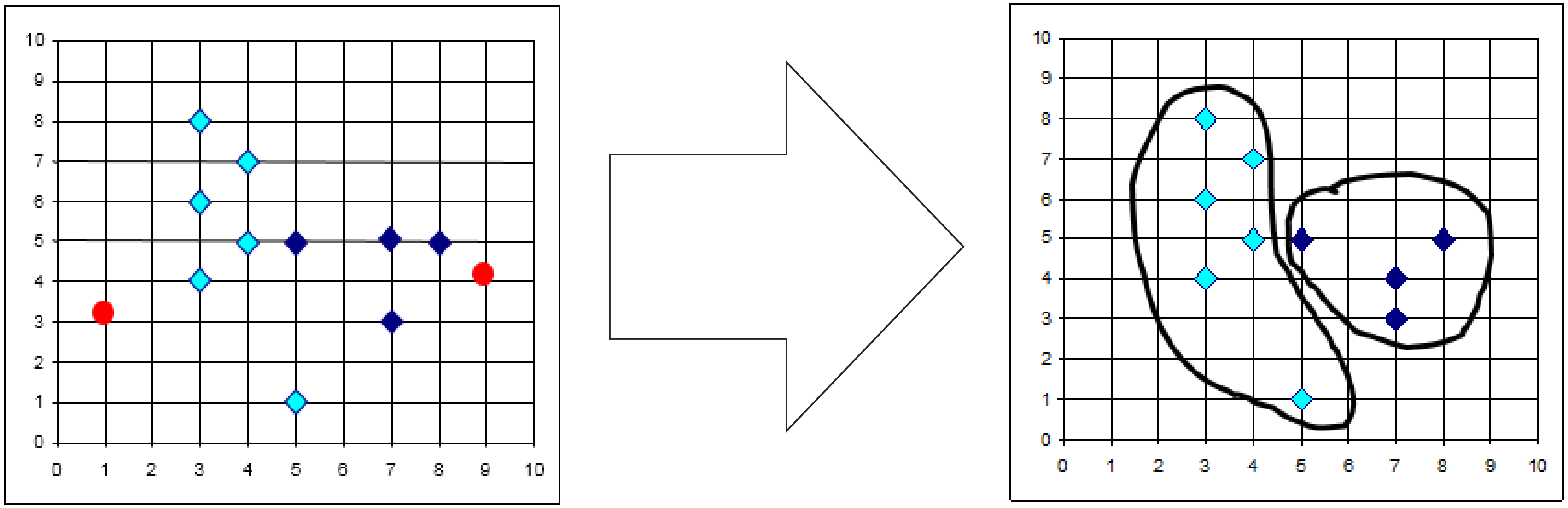
$$SSW = \sum_{l=1}^{k} \sum_{j=1}^{m} \sum_{i=1}^{n} (y_{ijk} - \bar{y}_{km})^2$$

K-means algorithm (Partitioning)



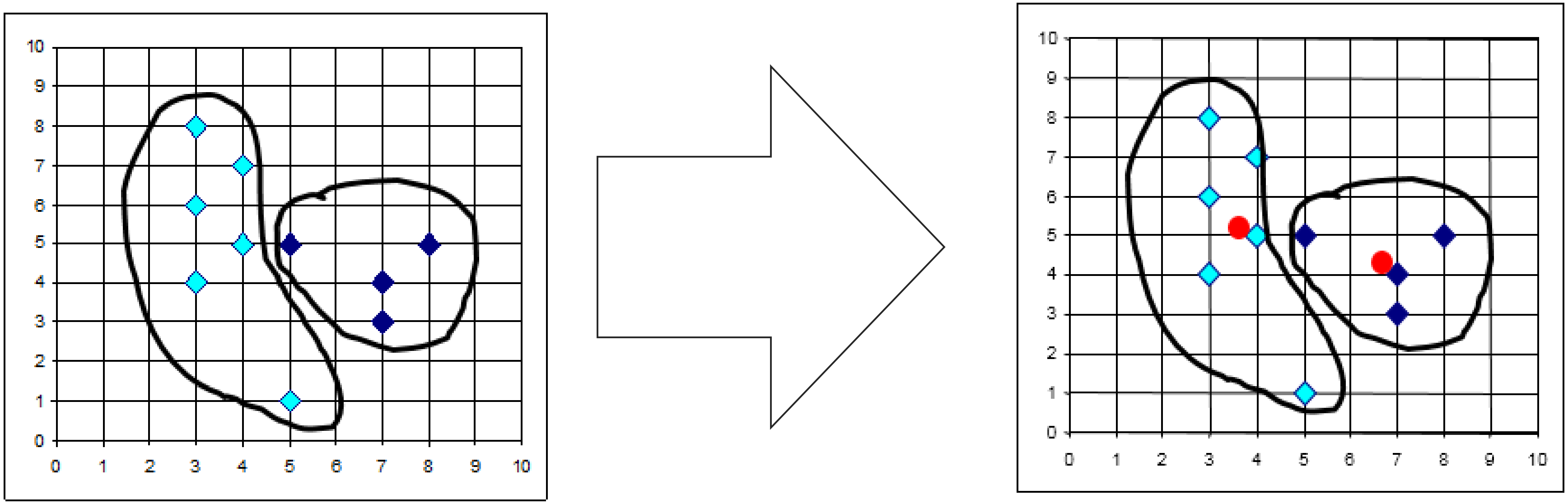
Randomly chose cluster centers

K-means algorithm (Partitioning)



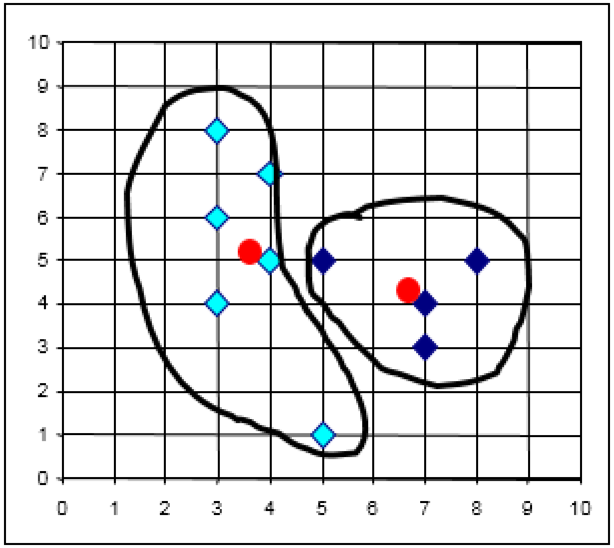
Create two clusters

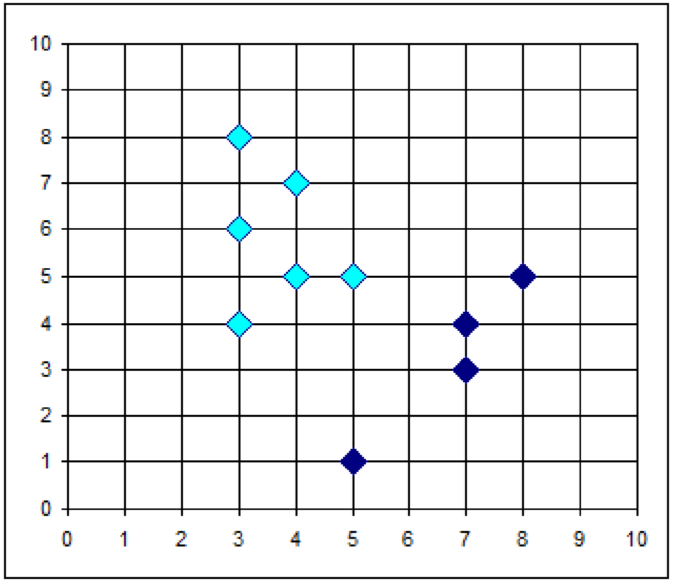
K-means algorithm (Partitioning)



Recalculate the cluster centers

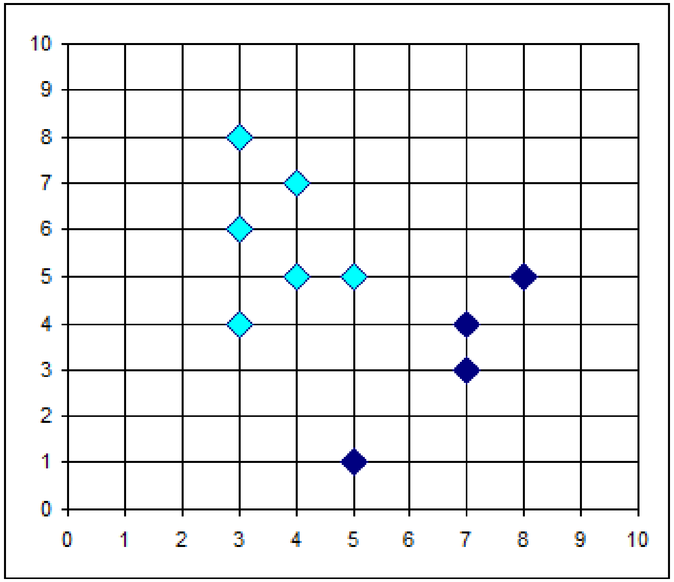
K-means algorithm (Partitioning)

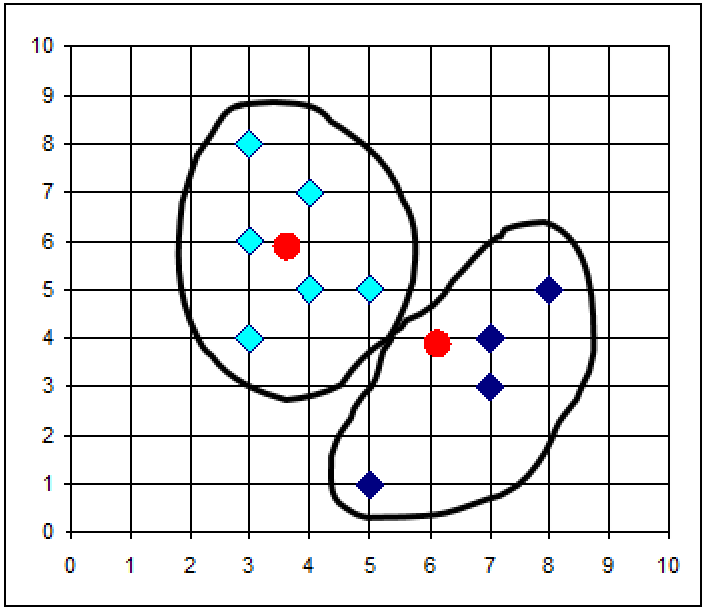




Reassign members to the clusters

K-means algorithm (Partitioning)





Recalculate cluster centers

Continue until there are no changes in clustering /the convergence criterion is met/, or after some given number of iterations

Doing in R

```r
index2<-index[,c("Unemployment", "GDP.per.Capita.PPP")]
index2<-index2[complete.cases(index2),]
index2_sc<-as.data.frame(scale(index2))
km1<-kmeans(index2_sc,3)
```

What is inside ?

```r
names(km1)
```

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

There should be no missing values

Doing in R

Shows the cluster membership of each case

```
km1$cluster
```

```
## AFG ALB DZA AGO ARG ARM AUS AUT AZE BHS BHR BGD BRB BLR BEL BLZ BEN BTN
##   1   3   1   1   1   3   2   2   1   1   2   1   1   1   2   1   1   1
## BOL BIH BWA BRA BGR BFA MMR BDI KHM CMR CAN CPV CAF TCD CHL CHN COL COM
##   1   3   3   1   1   1   1   1   1   1   2   1   1   1   1   1   1   3
## COD COG CRI CIV HRV CYP CZE DNK DJI DOM ECU EGY SLV GNQ ERI EST ETH FJI
##   1   1   1   1   3   3   1   2   3   1   1   1   1   1   1   1   1   1
## FIN FRA GAB GMB GEO DEU GHA GRC GTM GIN GNB GUY HTI HND HKG HUN ISL IND
##   2   2   3   3   1   2   1   3   1   1   1   1   1   1   2   1   2   1
## IDN IRN IRQ IRL ISR ITA JAM JPN JOR KAZ KEN KOR KWT KGZ LAO LVA LBN LSO
##   1   1   3   2   2   2   1   2   1   1   1   2   2   1   1   1   1   3
## LBR LBY LTU LUX MAC MKD MDG MWI MYS MDV MLI MLT MRT MUS MEX MDA MNG MNE
##   1   3   1   2   2   3   1   1   1   1   1   2   3   1   1   1   1   3
## MAR MOZ NAM NPL NLD NZL NIC NER NGA NOR OMN PAK PAN PNG PRY PER PHL POL
##   1   3   3   1   2   2   1   1   1   2   2   1   1   1   1   1   1   1
## PRT QAT ROU RUS RWA LCA VCT WSM STP SAU SEN SRB SLE SGP SVK SVN SLB ZAF
##   1   2   1   1   1   3   3   1   1   2   1   3   1   2   1   1   3   3
## ESP LKA SDN SUR SWZ SWE CHE TWN TJK TZA THA TGO TON TTO TUN TUR TKM UGA
##   3   1   1   1   3   2   2   2   1   1   1   1   1   2   1   1   1   1
## UKR ARE GBR USA URY UZB VUT VEN VNM YEM ZMB ZWE BRN
##   1   2   2   2   1   1   1   1   1   3   1   1   2
```

Doing in R

Shows the means/centers of each variable for each cluster

```
km1$centers
```

```
##   Unemployment GDP.per.Capita.PPP
## 1   -0.2916477         -0.4211617
## 2   -0.5249025          1.6272829
## 3    1.9010286         -0.3468012
```

Doing in R

Look at the variances

```r
# Total Sum of Squares
km1$totss
```

```
## [1] 348
```

```r
# Within group Sum of squares for each cluster
km1$withinss
```

```
## [1] 42.21621 38.53746 34.44320
```

```r
# Between group Sum of Squares
km1$betweenss
```

```
## [1] 232.8031
```

```r
sum(km1$withinss)+km1$betweenss
```

```
## [1] 348
```

Doing in R

How good is the clustering?

```
km1$betweenss/km1$totss
```

```
## [1] 0.6689745
```

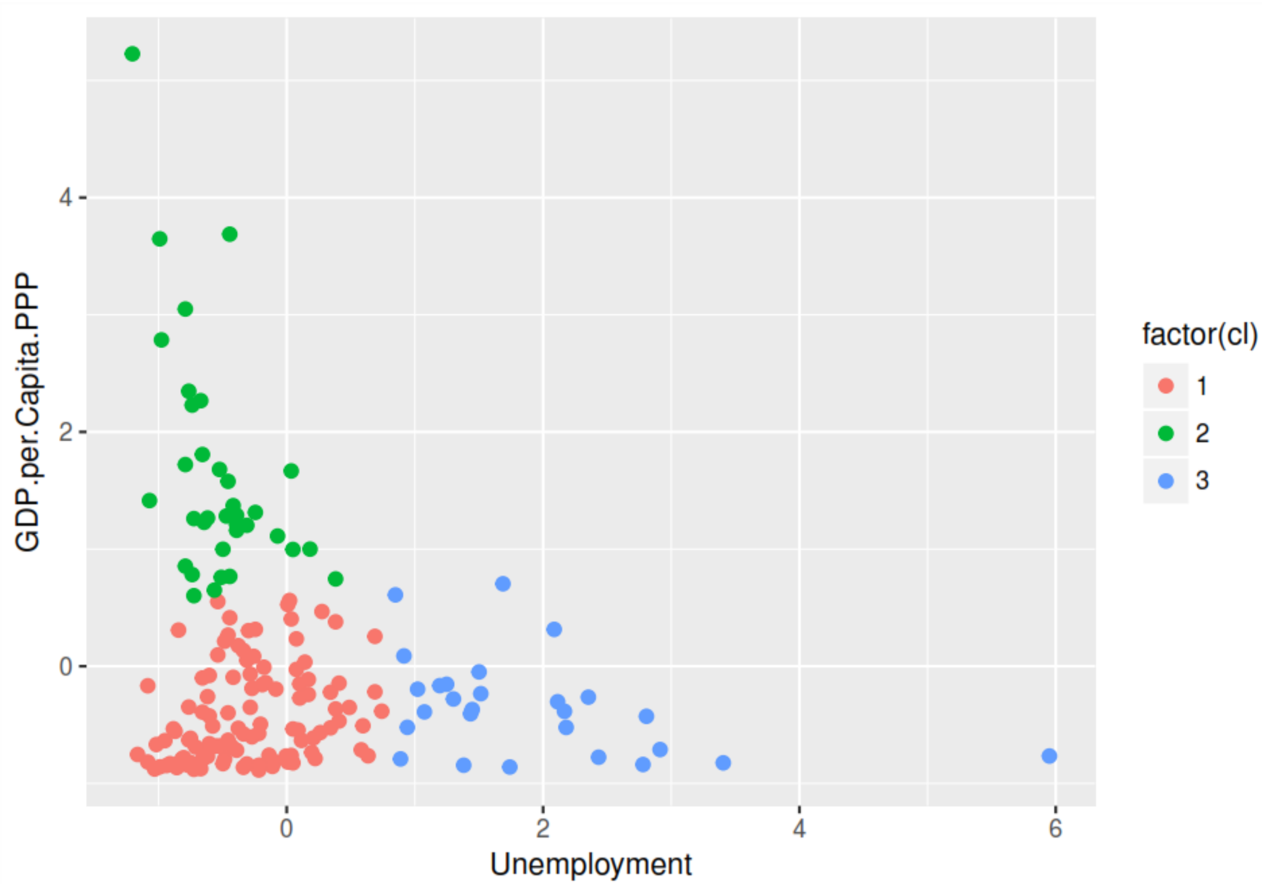
66.7% of the total variance in the data can be explained by the clusters

Doing in R

Visualizing the clusters

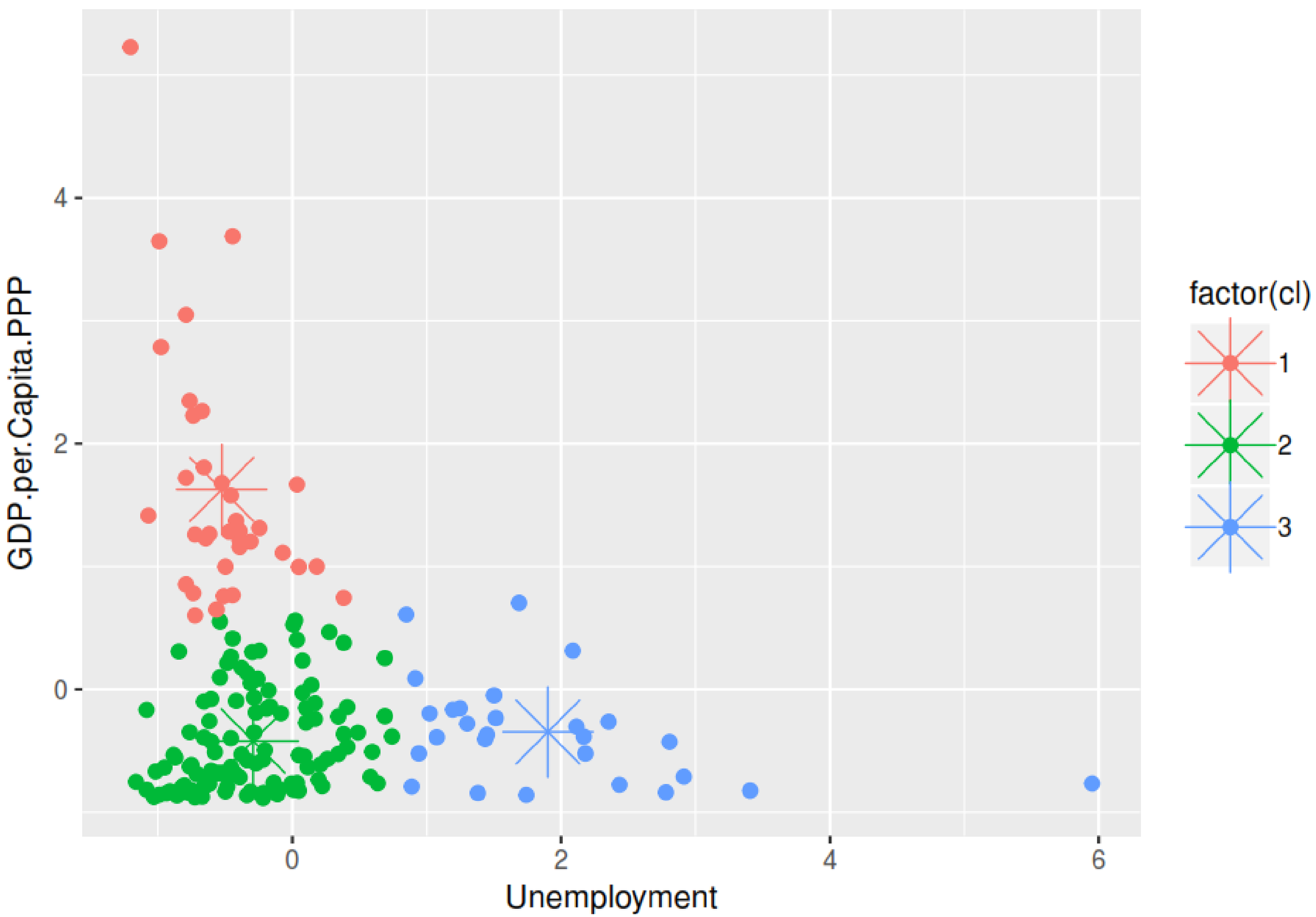
```
index2_sc$cl<-km1$cluster
```

```
ggplot(data=index2_sc, aes(x=Unemployment, y=GDP.per.Capita.PPP, col=factor(cl)))+
  geom_point(size=2)
```



Doing in R

```
ggplot(data=index2_sc, aes(x=Unemployment, y=GDP.per.Capita.PPP, col=factor(cl)))+
  geom_point(size=2)+
  geom_point(data=data.frame(km1$centers,cl=factor(1:3)), aes(Unemployment,GDP.per.Capita.PPP, col=cl),
             pch=8,size=10)
```



Determining the number of clusters

The goal is to increase the BetweenSS/TotalSS, or decrease WithinSS/TotalSS

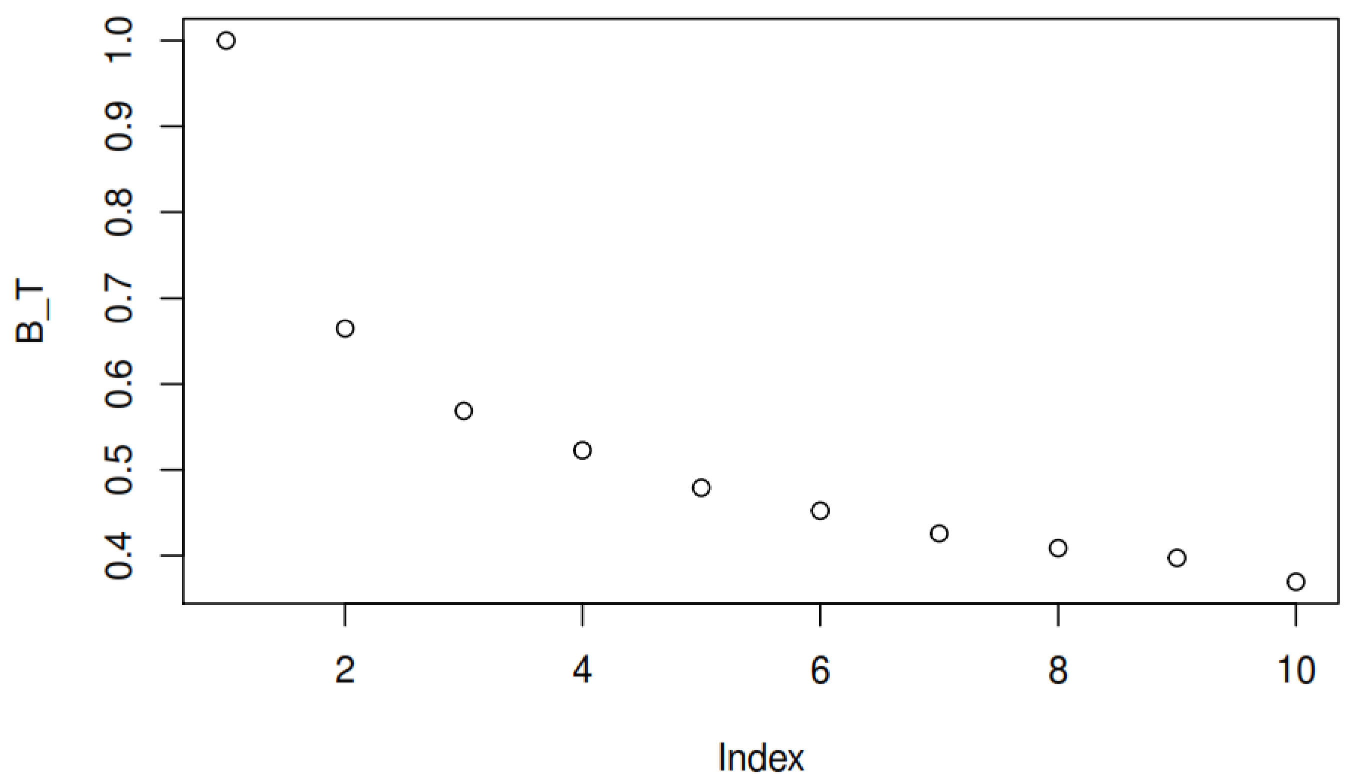
```
index<-read.csv("index2017.csv")
index1<-index[complete.cases(index),]
rownames(index2)<-index2$Abbr
index2<-index1[,c(8:19)]
index2_sc<-as.data.frame(scale(index2))
```

Make a simple loop to calculate WithinSS/TotalNSS

```
B_T<-c()
for (i in 1:10){
  set.seed(1)
  km1<-kmeans(index2_sc,i)
  B_T[i]<-km1$tot.withinss/km1$totss
}
```

Plotting the results

```
# look for the elbow
plot(B_T)
```



Using factoextra

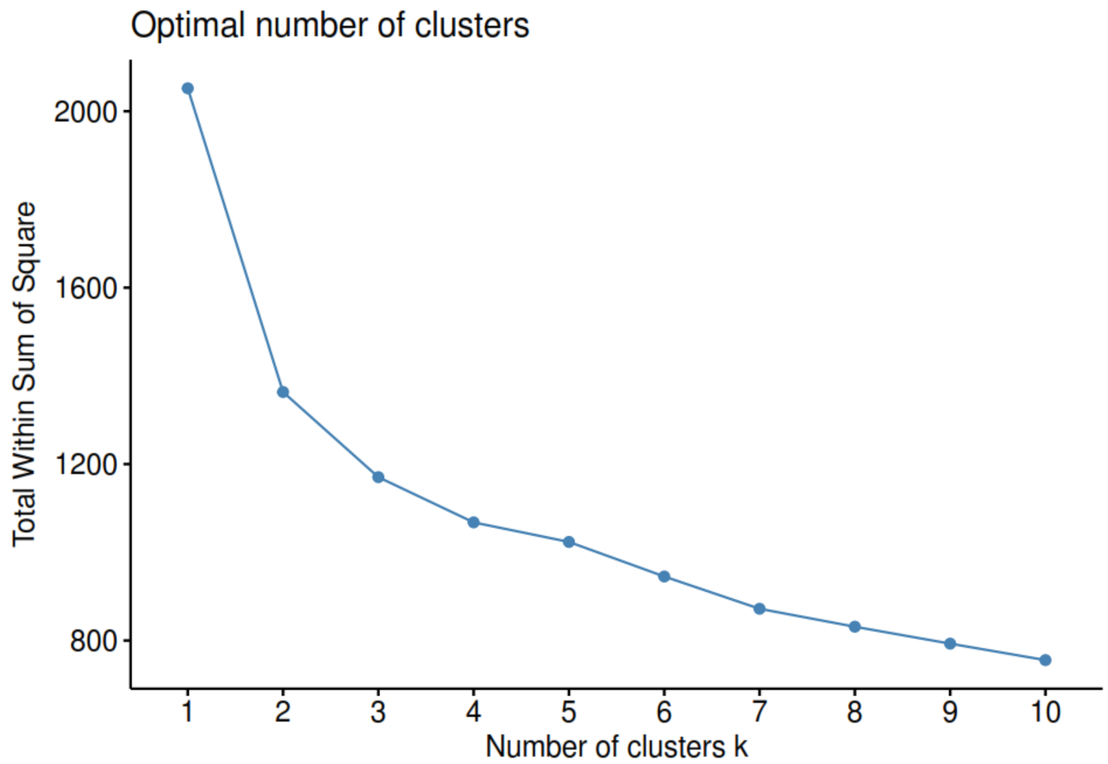
```r
# using packages
install.packages("factoextra")
```

```r
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.3.3

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

```r
set.seed(1)
fviz_nbclust(index2_sc, kmeans, method = "wss")
```



Example: NBA dataset

```
library(devtools)
install_github('Habet/CSE270')
library(SportsAnalytics270)
data(nba_misc)
```

The link

https://www.basketball-reference.com/leagues/NBA_2017.html#all_misc_stats