**#################################################**

**Unsupervised learning**

**Class 03 – Quality measures in clustering**

**Dr hab. Katarzyna Kopczewska**

**#################################################**

**# Clustering**

# recommended packages for cluster analysis

# also other packages are available

install.packages("cluster")

install.packages("factoextra")

install.packages("flexclust")

install.packages("fpc")

install.packages("clustertend")

install.packages("ClusterR")

**library(cluster)**

**library(factoextra)**

**library(flexclust)**

**library(fpc)**

**library(clustertend)**

**library(ClusterR)**

**# changing the path for accessing the Working Directory**

**# warning: change \ on /**

**getwd() # checking current WD**

**setwd("E:/My all/&Wykłady/Wykłady - WNE Unsupervised Learning/01. Clustering")**

**getwd()**

**#import of data**

**dane<-read.csv("cluster.csv", sep=";", dec=",", header=TRUE)**

**dim(dane) # checking the dimensions of the dataset**

**# data selection**

**xxx<-dane[,c(25,32)] #XA06–unemployment rate, XA31–salaries (Poland=100%)**

**dane$population.total<-dane$XA19+dane$XA20+dane$XA21**

**dane$workforce.ratio<-dane$XA20/dane$population.total**

**names(dane)**

**yyy<-dane[,c(25,32, 36)]**

Spis treści

[## 01. Prediagnostics - Optimal number of clusters 3](#_Toc526703328)

[## 02. Prediagnostics - How many clusters there should be? 4](#_Toc526703329)

[## 03. Prediagnostics - Automatic selection of number of clusters 5](#_Toc526703330)

[## 04. Prediagnostics - Assessing Clustering Tendency using Hopkins' statistic 6](#_Toc526703331)

[## 05. Prediagnostics - Visual assessment of ordered dissimilarity matrix (ODM) 7](#_Toc526703332)

[## 06. Post diagnostics – Calinski-Harabsasz and Duda-Hart measures of clustering quality 8](#_Toc526703333)

[## 07. Post-diagnostics - Shadow statistics 9](#_Toc526703334)

[## 08. Post-diagnostics - Rand index (ARI), Jaccard index, Fowlkes-Mallows index 10](#_Toc526703335)

[## 09. Post-diagnostics – How clustering changes when number of clusters increases? 11](#_Toc526703336)

[## 10. Statistics in clustered groups 12](#_Toc526703337)

[### TASK 1 16](#_Toc526703338)

[### TASK 2 17](#_Toc526703339)

## 01. Prediagnostics - Optimal number of clusters

**# checking how many clusters should be**

**# a) defines temporary clusters with kmeans, pam, clara, funny, hcut**

**# b) calculates the statistics for clusters number:**

**# - silhouette, - total within sum of square (wss)and - gap statistics**

**# c) plots the result**

**# shilhouette**

# **method of interpretation and validation of consistency within clusters of data**

# shilhouette statistics s=(bi–ai)/max(ai,bi)

# ai 🡪 average distance to all other objects in the cluster

# bi 🡪 minimum of average distance to other clusters (cluster by cluster)

# statistics is between -1 and 1

# negative s – undesirable, a>b other clusters are closer than “our” cluster

# positive s – desirable, good when a~0 (distance in our cluster), then s~1

# more on: <https://cs.fit.edu/~pkc/classes/ml-internet/silhouette.pdf>

**fviz\_nbclust(xxx, FUNcluster=pam) # factoextra::**

**fviz\_nbclust(xxx, clara, method="silhouette")+ theme\_classic() # factoextra::**

|  |  |
| --- | --- |
|  |  |

Observations / comments:

* The best division is for maximum silhouette
* One should be aware that the differences in silhouette between groupings may be small

## 02. Prediagnostics - How many clusters there should be?

**# using package NbClust**

**# more at:** [**https://www.jstatsoft.org/article/view/v061i06/v61i06.pdf**](https://www.jstatsoft.org/article/view/v061i06/v61i06.pdf)

**install.packages("NbClust")**

**library(NbClust)**

**library(help="NbClust") # only one function in a package**

**?NbClust # comprehensive description on distances, methods, tests**

**# function may work with distance matrix or dataset**

**c3<-NbClust(xxx, distance="euclidean", min.nc=2, max.nc=8, method="complete", index="ch")**

**c3**

**c3$All.index**

**c3$Best.nc**

**c3$Best.partition**

**# number of optimal clusters - there are few available criteria are:**

**# a) variance\_explained (default)**

**# b) WCSSE (within-cluster-sum-of-squared-error)**

**# c) dissimilarity,**

**# d) silhouette,**

**# e) distortion\_fK,**

**# f), g), h), AIC, BIC and Adjusted\_Rsquared.**

**xxx.s<-center\_scale(xxx) # from ClusterR:: to scale or center the data**

**opt<-Optimal\_Clusters\_KMeans(xxx.s, max\_clusters=10, plot\_clusters = TRUE)**

**opt<-Optimal\_Clusters\_KMeans(xxx.s, max\_clusters=10, plot\_clusters=TRUE, criterion="silhouette")**

**opt<-Optimal\_Clusters\_KMeans(xxx.s, max\_clusters=10, plot\_clusters=TRUE, criterion="AIC")**

|  |  |  |
| --- | --- | --- |
|  |  |  |

Observations / comments:

* Methods give different results – sometimes consistent, sometimes opposite
* Most often used is silhouette, however the others are also interesting
* Analysis of measures is to be performed in task 2

## 03. Prediagnostics - Automatic selection of number of clusters

**pamk.best<-pamk(xxx, krange=2:10,criterion="asw", usepam=TRUE, scaling=FALSE, alpha=0.001, diss=inherits(xxx, "dist"), critout=FALSE) # fpc::pamk()**

**class(pamk.best)**

**pamk.best**

$pamobject

Medoids:

ID XA06 XA31

[1,] 14 13.3 79.1

[2,] 306 9.1 92.1

Clustering vector:

[1] 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 2 2 1 1 1 1 1 1 2 2 2 1 1 2 1 1 1 2 2 1 1

[37] 2 2 2 1 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 1 1 2 1 2 1 1 2 1

[73] 2 2 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 1 1 1 2

[109] 1 1 1 1 1 1 1 1 2 1 2 2 1 2 2 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

[145] 1 1 1 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 2

[181] 2 2 1 2 1 1 2 2 1 1 1 1 1 1 1 1 1 2 1 1 2 1 2 1 2 1 1 1 1 2 1 2 1 1 1 2

[217] 1 1 2 1 2 1 2 2 2 1 2 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 1

[253] 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 1 2 1 2 2 2 2 1 2 2 2 2 1 1 2 1 1 2 2 1

[289] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 1 2 1 1 2 2 2 2 1 2 2 2 2 2 2

[325] 2 2 2 2 2 2 2 2 1 1 2 2 1 1 2 1 1 1 1 2 1 1 2 1 2 1 1 2 2 2 2 1 2 2 1 2

[361] 1 2 1 1 1 1 1 1 2 1 1 2 1 1 1 1 1 1 1 2

Objective function:

build swap

7.489160 7.117281

Available components:

[1] "medoids" "id.med" "clustering" "objective" "isolation"

[6] "clusinfo" "silinfo" "diss" "call" "data"

$nc

[1] 2

$crit

[1] 0.0000000 0.4395977 0.3582930 0.3839186 0.3512097 0.3554504 0.3472663

[8] 0.3294828 0.3344453 0.3300354

**# announcement printed out**

**cat("number of clusters estimated by optimum average silhouette width:", pamk.best$nc, "\n")**

**# plot of the result**

**plot(pam(xxx, pamk.best$nc))**

**# the same – selection of optimal number of clusters**

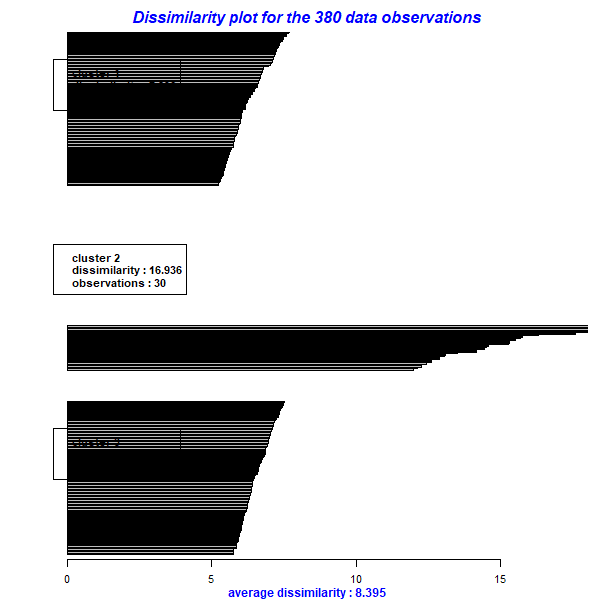
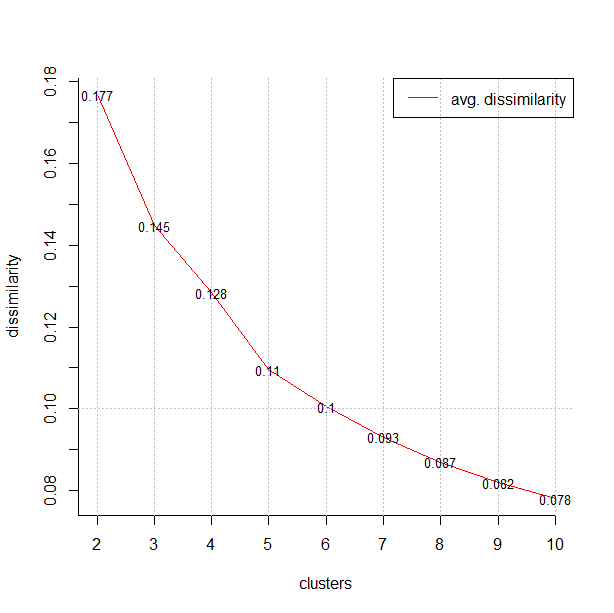
# ClusterR::Optimal\_Clusters\_Medoids()

# inside nice selection of distance metrics, possibility of sampling

# on the basis of dissimilarity figure one can decide about clusters

# no automatic decision

**opt\_md<-Optimal\_Clusters\_Medoids(xxx, 10, 'euclidean', plot\_clusters=TRUE)**



## 04. Prediagnostics - Assessing Clustering Tendency using Hopkins' statistic

# in simple words: “how well the data can be clustered”

# Hopkins statistics: total y / (total x + total y)

# total x 🡪 average distance to nearest neighbor between real data

# total y 🡪 average distance to nearest neighbor between real point

# and uniformly generated random point (with the same variance as real data)

# because of randomly generated data statistics may differ

Null hypothesis: the dataset is uniformly distributed (i.e., no meaningful clusters)

Alternative hypothesis: the dataset is not uniformly distributed (i.e., contains meaningful clusters)

Usually the interpretation is as follows:

|  |  |  |
| --- | --- | --- |
| h~0 | h~0.5 | h~1 |
| - accept the null hypothesis  - unlikely that there are statistically significant clusters  - no clusters are visible, uniformely distributed data | Random data | - reject the null hypothesis  - dataset is significantly a clusterable data  - some clusters are visible |

**However, implementations sometimes are opposite, as below, and implement the “1-h” (where h is the definition of H as above)**

# note that results from get\_clust\_tendency() may dramatically differ from hopkins()

# more on <http://www.sthda.com/english/articles/29-cluster-validation-essentials/95-assessing-clustering-tendency-essentials/>

hopkins(xxx, n=nrow(xxx)-1) # clustertend::

$H

[1] 0.1132107

**get\_clust\_tendency(xxx, 2, graph=TRUE, gradient=list(low="red", mid="white", high="blue"), seed = 123) # factoextra::**

|  |  |
| --- | --- |
| $hopkins\_stat  **[1] 0.06476448**  **Data are clusterable (clusters visible).** | **$plot** |

Observations / comments:

* In Hopkins statistic we interpret mainly the value of statistics, figure which appears automatically is a dissimilarity matrix, which is analysed below
* We do not need any significance test, as the comparison with the reference distribution is inherited in the function itself.

## 05. Prediagnostics - Visual assessment of ordered dissimilarity matrix (ODM)

**The figure as below appears also as an accompanying graphics to the Hopkins statistics. It can be obtained in at least two ways:**

* **with factoextra::get\_clust\_tendency()**
* **with factoextra::fviz\_dist()**

**Plotting the Ordered Dissimilarity Matrix**

# 1.Compute the dissimilarity matrix (DM) between the objects in the data set (Euclidean distance or other)

# 2. Reorder the DM to put similar objects close to one another and plot - one gets an ordered dissimilarity matrix (ODM)

# result blue (violet): high distance 🡪 high dissimilarity 🡪 low similarity

# result red (pink): low distance 🡪 low dissimilarity 🡪 high similarity

# interpretation: clustering tendency is present when blocks of colours are visible

# random data - when ordered data look like unordered data

**# two commands for distance (dissimilarity) are possible**

# stats:: dist() is basic function, output in class dist

# factoextra:: get\_dist() is tailored fuction, output also in class dist

# dist() allows for distances as "euclidean", "maximum",  "manhattan", "canberra",

"binary" or "minkowski"

# get\_dist() allows for distances as dist() and extra "pearson", "spearman" or "kendall"

**d<-dist(xxx)**

**d<-get\_dist(xxx, method = "euclidean")**

**fviz\_dist(d, show\_labels = FALSE)+ labs(title =** "our data"**) #factoextra::**

|  |  |
| --- | --- |
|  |  |

Observations / comments:

* One can see the blocks of colours at the figure, what confirms that data are clusterable and clustering is feasible.

## 06. Post diagnostics – Calinski-Harabsasz and Duda-Hart measures of clustering quality

# special package clusterCrit:: with many measures

# <https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf>

# overview of clustering stat is also available in fpc::cluster.stats()

**dxxx<-dist(xxx)**

**complete3 <- cutree(hclust(dxxx),3)**

**c.stat<-cluster.stats(dxxx,complete3)**

**c.stat**

**# quality of clustering for k-means - Calinski-Harabasz index (CH)**

# counter: BGSS/(K-1) 🡪 between-group sum of squares (for K clusters)

# nominator: WGSS / (N-K) 🡪 within-cluster sum of squares

# (sum of the within-cluster dispersions for all clusters) (for N observations)

# the higher statistics the better

# statistic used for comparing solutions for alternative number of clusters

**km1<-kmeans(xxx, 2) # stats::**

**round(calinhara(xxx,km1$cluster),digits=2) #fpc::calinhara()**

**km2<-kmeans(xxx, 3) # stats::**

**round(calinhara(xxx,km2$cluster),digits=2) #fpc::calinhara()**

# **Calinski-Harabasz is also available in fpc::cluster.stats()**

**dxxx<-dist(xxx)**

**complete3<-cutree(hclust(dxxx),3)**

**c.stat<-cluster.stats(dxxx,complete3)**

**c.stat$ch**

**# Duda-Hart test for whether a data set should be split into two clusters**

**# for kmeans class**

**# H0: homogeneity of cluster (data within cluster as similar)**

**# H1: heterogeneity of cluster (one can easily split the cluster)**

**# statistics dh: ratio of within-cluster sum of squares for two clusters and overall sum of squares.**

**# verification: cluster1=FALSE (H0 of homogeneity rejected, accept H1)**

**# verification: when dh statistics is lower than “compare” (critical value), accept H1**

**km1 <- kmeans(xxx,2)**

**dudahart2(xxx,km1$cluster) #fpc::**

$p.value

[1] 9.545304e-07

$dh

[1] 0.4932701

$compare

[1] 0.5594387

$cluster1

[1] FALSE

$alpha

[1] 0.001

$z

[1] 3.090232

Observations / comments:

* Following the Duda-Hart test we should split the clusters (as cluster1=FALSE )
* Following Calinski-Harabasz test, we should prefer k=3 over k=2

## 07. Post-diagnostics - Shadow statistics

# Shadow statistics very close to silhouette

# more in Leisch F (2009) Neighborhood Graphs, Stripes and Shadow Plots for Cluster Visualization# <https://pdfs.semanticscholar.org/4d41/34253b34c3425b6cf29595d3f84c96a76892.pdf>

“The main difference between silhouette values and shadow values is that we replace average dissimilarities to points in a cluster by dissimilarities to point averages (=centroids).” <https://rdrr.io/cran/flexclust/man/shadow.html>

# k-means clustering

**d1<-cclust(xxx, 4, dist="euclidean") #flexclust:: for k-means**

**plot(xxx, col=predict(d1))**

**points(d1@centers, pch="x", cex=2, col=1)**

**plot(d1)**

|  |  |
| --- | --- |
|  |  |

**# definition of shadow:** twice the distance to the closest centroid divided by the sum of distances to closest and second-closest centroid.

# interpretation: shadow~0 🡪 points are close to their centroids

# shadow~1 🡪 points are equidistant to the two centroids

# good cluster: many points with small shadow values

**shadow(d1)**

**plot(shadow(d1))**

|  |  |
| --- | --- |
| **1 2 3 4**  **0.5936298 0.6362001 0.6280041 0.5344348** |  |

Observations / comments:

* In some special cases shadow statistics performs better than shillouette.

## 08. Post-diagnostics - Rand index (ARI), Jaccard index, Fowlkes-Mallows index

**# Measures to be applied when few partitions are considered**

**# In general they check how many points changed the cluster in new partition**

**# They are measuring the so called “agreement of partitions”**

**# Most popular: Adjusted Rand Index (ARI), Jaccard index, Fowlkes-Mallows (FM) index**

**# All measures are calculated with one command flexlust::randIndex()**

**# Rand index=A/(A+D)**

# A 🡪 number of all pairs of data points which are either put into the same cluster by both partitions or put into different clusters by both partitions (always the same or always opposite cluster)

# D 🡪 number of all pairs of data points that are put into one cluster in one partition, but into different clusters by the other partition (“migrating” points)

# when D=0 (always the same partitioning), ARI=1, otherwise ARI falls down

**# Jaccard Index**

# almost the same as Rand Index, but in A there are only “always the same cluster”

**# Fowlkes-Mallows**

# almost the same as Rand Index, but A is number of pairs in the same cluster divided by the geometric mean of the sums of the number of pairs in each cluster of the two partitions.

**d1<-cclust(xxx, 4, dist="euclidean")**

**d2<-cclust(xxx, 4, dist="manhattan")**

**randIndex(d1, d2)**

**comPart(d1, d2)**

ARI RI J FM

0.4105315 0.7550896 0.4124858 0.5840844

Observations / comments:

* Partitionings differed in distance metrics. It shows that cluster compositions differ under both scenarios

## 09. Post-diagnostics – How clustering changes when number of clusters increases?

**# One can ran the full track of clustering changes for k-means procedure.**

**# This is to analyse the stability of solution step-by step.**

**# One can check the clustering for different number of clusters given by krange**

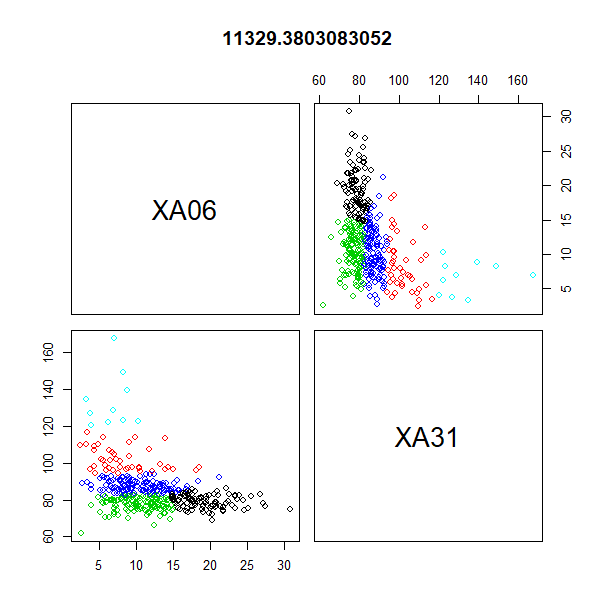
**# Function shows the iterations of clusterings**

**# Function is available with fpc::kmeansrus()**

**# criterion might be one of "asw" (average silhouette) or "ch" (Calinski-Harabasz)**

**c9<-kmeansruns(xxx, krange=2:5,criterion="ch", iter.max=2, runs=2, scaledata=FALSE, alpha=0.001, critout=FALSE, plot=TRUE)**

**c9**



K-means clustering with 5 clusters of sizes 115, 10, 42, 94, 119

Cluster means:

XA06 XA31

1 10.394783 77.33217

2 6.690000 133.37000

3 8.366667 101.86190

4 19.388298 79.04043

5 10.301681 87.42269

Clustering vector:

[1] 4 4 4 4 1 1 5 4 5 4 1 3 5 1 5 3 3 5 4 1 4 4 1 3 3 3 5 5 5 4 4 4 5 5 1 4 5 5 2 1 4 2 5 3 3 5 4 4

[49] 5 1 1 4 4 5 4 1 1 1 2 1 1 1 3 1 1 3 1 5 4 1 3 4 5 5 5 5 1 1 3 1 1 1 1 1 1 1 1 4 1 5 1 5 5 3 5 1

[97] 1 2 5 1 5 1 1 1 1 5 1 5 1 1 1 1 1 1 1 1 5 1 3 5 1 5 5 5 4 4 4 5 4 1 5 4 4 4 1 5 1 4 1 4 1 1 5 1

[145] 5 5 1 3 5 5 3 5 5 1 1 5 1 5 3 3 3 5 2 5 5 1 1 4 5 1 1 5 1 1 5 1 1 5 1 5 5 5 1 5 5 1 5 5 1 1 1 4

[193] 5 4 4 1 5 5 1 1 3 1 3 1 5 1 1 1 1 3 1 3 1 1 1 5 1 4 5 5 5 5 5 5 5 1 5 4 1 5 5 4 4 4 2 4 1 4 4 1

[241] 5 4 4 4 4 1 3 3 4 1 4 4 4 1 4 4 4 1 1 4 4 4 4 4 5 1 5 3 1 5 4 3 5 5 5 1 3 3 5 5 5 5 5 4 4 5 3 4

[289] 4 4 4 4 5 1 4 3 4 4 4 4 4 5 3 1 4 5 5 1 5 1 5 1 1 5 5 5 5 1 5 3 2 5 3 3 3 5 3 3 2 2 5 5 5 5 5 5

[337] 1 5 5 4 4 1 1 3 4 4 3 4 5 4 1 5 5 3 5 1 5 5 1 2 4 5 4 4 4 4 4 5 5 4 4 3 4 4 4 4 1 4 4 5

Within cluster sum of squares by cluster:

[1] 2437.648 2095.050 2307.012 2103.904 2402.288

(between\_SS / total\_SS = 82.3 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss"

[7] "size" "iter" "ifault" "crit" "bestk"

Warning: did \*not\* converge in specified number of iterations

Observations / comments:

* One can see which cluster was split into two in every step

## 10. Statistics in clustered groups

**# Clustering methods are often used in segmentation. They are to split the sample into heterogeneous groups. In consequence, as a post-action we want to see the descriptive statistics in the defined groups. By assumption, they should be different.**

**# stripes for k-means**

**# to plot with stripes the distance of data points to cluster centroids**

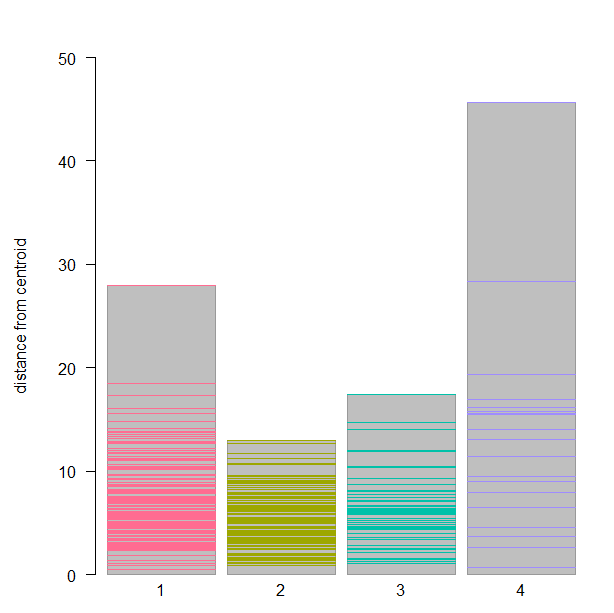
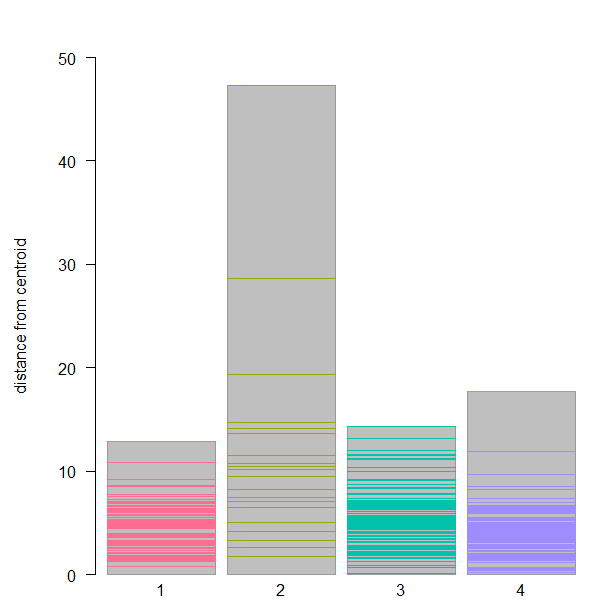
**# stripes() works only with class kcca (from cclust())**

**d1<-cclust(xxx, 4, dist="euclidean") #flexclust::**

**stripes(d1) #flexclust::**

**d2<-cclust(xxx, 4, dist="manhattan") #flexclust::**

**stripes(d2) #flexclust::**



Observations / comments:

* On stripes chart we see the distance of every single observation from the centroid of cluster. The higher bin the more distant locations of points within given cluster (undesirable)

**# boxplots for variables in groups**

**# the code below links to kmeans() command and operates on clustering object**

**# for k-means**

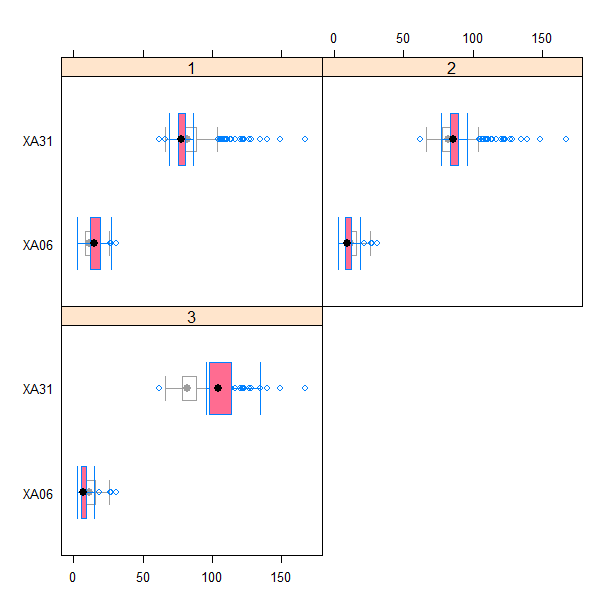
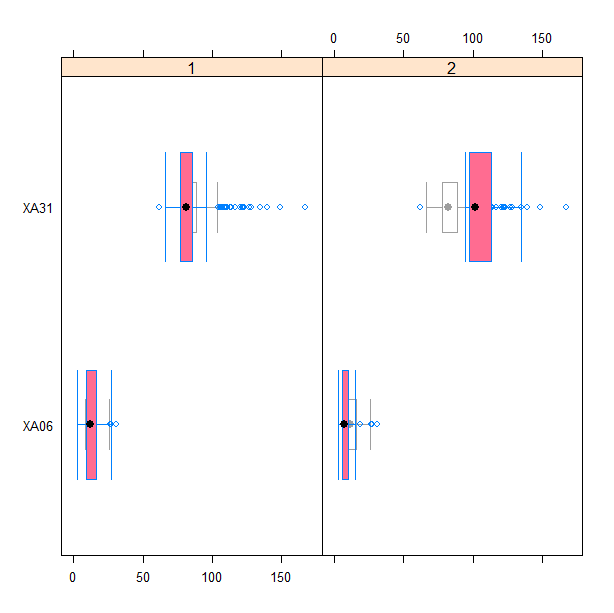
**km1 <- kmeans(xxx, 2) # stats::**

**groupBWplot(xxx, km1$cluster, alpha=0.05) #flexclust::**

**# for PAM**

**c1<-pam(xxx,3)#cluster::**

**groupBWplot(xxx, as.factor(c1$clustering), alpha=0.05) #flexclust::**



**# One can use also the standard commands for boxplots and other statistical analyses.**

**# We need only to add to the subset the clustering vector obtained from k-means procedure.**

**# Than we use this as a grouping variable.**

**# data subset with information on cluster**

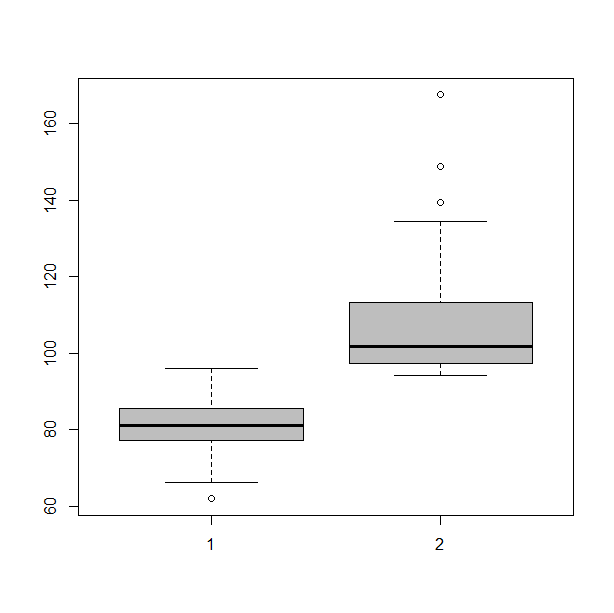
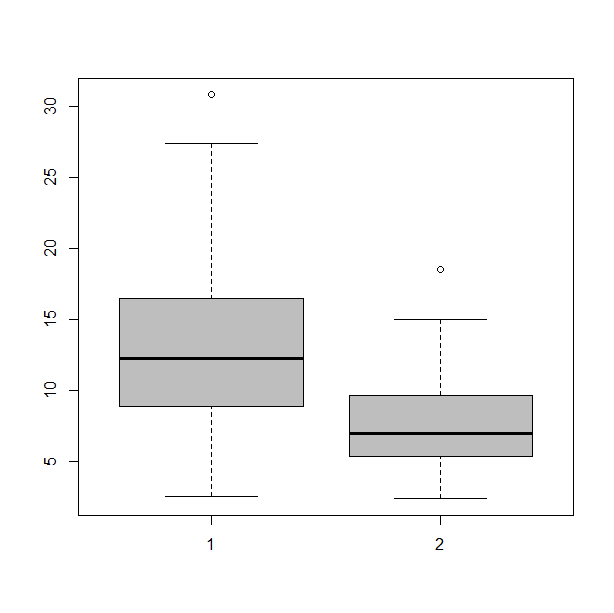
**xxc<-cbind(xxx, km1$cluster)**

**colnames(xxc)[3]<-c("group")**

**# boxplot in a typical way – each graph is for separate variable**

**boxplot(xxc[,1]~xxc[,3], vertical=TRUE, col="grey")**

**boxplot(xxc[,2]~xxc[,3], vertical=TRUE, col="grey")**

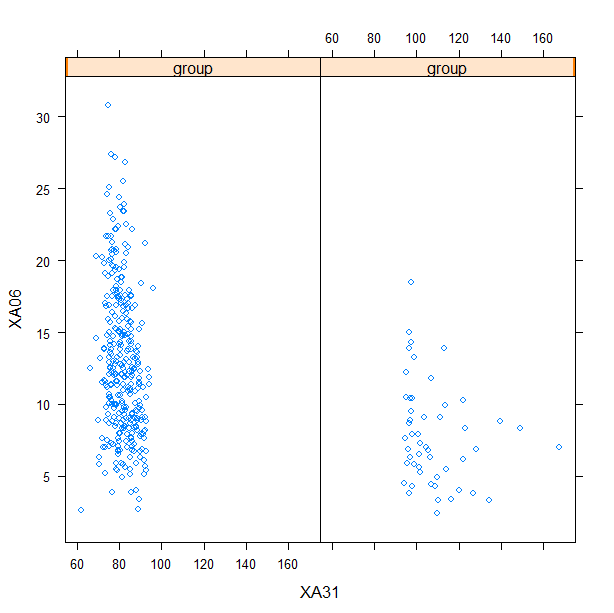


**# dotplots in groups**

**install.packages("lattice")**

**library(lattice)**

**xyplot(XA06 ~ XA31 | group, data = xxc)**



Observations / comments:

* Histograms and dotplots prove that values of observations on both groups are different.

**Beyond graphical summaries, one can easily summarize the data with descriptive statistics. There are few methods for this.**

**# typical statistics**

**xxc<-cbind(xxx, km1$cluster)**

**#install.packages("psych")**

**library(psych)**

**describeBy(xxc[,1:2], xxc[,3])**

Descriptive statistics by group

group: 1

vars n mean sd median trimmed mad min max range skew kurtosis se

XA06 1 329 12.95 5.16 12.3 12.64 5.19 2.6 30.8 28.2 0.56 -0.09 0.28

XA31 2 329 81.53 5.70 81.2 81.45 6.38 61.9 96.0 34.1 0.05 -0.31 0.31

------------------------------------------------------------------------

group: 2

vars n mean sd median trimmed mad min max range skew kurtosis se

XA06 1 51 7.85 3.51 7.0 7.52 3.11 2.4 18.5 16.1 0.81 0.25 0.49

XA31 2 51 108.15 15.22 101.9 105.40 8.15 94.3 167.6 73.3 1.78 3.41 2.13

Observations / comments:

* With describeBy() we get in a simple way all statistics in both groups for both variables.

**#install.packages("doBy")**

**library(doBy)  
summaryBy(xxc[,1]+xxc[,2]~xxc[,3], data=xxc, FUN=function(x) {c(m=mean(x), s=sd(x))} )**

xxc[, 1].m xxc[, 1].s xxc[, 2].m xxc[, 2].s

1 12.26868 5.267673 85.10053 11.88576

Observations / comments:

* With summaryBy we define in formula as the LHS the variables to be analysed and as the RHS the grouping variable, while in FUN we specify statistics to be calculated

**# with aggregate function**

**s1<-aggregate(xxc[,1], by=list(xxc[,3]), mean)**

**s2<-aggregate(xxc[,1], by=list(xxc[,3]), sd)**

**s1**

Group.1 x

1 1 12.954103

2 2 7.847059

**s2**

Group.1 x

1 1 5.164211

2 2 3.514334

Observations / comments:

* The aggregate() command always outputs the column Group.1 and x and whole object is *data.frame*.
* In this command one can run in one line only a single statistics for a single variable (multi grouping criteria are allowed)

**# summary with data.table**

**#install.packages("data.table")**

**library(data.table)**

**colnames(xxc)<-c("unemployment","salaries","group")**

**dt <- data.table(xxc)**

**dt[,list(mean=mean(unemployment),sd=sd(unemployment)),by=group]**

group mean sd

1: 1 12.954103 5.164211

2: 2 7.847059 3.514334

**# statistics in a loop**

**# for each variable by groups, for many statistics**

**stats<-matrix(0, nrow=4, ncol=4)**

**colnames(stats)<-c("mean","sd","min","max")**

**rownames(stats)<-rep(c("cluster1","cluster2"),times=2)**

**rownames(stats)<-paste(rownames(stats), rep(c("var1","var2"), each=2))**

**funs<-c("mean","sd","min","max")**

**for(i in 1:2){ # iterating by variables**

**for(j in 1:4){ # iterating by functions**

**temp<-aggregate(xxc[,i], by=list(xxc[,3]), funs[j])**

**stats[(2\*i-1):(2\*i),j]<-temp$x}}**

mean sd min max

cluster1 var1 12.954103 5.164211 2.6 30.8

cluster2 var1 7.847059 3.514334 2.4 18.5

cluster1 var2 81.526748 5.699659 61.9 96.0

cluster2 var2 108.154902 15.216889 94.3 167.6

**# comparison of distances**

**dd1<- get\_dist(xxx, method="euclidean")**

**dd1.mat<-as.matrix(dd1)**

**dd1.mat[1:10, 1:10]**

1 2 3 4 5 6 7 8 9 10

1 0.0000000 4.701064 3.962323 8.854377 10.332473 11.374093 18.388312 0.8062258 15.147607 11.751596

2 4.7010637 0.000000 1.886796 4.197618 6.074537 8.104937 13.780058 5.5036352 10.575916 7.984986

3 3.9623226 1.886796 0.000000 5.700877 6.389053 7.576939 14.561937 4.7381431 11.301770 9.848858

4 8.8543774 4.197618 5.700877 0.000000 4.275512 7.632169 10.117806 9.6462428 7.128113 5.108816

5 10.3324731 6.074537 6.389053 4.275512 0.000000 3.522783 8.338465 11.1198022 5.092151 9.078546

6 11.3740934 8.104937 7.576939 7.632169 3.522783 0.000000 9.704638 12.0739389 6.905071 12.578156

7 18.3883115 13.780058 14.561937 10.117806 8.338465 9.704638 0.000000 19.1927069 3.264966 11.859595

8 0.8062258 5.503635 4.738143 9.646243 11.119802 12.073939 19.192707 0.0000000 15.950549 12.419742

9 15.1476071 10.575916 11.301770 7.128113 5.092151 6.905071 3.264966 15.9505486 0.000000 9.824968

10 11.7515956 7.984986 9.848858 5.108816 9.078546 12.578156 11.859595 12.4197423 9.824968 0.000000

**dd2<-get\_dist(xxx, method="manhattan")**

**dd2.mat<-as.matrix(dd2)**

**dd3<-get\_dist(xxx, method="minkowski")**

**dd3.mat<-as.matrix(dd3)**

**dd4<-get\_dist(xxx, method="canberra")**

**dd4.mat<-as.matrix(dd4)**

**plot(density(dd1.mat))**

**lines(density(dd2.mat), lwd=2)**

**lines(density(dd3.mat), lwd=3)**

**lines(density(dd4.mat), lty=2, lwd=1)**

### TASK 1

Repeat the loop task for three variables (subset yyy) in tree clusters:

1. For Euclidean PAM
2. For Manhattan K-means

Add boxplots below.

|  |  |
| --- | --- |
| Statistics for Euclidean PAM  (three variables, three clusters) | Statistics for Manhattan K-means  (three variables, three clusters) |
|  |  |
| Boxplots for Euclidean PAM  (three variables, three clusters) | Boxplots for Manhattan K-means  (three variables, three clusters) |
|  |  |

### TASK 2

**For a few subsets (xxx, yyy, etc.) generate a set of measures and check correlations between them. Are the measures consistent?**

Instruction

methods<-c(**"**variance\_explained**"**, **"**WCSSE**"**, **"**dissimilarity**"**, **"**silhouette**"**, **"**distortion\_fK**"**, **"**AIC**"**, **"**BIC**", "**Adjusted\_Rsquared**")**

mmm<-matrix(0, nrow=10, ncol=8)

colnames(mmm)<-methods

xxx.s<-center\_scale(xxx) # from ClusterR:: to scale or center the data

opt1<-Optimal\_Clusters\_KMeans(xxx.s, max\_clusters=10, plot\_clusters=TRUE, criterion=methods[1])

mmm[,1]<-opt1

cor(mmm) # correlation matrix

pairs(mmm) # correlation scatterplot

library(corrplot)

corrplot(cor(mmm), method = "circle") #colorful graphics

|  |  |
| --- | --- |
| correlation matrix |  |
| correlation scatterplot |  |
| colorful graphics |  |