Workshop: Data science with R (ZEW)

Session #6: Unsupervised learning

Obryan Poyser

2019-04-04

Outline

1. Unsupervised learning
    1. Types
    2. Challenges
2. Clustering
    1. (Dis)similiarity measures
        1. Quantitative and non-quantitative
    2. Algorithms
3. Dimensionality reduction

End-to-end machine learning project

1- Look at the big picture.
2- Get the data.
3- Discover and visualize the data to gain insights.
4- Prepare the data for Machine Learning algorithms.

5- Select a model and train it.
6- Fine-tune your model.
7- Present your solution.
8- Launch, monitor, and maintain your system

Unsupervised learning

Definition

1. Subsumes all kinds of machine learning where there is **no known output**
2. "Learning without a teacher"

Types

1. Unsupervised transformations (UT): algorithms that create new representation of high dimensional data, which output is potentially easier for other algorithms as well as human to understand the underlying structure.
    1. Dimensionality reduction is the most common application of UT. It summarises the data by creating a smaller feature space that contains as much as possible the variance of the original representation.
    2. Clustering algorithms: partition of data into different groups according to a similarity rule.

Challenges

1. Evaluate if the algorithm is learning something salient
2. With unlabeled data is hard to tell if the we have done right.
    1. In psychological studies is common to construct factors from a set of features.
3. Is considered as "pre-processing" the data which is going to be used for supervised learning models.
4. Unsupervised learning algorithms are **highly sensitive to scales**.

Unsupervised learning: Clustering (data segmentation)

Definition

1. The process of grouping similar objects together.
2. Grouping/segmenting a collection of objects into subsets or "clusters", such that those within each cluster are more closely related to one another than objects assigned to different clusters.

Inputs

1. Similarity-based clustering $(N \times N)$
    1. Instrument are typically dissimilarity or distance matrices.
2. Feature-based clustering $(N \times X_p)$

Outputs

1. Flat clustering: partition into disjoint sets
    1. It is necessary to provide ex-ante the number of sets
2. Hierarchical clustering: nested tree sets
    1. It is not required to specify the number of sets

Unsupervised learning: Clustering (data segmentation)

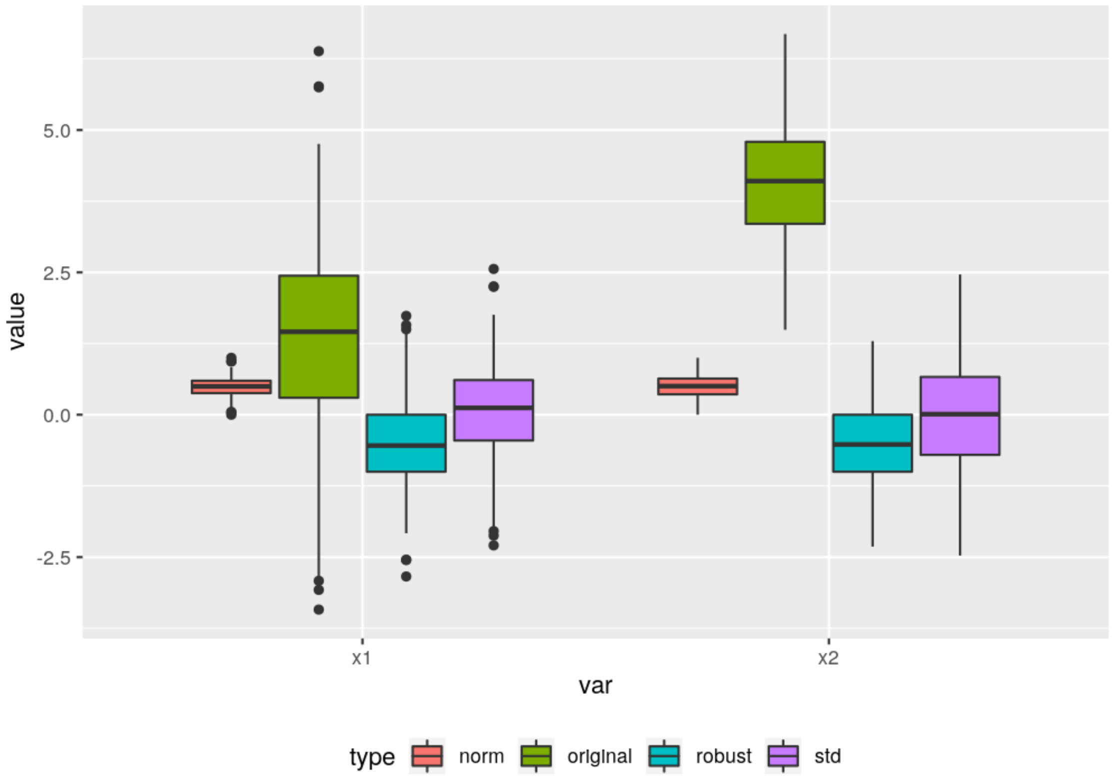
Transformations

1. As stated, clustering algorithms are sensitive to scales. It is a common practice to transforma the features.
    1. Normalize $x_i \in (0,1)$: $\frac{x_i - min(x_i)}{max(x_i) - min(x_i)}$
    2. Standardize $x_i \sim (0,1)$: $\frac{x_i - E(x_i)}{\sigma_{x_i}}$
    3. Robust scaler: $\frac{x_i - Q_1(x_i)}{Q_3(x_i) - Q_1(x_i)}$

```
x1=1*rep(1, 100)+rnorm(100, sd = 2)
x2=4*rep(1, 100)+rnorm(100)
```

```
stdF <- function(x){
    (x-mean(x, na.rm = T))/sd(x, na.rm = T)
}

normF <- function(x){
    (x-min(x, na.rm = T))/(max(x, na.rm = T)-min(x, na.rm = T)
}
```



Unsupervised learning: Clustering (data segmentation)

Measuring (dis)similarity

1. A clustering method attempts to group the objects based on the definition of similarity supplied to it.
2. Data is defined in terms of proximity between a pair of objects. Proximity can be measure either by affinity (similarities) or lack of it (dissimilarities).
    1. D of size $N \times N$, where N is the number of objects.
    2. Matrix D is used as an input for the algorithm
    3. Algorithms assumes that the matrix D is simmetric
3. A dissimilarity matrix D is a matrix where $d_{i,i} = 0$ and $d_{i,j} \geq 0$, that is, it measures the distance between elements i and j.

Unsupervised learning: Clustering (data segmentation)

Measuring (dis)similarity based on attributes

1. Most often we have measurements x_{ij} for $i = 1, 2, \ldots, N$, on variables $j = 1, 2, \ldots, p$ (also called attributes)
2. Step 1: construct pairwise dissimilarities between the observations and used as input.
3. Step 2: We define a dissimilarity $d_j(x_{ij}, x_{i'j})$ between values of the jth attribute.
4. Step 3: Define the measure according to the type of attribute (feature)

Quantitative

Measure	**Formula**
Absolute	$d(x_i, x_i') = abs(x_i - x_i')$
Squared	$d(x_i, x_i') = (x_i - x_i')^2$
Correlation	$\frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2(y_i-\overline{y})^2}}$

Unsupervised learning: Clustering (data segmentation)

Measuring (dis)similarity based on attributes

Categorical

1. Ordinal: Ordered set of elements eg likert scales. Error measures for ordinal features are generally defined by replacing their M original values following:

$$\frac{i - 1/2}{M}, \ i = 1, \ldots, M$$

1. Categorical: With unordered categorical (nomimal) we must assess the degree-of-difference between pairs of values by creating $M \times M$ matrix of distinct elements. Distance is given by $L_{rr'} = 1$ if the elements match.

One can create a correlation of categorical variables with **tetrachoric** (binary) and **polychoric** (ordinal) correlations. Nice description

In the appendix I have included more distance measures.

Unsupervised learning: Clustering (data segmentation)

Algorithms

1. Combinatorial: work directly on the observed data with no direct reference to an underlying probability model
    1. Most popular
    2. Hard to assess the quality of grouping output
2. Mixture modeling: data is an i.i.d sample from some population described by a probability density function.
3. Mode seekers ("bump hunters") take a nonparametric perspective, attempting to directly estimate distinct modes of the probability density function.

Unsupervised learning: Clustering (data segmentation)

Combinatorial

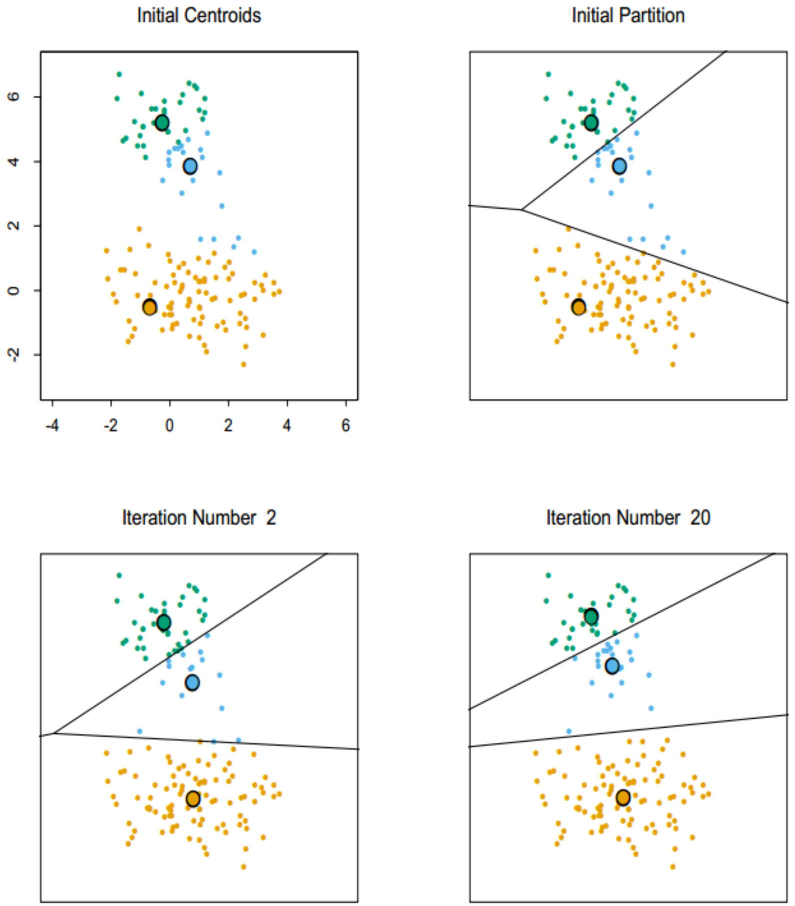
1. Each observation is uniquely labeled by an integer $i \in \{1, \cdots, N\}$.
2. Prespecified number of clusters $K < N$ is postulated, and each one is labeled by an integer $k \in \{1, \ldots, K\}$
3. Each observation is assigned to one and only one cluster.
4. Heuristic procedure
5. Popular algorithms
    1. K-means clustering
    2. Hierarchical clustering

Unsupervised learning: Clustering (data segmentation)

K-means

K-means Clustering.

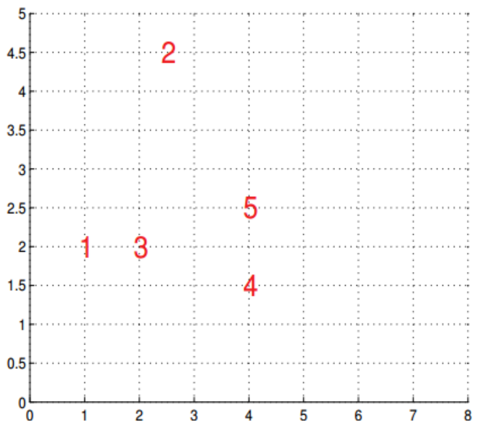
1. For a given cluster assignment C, the total cluster variance is minimized with respect to $\{m_1, \ldots, m_K\}$ yielding the means of the currently assigned clusters
2. Given a current set of means $\{m_1, \ldots, m_K\}$, is minimized by assigning each observation to the closest (current) cluster mean. That is,
$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} ||x_i - m_k||^2.$$
3. Steps 1 and 2 are iterated until the assignments do not change.

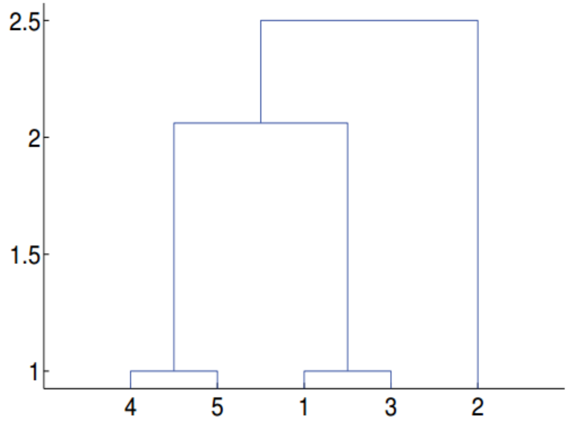


Unsupervised learning: Clustering (data segmentation)

Hierarchical clustering (HC)

1. K-means demand to establish the number of clusters beforehand. In contrast, hierarchical clustering methods do no require such specifications.
2. With HC the user have to specify a measure of dissimilarity between (disjoint) groups of observations, based on the pairwise dissimilarities among the observations in the two groups.
3. Produces hierarchical reprentation of the data

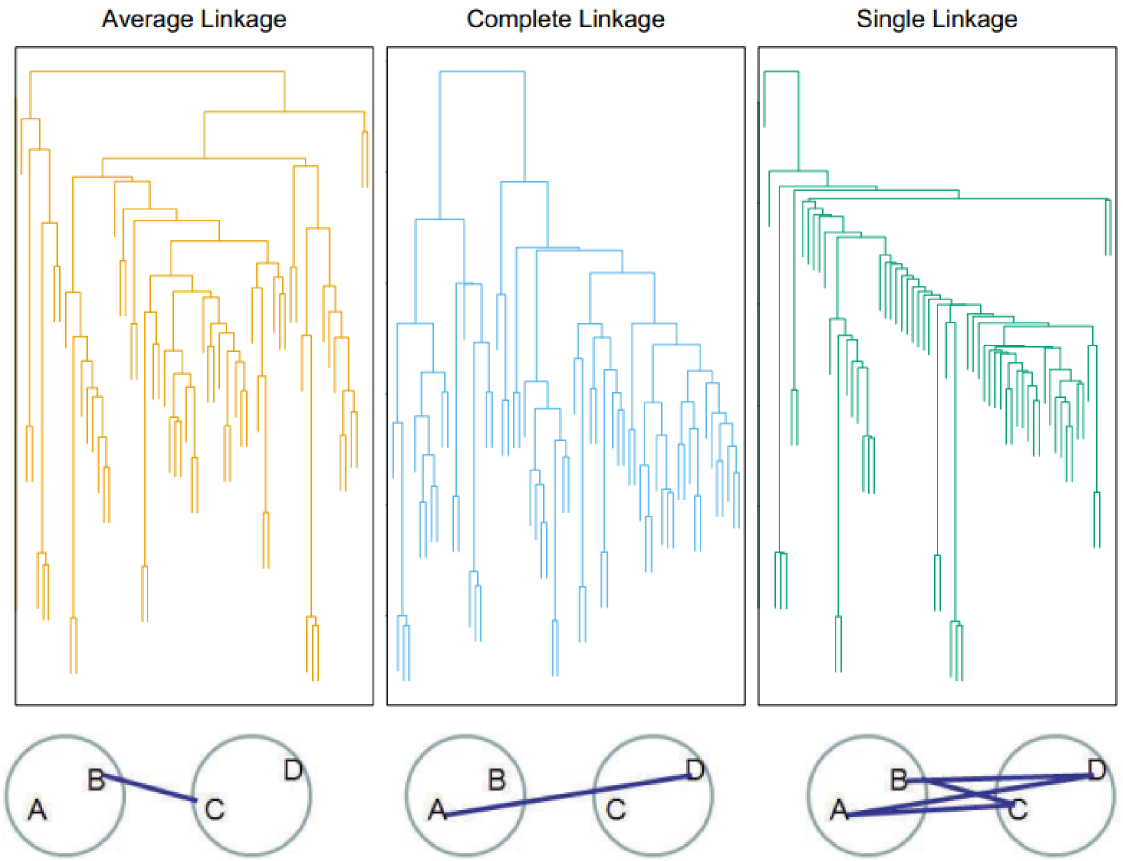




Unsupervised learning: Clustering (data segmentation)

Hierarchical clustering (HC)

1. Agglomerative (bottom-up): it starts by merging two set of clusters, then repeat the process in the next level.
    1. Step 1: Every observation represents a cluster
    2. Step 2: Cluster merge into a single cluster in the next level
    3. The dissimilarity at each level could be:
        1. Single linkage (SL) or nearest-neighbor
        2. Complete linkage (CL)
        3. Average linkage (AL)
        4. Centroid linkage (CeL)
        5. Ward's minimum variance method (ward)



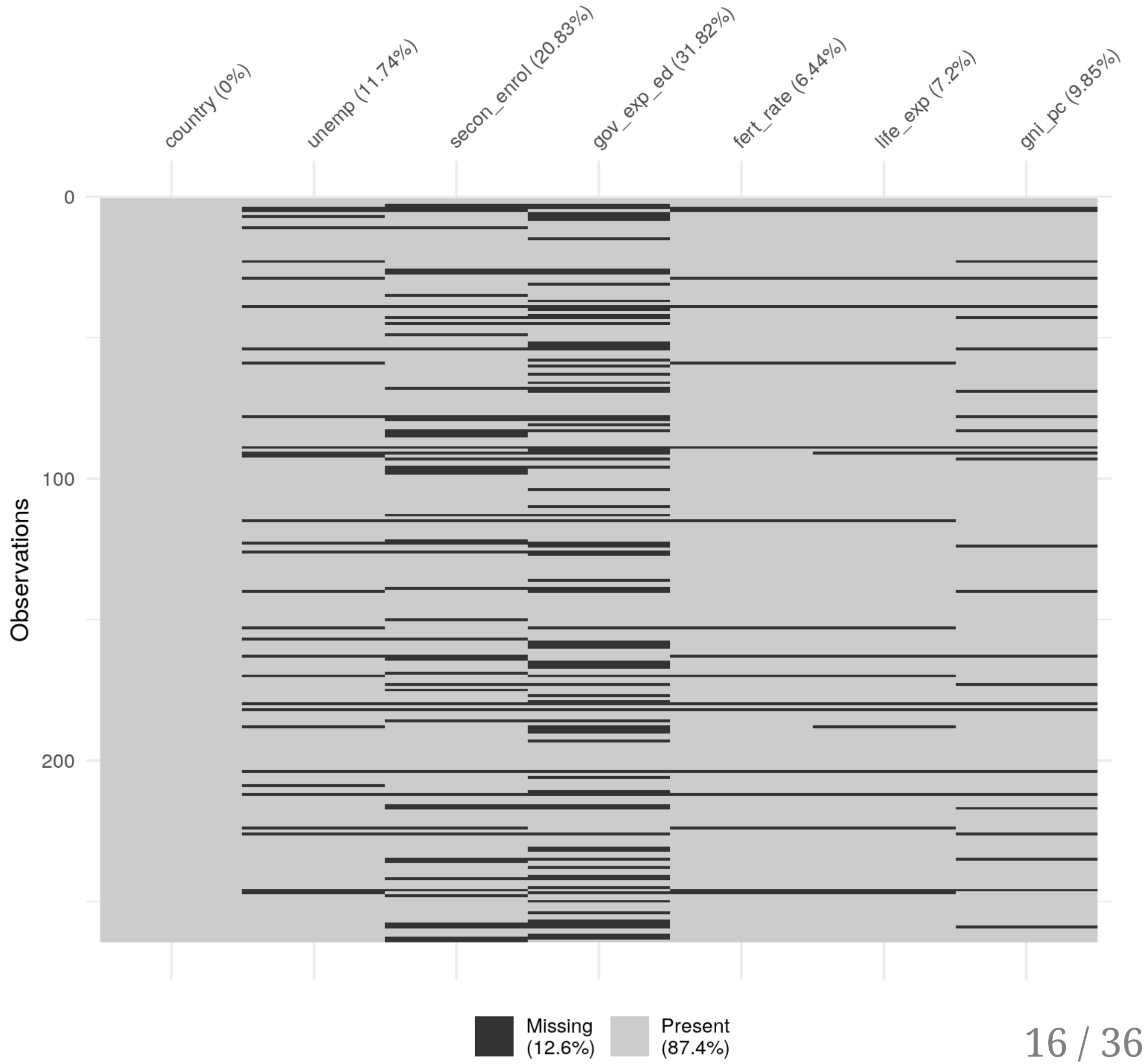
K-means clustering: segmentating the countries according to WDI

```
wdi <- WDI::WDI(country = "all", indicator = c("unemp"="SL.UEM.TOTL.ZS"
                                               , "secon_enrol"="SE.SEC.ENRR"
                                               , "gov_exp_ed"="SE.XPD.TOTL.GD.ZS"
                                               , "fert_rate"="SP.DYN.TFRT.IN"
                                               , "life_exp"="SP.DYN.LE00.MA.IN"
                                               , "gni_pc"="NY.GNP.PCAP.CD")
                 , start = 2014
                 , end = 2018) %>%
    as_tibble()
```

```
## # A tibble: 1,320 x 9
##    iso2c country  year unemp secon_enrol gov_exp_ed fert_rate life_exp
##    <chr> <chr>   <int> <dbl>       <dbl>      <dbl>     <dbl>    <dbl>
##  1 1A    Arab W…  2014 10.3         70.2      NA         3.41     69.1
##  2 1A    Arab W…  2015 10.3         70.6      NA         3.37     69.3
##  3 1A    Arab W…  2016 10.0         70.9      NA         3.33     69.5
##  4 1A    Arab W…  2017  9.93        71.1      NA        NA        NA
##  5 1A    Arab W…  2018  9.81        NA        NA        NA        NA
##  6 1W    World    2014  5.44        76.3       4.72      2.46     69.6
##  7 1W    World    2015  5.45        76.5       4.81      2.45     69.8
##  8 1W    World    2016  5.53        76.8      NA         2.44     70.0
##  9 1W    World    2017  5.49        76.6      NA        NA        NA
## 10 1W    World    2018  5.38        NA        NA        NA        NA
## # … with 1,310 more rows, and 1 more variable: gni_pc <dbl>
```

K-means clustering: segmentating the countries according to WDI

```
wdi %>%
  group_by(country) %>%
  summarise_at(.vars = vars(unemp:gni_pc)
               , .funs = ~mean(., na.rm = T)
               ) %>%
  naniar::vis_miss()
```



K-means clustering: segmentating the countries according to WDI

```r
(wdi_case <- wdi %>%
  group_by(country) %>%
  summarise_at(.vars = vars(unemp:gni_pc)
               , .funs = ~mean(., na.rm = T)
               ) %>%
  filter(complete.cases(.)) %>% # filter and keep complete cases only
  mutate(country=str_to_lower(country)) %>%
  filter(!grepl(country, pattern = "asia|euro|america|demographic|countries|income|ida|oecd|small|africa|world")
         )
 )
```

```
## # A tibble: 116 x 7
##    country     unemp secon_enrol gov_exp_ed fert_rate life_exp gni_pc
##    <chr>       <dbl>       <dbl>      <dbl>     <dbl>    <dbl>  <dbl>
##  1 afghanistan  8.81        53.6       3.76      4.81     62.0    595
##  2 albania     15.8         95.8       3.19      1.71     76.2   4392.
##  3 argentina    7.96       107.        5.57      2.31     72.6 12488.
##  4 armenia     18.1         86.0       2.60      1.62     71.1   3985
##  5 australia    5.78       156.        5.25      1.81     80.4 57708.
##  6 austria      5.65       100.        5.45      1.48     78.8 47335
##  7 azerbaijan   4.99        91.7       2.83      1.95     68.9   5772.
##  8 bahrain      1.21       102.        2.60      2.06     75.9 21538.
##  9 bangladesh   4.39        66.6       1.54      2.13     70.6   1265
## 10 barbados    10.5        108.        5.33      1.80     73.3 15365
## # … with 106 more rows
```

Necessary packages for this session

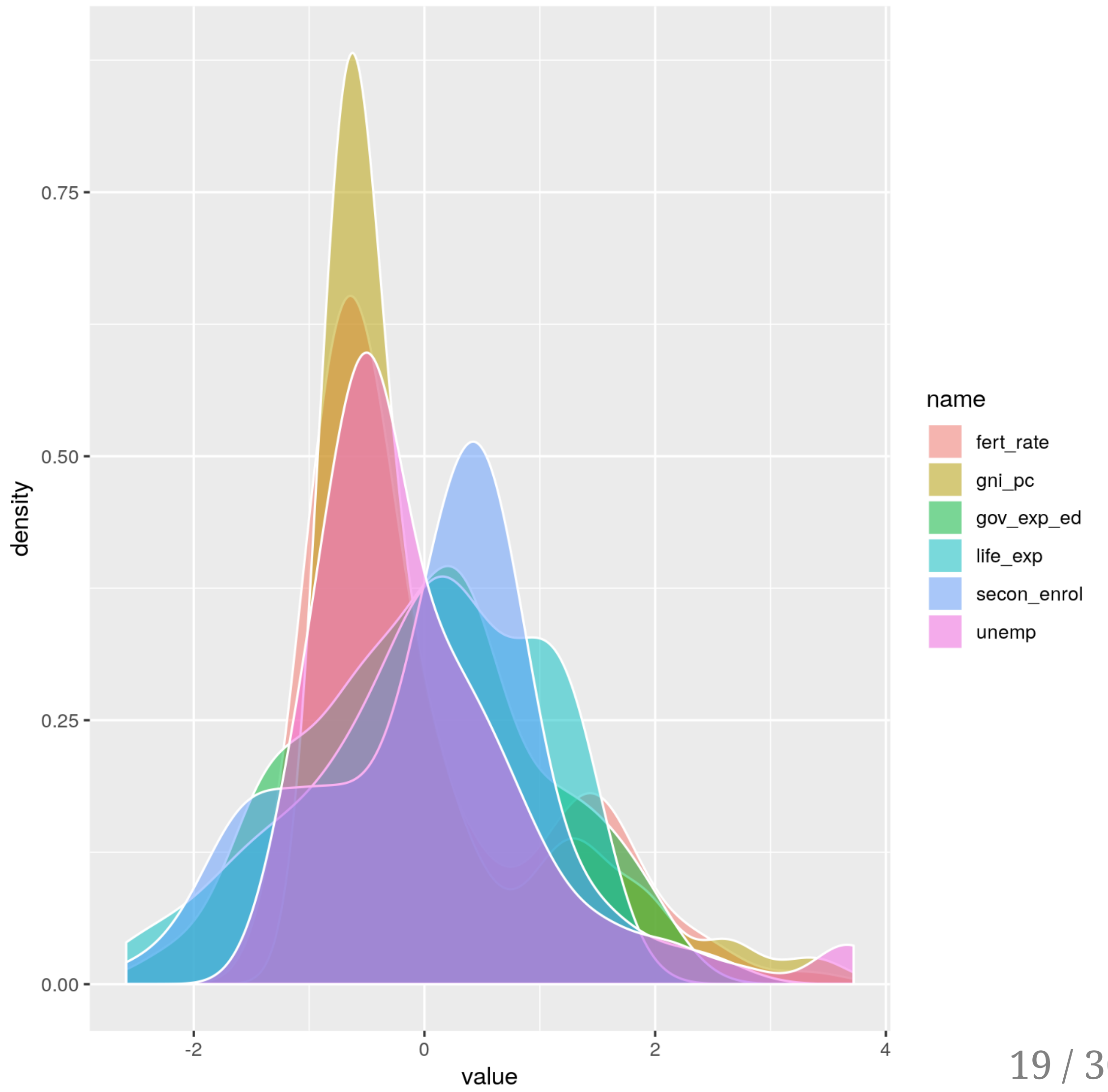
```
library(tidyverse)
library(knitr)
library(magrittr)
library(factoextra)
library(cluster)
library(gridExtra)
```

K-means clustering: segmentating the countries according to WDI

Transformations

```r
(wdi_case %<>%
  mutate_at(.vars = vars(-country), .funs = ~stdF(.))) %>%
  pivot_longer(cols = -country) %>%
  ggplot(aes(fill=name, value))+
  geom_density(alpha=.5, col="white")
```

Tip: The build-in function `scale(x)` does the same as the user-defined function `stdF(x)`

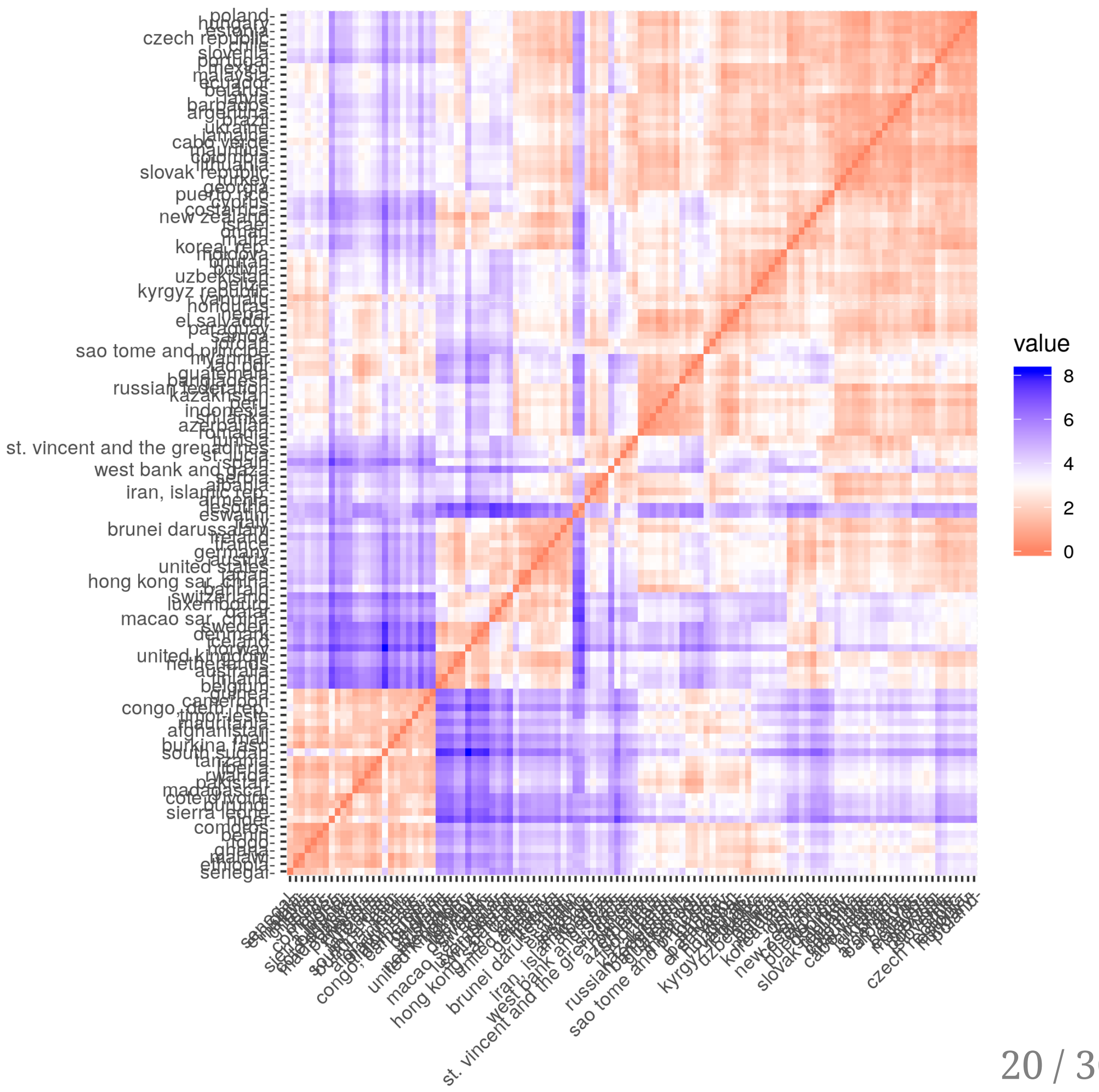


K-means clustering: segmentating the countries according to WDI

Distance

```r
(wdi_case_df <- wdi_case %>%
    tibble::column_to_rownames("country")) %>%
  get_dist(method = "euclidean") %>%
  fviz_dist()
```

Tip: The package `factoextra` is a good tool to visualize clusters in R.



K-means clustering: segmentating the countries according to WDI

```
set.seed(123)
wdi1 <- wdi_case %>%
    sample_n(50) %>%
    tibble::column_to_rownames("country")  # kmeans does not w

(kmeans <- c(2:10) %>%
    enframe(name = "name", value = "centers") %>%
    dplyr::select(-name) %>%
    mutate(k_means=map(centers, ~kmeans(wdi1, centers = ., nst
                          )
            )
  )
```

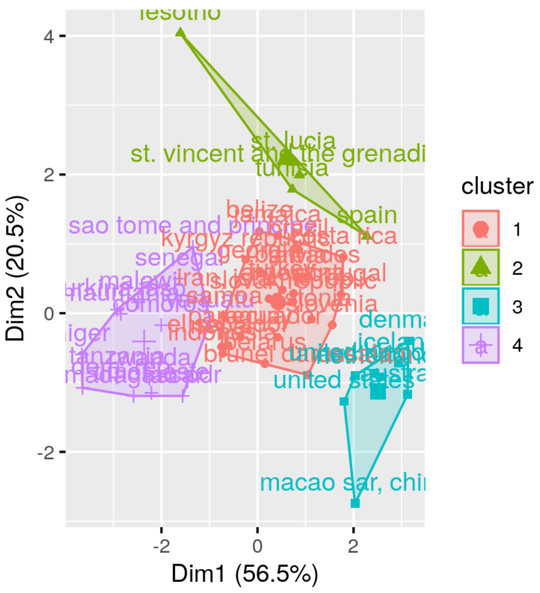
```
## # A tibble: 9 x 2
##   centers k_means
##     <int> <list>
## 1       2 <S3: kmeans>
## 2       3 <S3: kmeans>
## 3       4 <S3: kmeans>
## 4       5 <S3: kmeans>
## 5       6 <S3: kmeans>
## 6       7 <S3: kmeans>
## 7       8 <S3: kmeans>
## 8       9 <S3: kmeans>
## 9      10 <S3: kmeans>
```

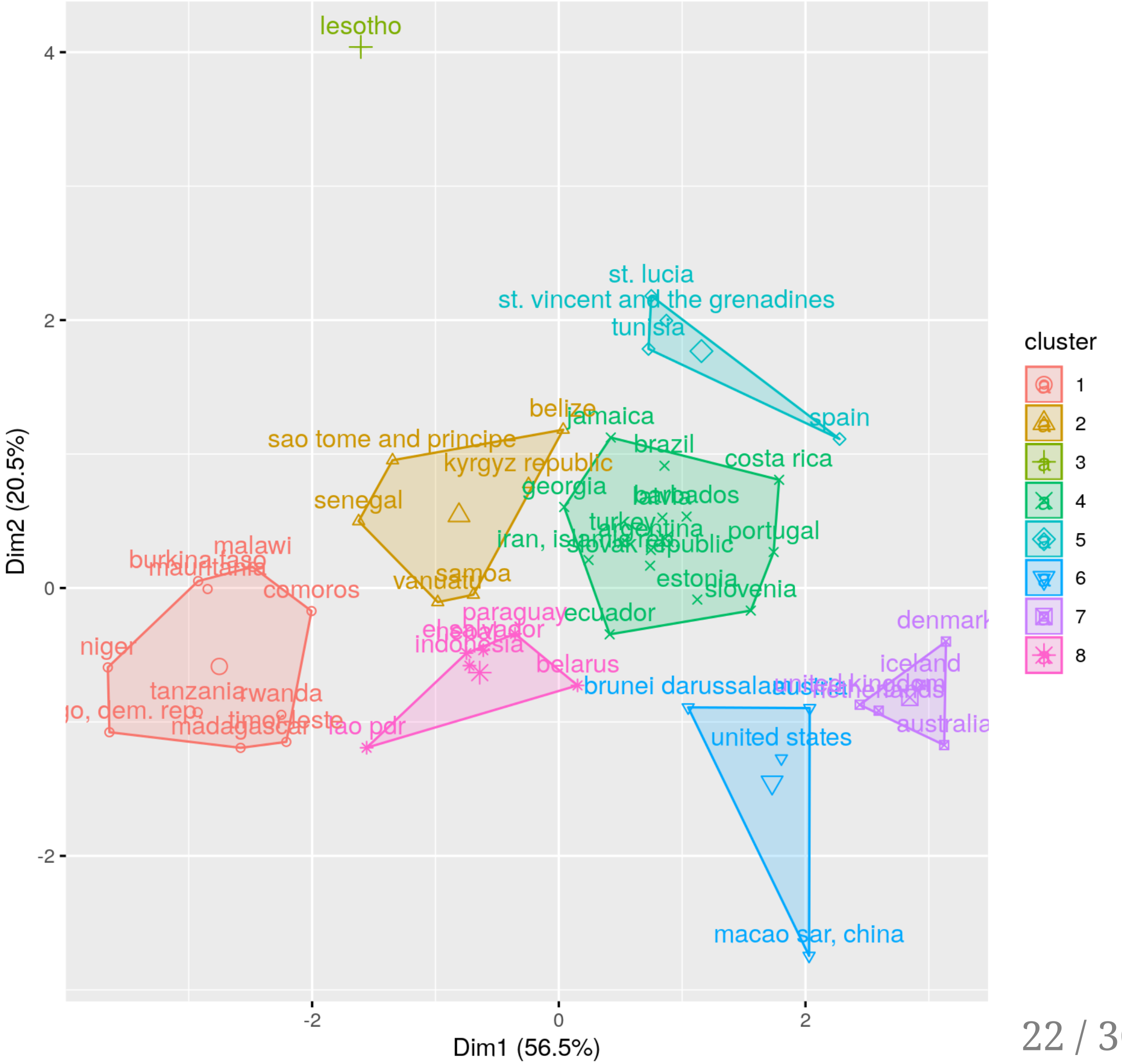
```
kmeans$k_means[[1]] %>% summary()
```

```
##              Length Class  Mode
## cluster      50     -none- numeric
## centers      12     -none- numeric
## totss         1     -none- numeric
## withinss      2     -none- numeric
## tot.withinss  1     -none- numeric
## betweenss     1     -none- numeric
## size          2     -none- numeric
## iter          1     -none- numeric
## ifault        1     -none- numeric
```

K-means clustering: segmentating the countries according to WDI

```
kmeans %<>%
  mutate(k_graph=map(k_means
                     , ~fviz_cluster(object = .
                                     , data = wdi1)+
                     labs(title="")
                     )
  )
```





K-means clustering: segmentating the countries according to WDI

```
kmeans %<>%
  mutate(tidy=map(k_means, ~broom::tidy(.)
                 )
         , augment=map(k_means
                      , ~broom::augment(.
                                       , data=wdi1) %>%
                      dplyr::select(.rownames, .cluster))
         , glance=map(k_means
                     , ~broom::glance(.
                                     , data=wdi1))
         )
```

1. `tidy`: Summarizes the model's statistical findings (varies across models)
2. `augment`: Add predictions, residuals, and cluster assignments.
3. `glance`: construct a concise one-row summary of the model.

```
kmeans$tidy[[1]]
```

```
## # A tibble: 2 x 9
##       x1     x2     x3     x4     x5     x6  size withinss clus
##    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl> <int>    <dbl> <fct
## 1  0.359  0.715  0.410 -0.587  0.548  0.346    30     93.2 1
## 2 -0.251 -1.06  -0.421  1.09  -0.842 -0.694    20     74.0 2
```

```
kmeans$augment[[1]] %>% head(3)
```

```
## # A tibble: 3 x 2
##   .rownames .cluster
##   <chr>     <fct>
## 1 estonia   1
## 2 rwanda    2
## 3 iceland   1
```

```
kmeans$glance[[1]]
```

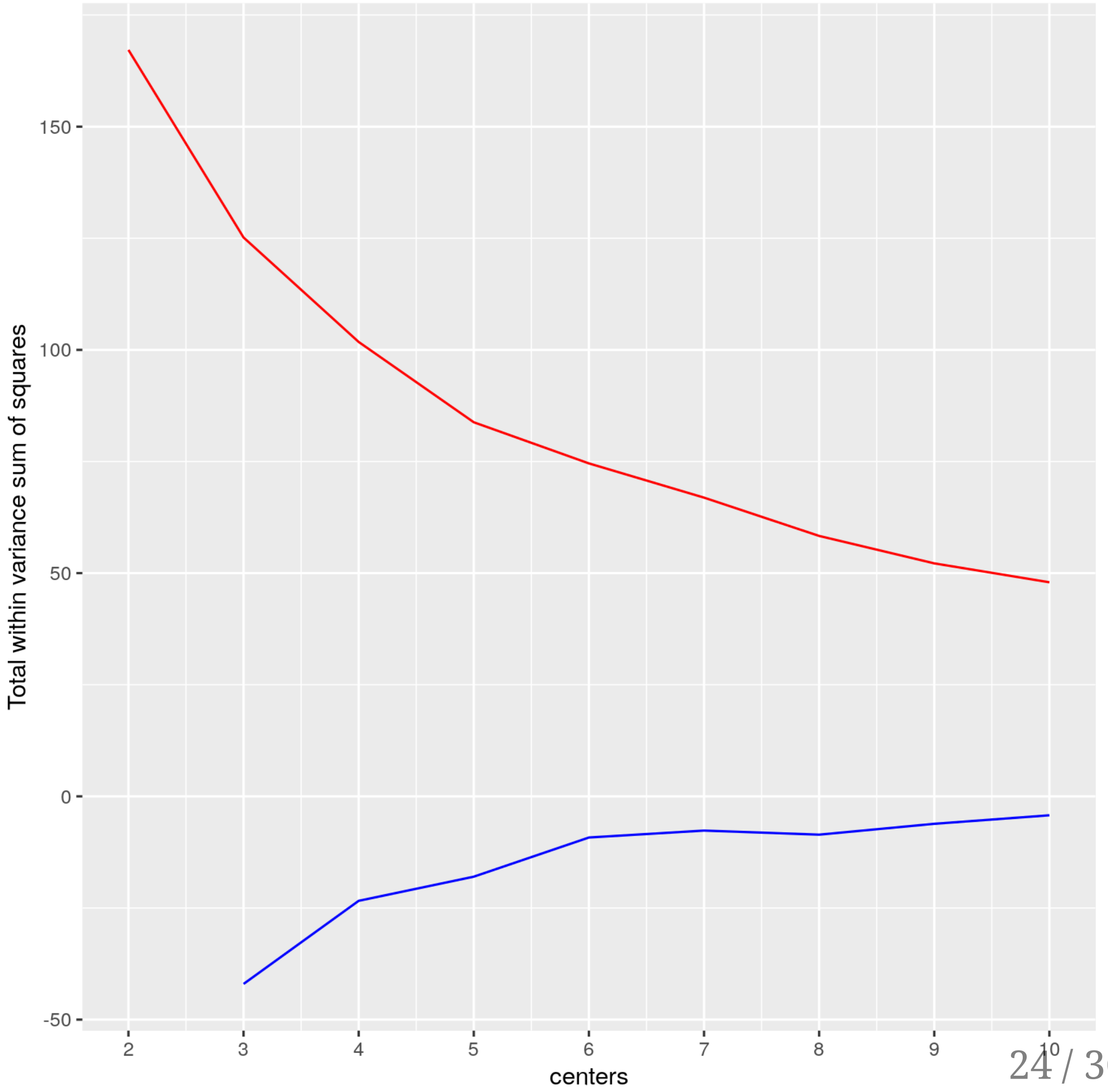
```
## # A tibble: 1 x 4
##   totss tot.withinss betweenss  iter
##   <dbl>        <dbl>     <dbl> <int>
## 1  288.         167.      121.     1
```

K-means clustering: segmentating the countries according to WDI

Optimal number of clusters

1. The rule of thumb (also named Elbow Method) for choosing the K amount of clusters is to visualize gaps in the total within variance inside the cluster.
   1. The total within cluster sum of square (wss) measures the closeness (minimum as possible)
   2. Alternatives method: Gap statistic, Silhouette.

```
kmeans %>%
  unnest(glance) %>%
  mutate(ratio=betweenss/tot.withinss
         , diff=tsibble::difference(tot.withinss)) %>%
  ggplot()+
  geom_line(aes(centers, diff), col="blue")+
  geom_line(aes(centers, tot.withinss), col="red")+
  scale_x_continuous(breaks = 1:10)+
  labs(y="Total within variance sum of squares")
```



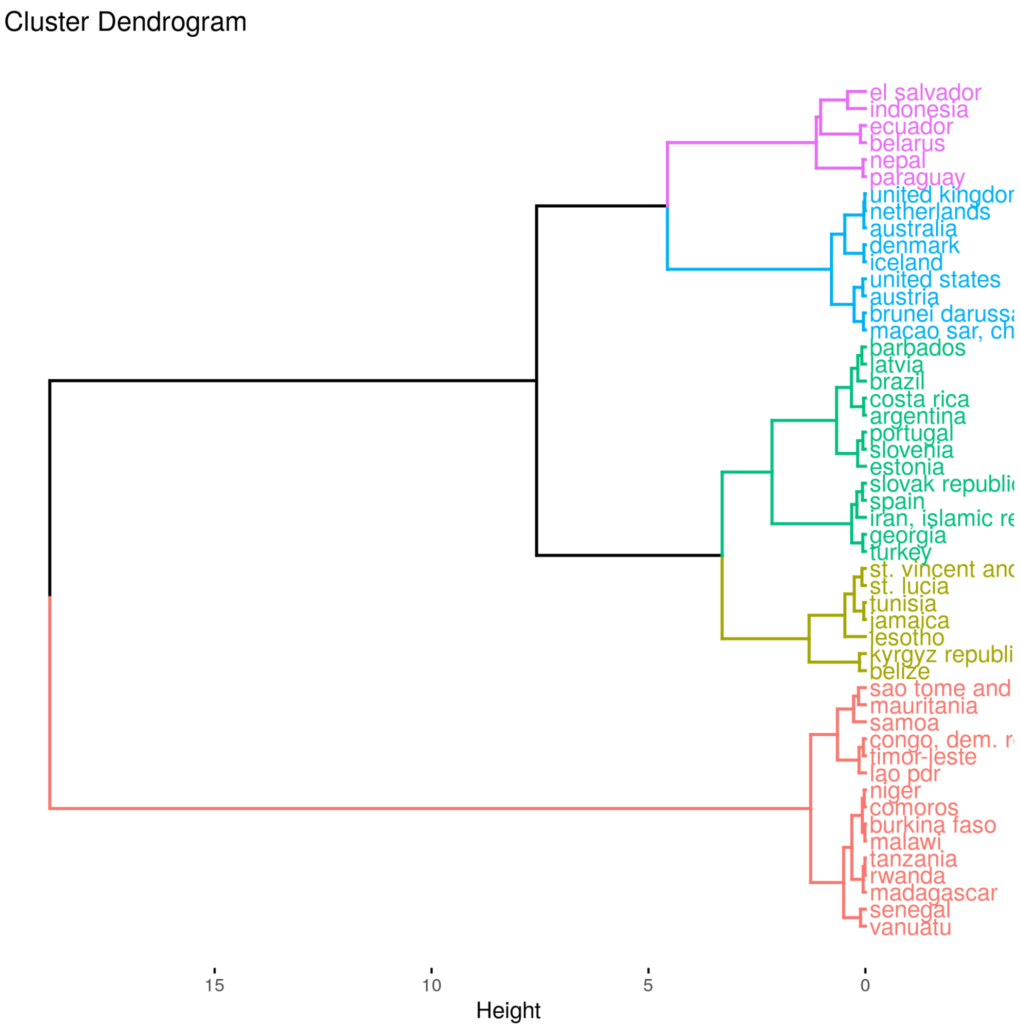
Hierarchical clustering: segmentating the countries according to WDI

```r
hclust <- tidyr::expand_grid(aggl=c("ward.D", "ward.D2", "single"
                                    , "complete", "average", "centroid")
                    , k=2:5) %>%
  mutate(h_clust=map2(.x = aggl, .y = k, .f = ~eclust(x = wdi1, FUNcluster = "hclust"
                                                    , hc_method = .x
                                                    , k=.y
                                                    , hc_metric = "pearson")
                       )
         )
```

```
## # A tibble: 24 x 3
##    aggl         k h_clust
##    <chr>    <int> <list>
##  1 ward.D       2 <S3: hclust>
##  2 ward.D       3 <S3: hclust>
##  3 ward.D       4 <S3: hclust>
##  4 ward.D       5 <S3: hclust>
##  5 ward.D2      2 <S3: hclust>
##  6 ward.D2      3 <S3: hclust>
##  7 ward.D2      4 <S3: hclust>
##  8 ward.D2      5 <S3: hclust>
##  9 single       2 <S3: hclust>
## 10 single       3 <S3: hclust>
## # … with 14 more rows
```

Hierarchical clustering: segmentating the countries according to WDI

```
fviz_dend(hclust$h_clust[[4]], horiz = T)
```



Hierarchical clustering: segmentating the countries according to WDI

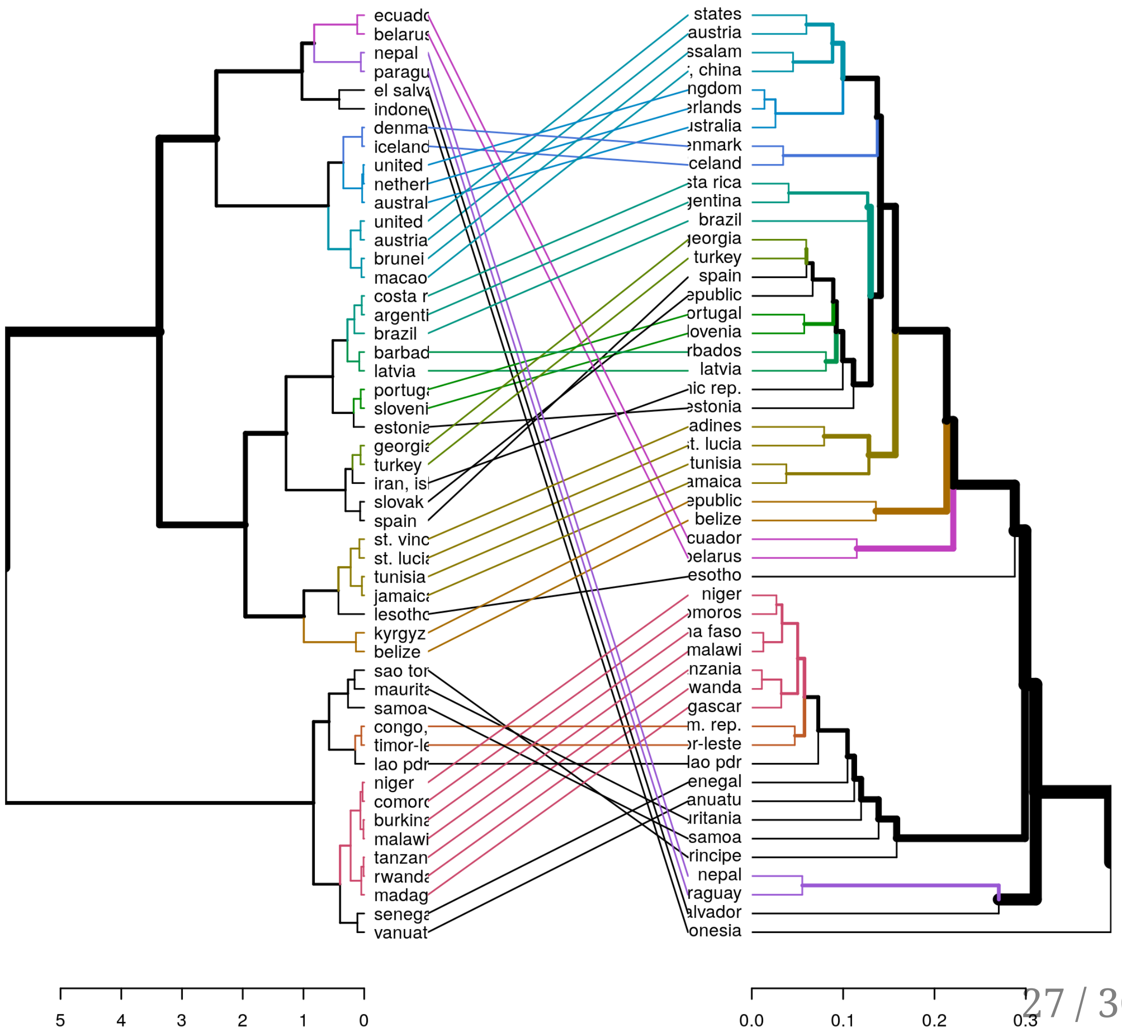
How sensitive are the cluster affinity to the agglomeration measure?

```
dendextend::tanglegram(hclust$h_clust[[8]], hclust$h_clust[[12
                       , lwd=1
                       , highlight_distinct_edges = FALSE
                       , common_subtrees_color_branches = TRUE
```

Entanglement measures the correspondency between two trees. It goes from 1 (full entanglement) to 0 (no entanglement).

```
dlist <- dendextend::dendlist(as.dendrogram(hclust$h_clust[[8]
                              , as.dendrogram(hclust$h_clust[[
dendextend::entanglement(dlist)
```

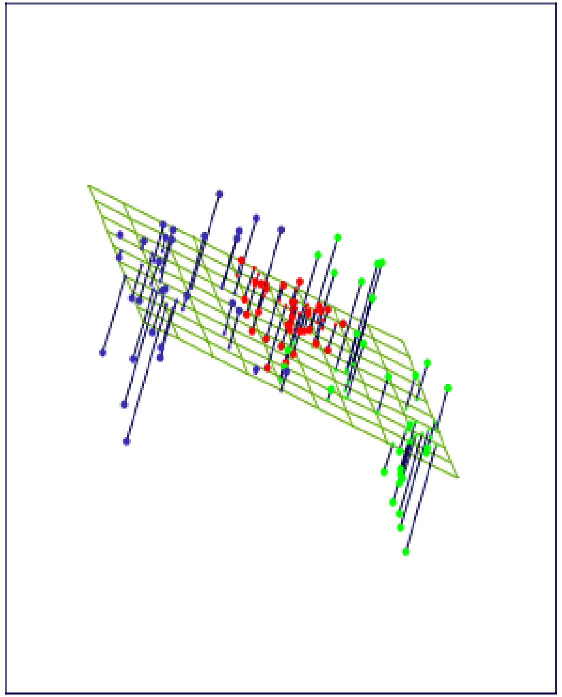
```
## [1] 0.3239344
```

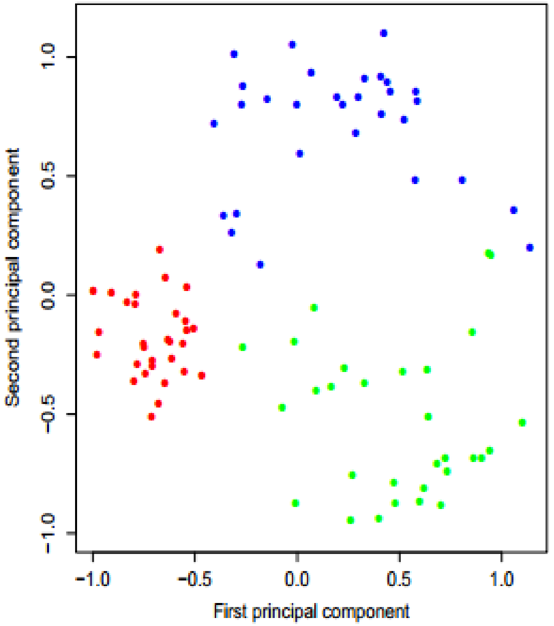


Unsupervised learning: Principal Components

1. Principal components are a sequence of projections of the data, mutually uncorrelated and ordered in variance.
2. First it identifies the hyperplane that lies closest to the data, and then it projects the data onto it.
3. Given a set of data $\mathbb{R}^p$, a PC provides a linear approximation to the data $(x_1, \ldots, x_N)$, of all ranks $q \leq p$.
    1. $f(\lambda) = \mu + V_q \lambda$

where: μ is a location vector in $\mathbb{R}^p$, V_q a $p \times q$ matrix with q orthogonal unit vector as columns, and λ is a q vector of parameters.





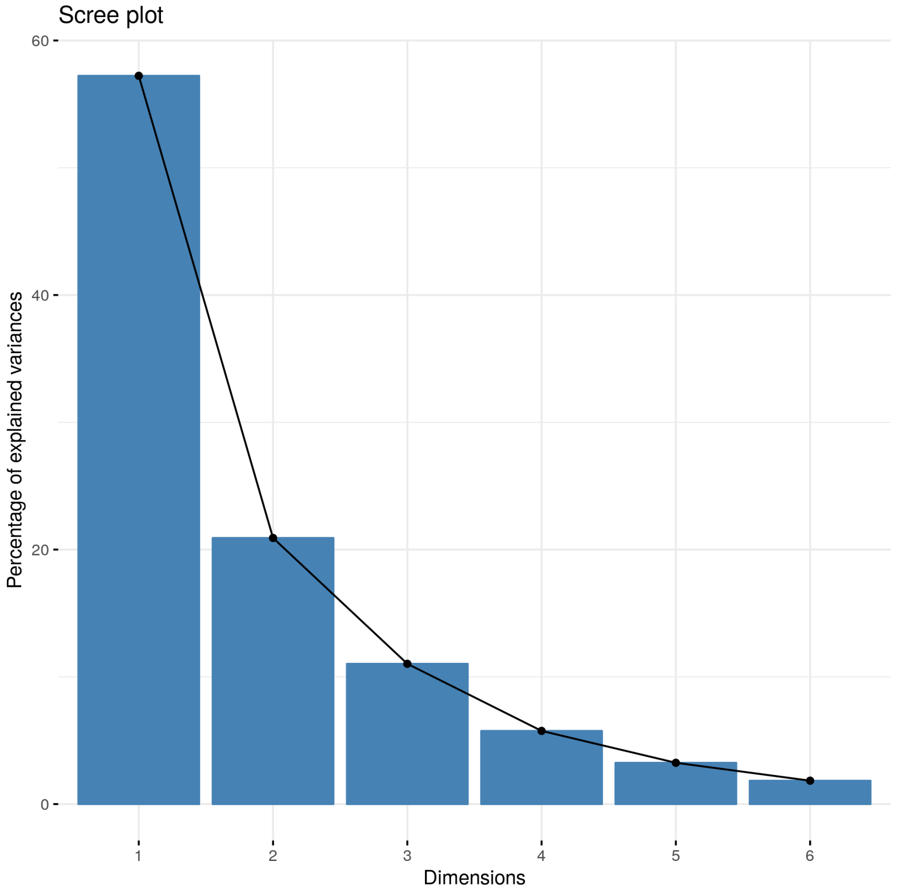
Source\: Friedman, J., Hastie, T., & Tibshirani, R. (2001)

Unsupervised learning: Principal Components

1. Components are orthogonal to each other, therefore, there is no common variance among the total set.
2. Typically, one applies Singular Value Decomposition (SVD) to decompose the original data in terms of eigenvalue and eigenvectors.

```
(pca <- prcomp(wdi1)) %>%
  summary()
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5    PC6
## Standard deviation     1.8336 1.1084 0.8048 0.58128 0.43670 0.3287
## Proportion of Variance 0.5723 0.2091 0.1103 0.05751 0.03246 0.0184
## Cumulative Proportion  0.5723 0.7814 0.8916 0.94914 0.98160 1.0000
```

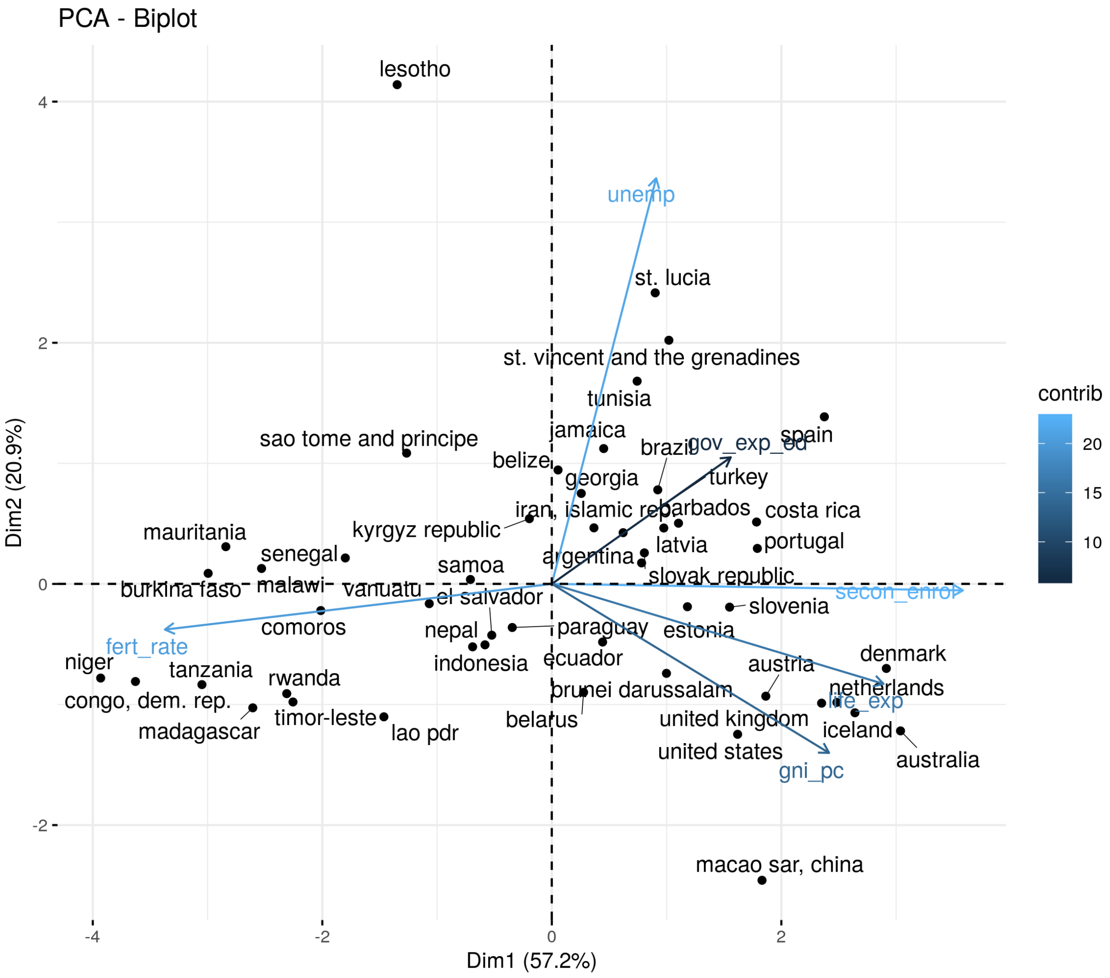


How many factors include?

Unsupervised learning: Principal Components

```r
fviz_pca(pca, col.var = "contrib", repel=T)
```

```r
pheatmap::pheatmap(cor(wdi1))
```



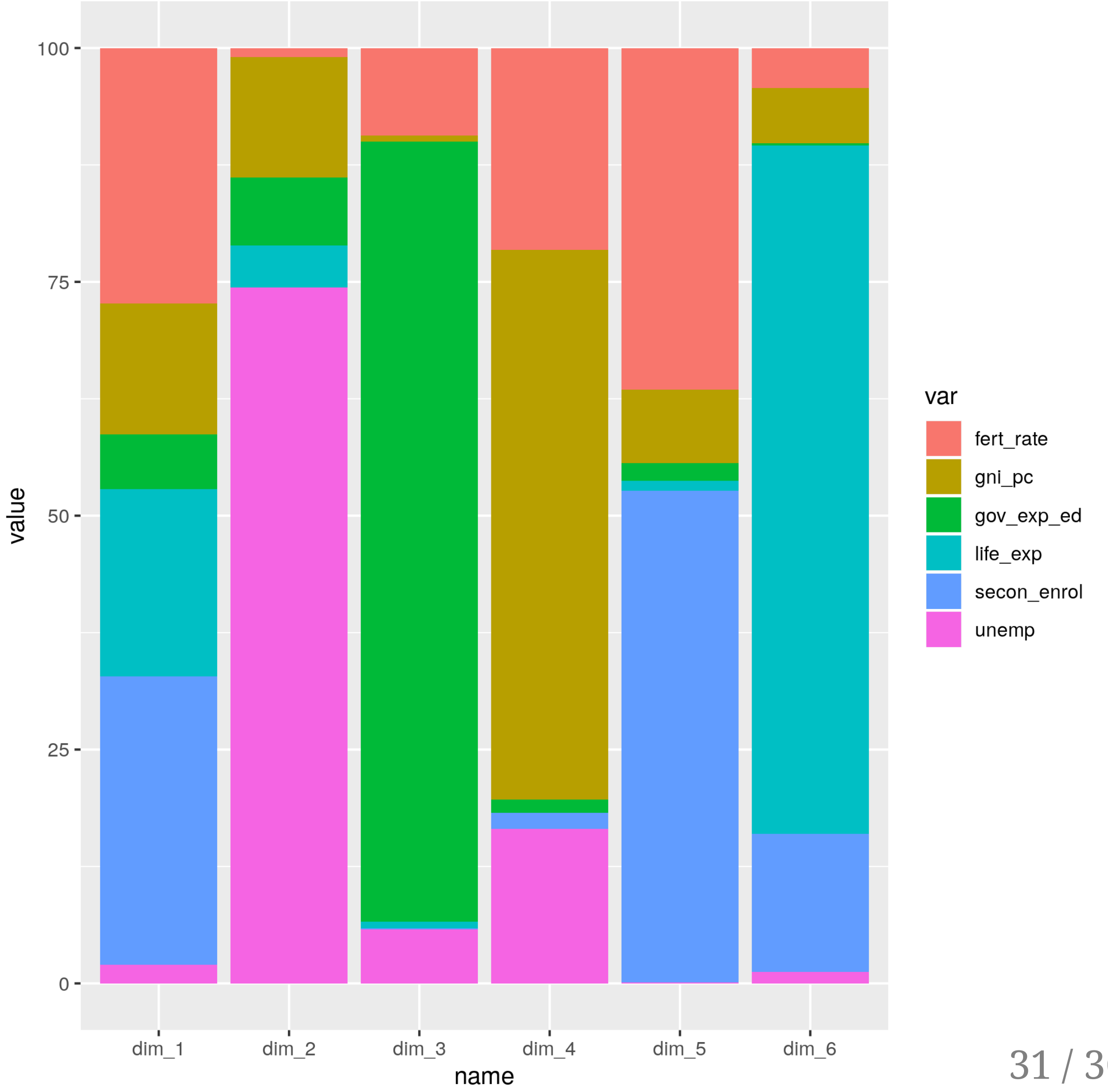
Unsupervised learning: Principal Components

Variables

```r
(pca_var <- get_pca_var(pca))
```

```
## Principal Component Analysis Results for variables
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

```r
pca_var$contrib %>%
  as.data.frame() %>%
  tibble::rownames_to_column("var") %>%
  janitor::clean_names() %>%
  pivot_longer(cols = dim_1:dim_6) %>%
  ggplot(aes(name, value, fill=var))+
  geom_col()
```



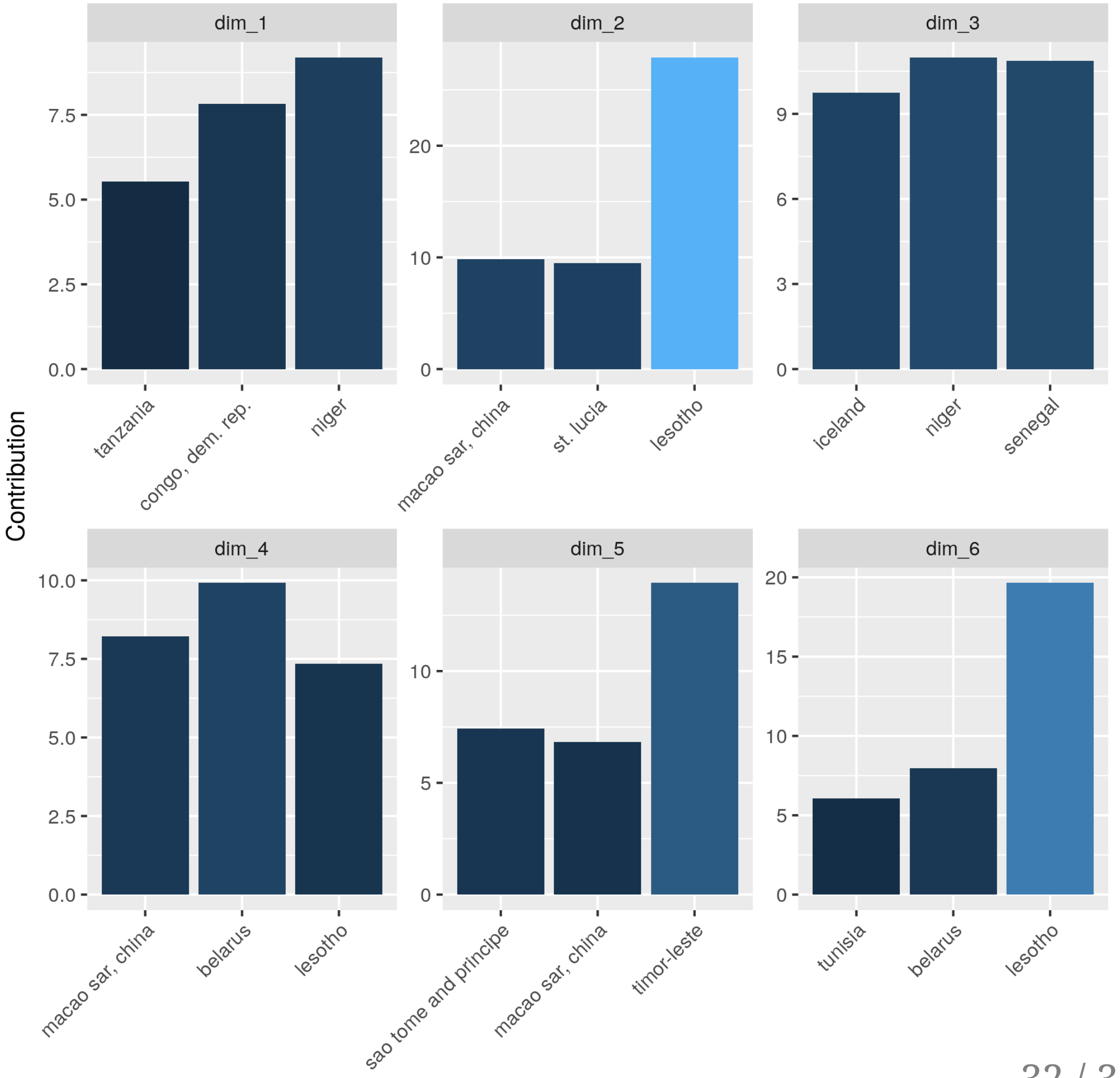
Unsupervised learning: Principal Components

Observations

```
(pca_var <- get_pca_ind(pca))
```

```
## Principal Component Analysis Results for individuals
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the individuals"
## 2 "$cos2"    "Cos2 for the individuals"
## 3 "$contrib" "contributions of the individuals"
```

```
pca_var$contrib %>%
    as.data.frame() %>%
    tibble::rownames_to_column("obs") %>%
    janitor::clean_names() %>%
    pivot_longer(cols = dim_1:dim_6, names_to = "comp") %>%
    group_by(comp) %>%
    top_n(n = 3, wt = value) %>%
    mutate(rank=rank(desc(value))) %>%
    ggplot(aes(reorder(obs, value), value))+
    geom_col(aes(fill=value))+
    facet_wrap(~comp, scales = "free")+
    theme(axis.text.x = element_text(angle = 45, hjust = 1), l
    labs(x="", y="Contribution")
```



References

1. Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). Cluster Analysis. Wiley.
2. Géron, A. (2017). Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. " O'Reilly Media, Inc.".
3. Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
4. Murphy, K. P. (2012). Machine learning: a probabilistic perspective. MIT press.

Appendix

	Individual i			
	Outcome	1	0	Total
Individual j	1	a	b	$a + b$
	0	c	d	$c + d$
	Total	$a + c$	$b + d$	$p = a + b + c + d$

Counts of binary outcomes for two individuals (source: Everitt et al 2011)

Appendix

Measure	Formula
S1: Matching coefficient	$s_{ij} = (a+d)/(a+b+c+d)$
S2: Jaccard coefficient (Jaccard, 1908)	$s_{ij} = a/(a+b+c)$
S3: Rogers and Tanimoto (1960)	$s_{ij} = (a+d)/[a+2(b+c)+d]$
S4: Sneath and Sokal (1973)	$s_{ij} = a/[a+2(b+c)]$
S5: Gower and Legendre (1986)	$s_{ij} = (a+d) \Big/ \left[a+\frac{1}{2}(b+c)+d\right]$
S6: Gower and Legendre (1986)	$s_{ij} = a \Big/ \left[a+\frac{1}{2}(b+c)\right]$

Similarity measures for binary data (source: Everitt et al 2011)

Appendix

Measure	Formula
D1: Euclidean distance	$d_{ij} = \left[\sum_{k=1}^{p} w_k^2 (x_{ik} - x_{jk})^2\right]^{1/2}$
D2: City block distance	$d_{ij} = \sum_{k=1}^{p} w_k \lvert x_{ik} - x_{jk} \rvert$
D3: Minkowski distance	$d_{ij} = \left(\sum_{k=1}^{p} w_k^r \lvert x_{ik} - x_{jk} \rvert^r\right)^{1/r} \quad (r \geq 1)$
D4: Canberra distance (Lance and Williams, 1966)	$d_{ij} = \begin{cases} 0 & \text{for } x_{ik} = x_{jk} = 0 \\ \sum_{k=1}^{p} w_k \lvert x_{ik} - x_{jk} \rvert / (\lvert x_{ik} \rvert + \lvert x_{jk} \rvert) & \text{for } x_{ik} \neq 0 \text{ or } x_{jk} \neq 0 \end{cases}$
D5: Pearson correlation	$\delta_{ij} = \left(1 - \phi_{ij}\right)/2$ with $\phi_{ij} = \sum_{k=1}^{p} w_k (x_{ik} - \bar{x}_{i\cdot})(x_{jk} - \bar{x}_{j\cdot}) \Big/ \left[\sum_{k=1}^{p} w_k (x_{ik} - \bar{x}_{i\cdot})^2 \sum_{k=1}^{p} w_k (x_{jk} - \bar{x}_{j\cdot})^2\right]^{1/2}$ where $\bar{x}_{i\cdot} = \sum_{k=1}^{p} w_k x_{ik} \Big/ \sum_{k=1}^{p} w_k$
D6: Angular separation	$\delta_{ij} = \left(1 - \phi_{ij}\right)/2$ with $\phi_{ij} = \sum_{k=1}^{p} w_k x_{ik} x_{jk} \Big/ \left(\sum_{k=1}^{p} w_k x_{ik}^2 \sum_{k=1}^{p} w_k x_{jk}^2\right)^{1/2}$

Dissimilarity measures for continuous data (source: Everitt et al 2011)