Clustering

Rui

May 26, 2026

目录

1 这章在讲什么

聚类分析（clustering）是**无监督学习**的入门：手里只有一堆样本的特征，没有「正确答案」标签，目标是把样本分成若干组，让组内尽量相似、组间尽量不同。心理学里常用来：把人格特征分群（找「人格类型」）、把消费者分群（市场细分）、把脑区分群（神经科学）。

两大主流方法的选择

层次聚类（hierarchical clustering）：从每个样本各自一类开始，反复合并最近的两类，直到所有样本合成一类。结果是一棵**树状图**（dendrogram），用户根据树的高度切一刀决定 k。优点：不需要预设 k、可视化直观。缺点：$O(n^3)$ 复杂度、对大数据慢；一旦合并不可回退（greedy）。

k-means：预设类数 k，先随机选 k 个中心，反复「分配最近中心 → 重新计算中心」直到收敛。优点：快、$O(nki)$。缺点：要预设 k、对初始值敏感（多次随机初始化取最优）、对球形 / 等方差簇假设强。

k 选择：层次聚类看树状图肘部；k-means 看**轮廓系数**（silhouette）或 elbow 法。在 R 里 `factoextra::fviz_nbclust(data, kmeans, method = "silhouette")` 一行画出 $k = 2 \sim 10$ 的得分图。

距离度量也很关键：欧氏距离对量纲敏感（必须先标准化）、曼哈顿距离对异常值更稳健、Gower 距离能处理混合类型变量（数值 + 分类）。聚类前先想清楚「相似」在你的场景下是什么意思。

```
library("parameters")  #for clustering summary
library("see")
library("factoextra")  #for clustering plotting
library(bruceR)
library(tidyverse)
```

```
athlete = import("data13-02.xlsx") %>% select(-1)
```

```
##   Successfully imported: 10 obs. of 4 variables
```

2 K-Means

2.1 Computation

- `x`: numeric matrix-like data
- `centers`: either the number of clusters, say kk, or a set of initial (distinct) cluster centres. If a number, a random set of (distinct) rows in x is chosen as the initial centres.
- `iter.max`: the maximum number of iterations allowed.
- `nstart`: if centers is a number, how many random sets should be chosen. Trying several random starts is often recommended.

```
kmean_results = kmeans(athlete, centers = 4,
                       iter.max = 10, nstart = 1)
```

kmeans returns an object of class "kmeans" which has a print and a fitted method. It is a list with at least the following components:

- `cluster` : A vector of integers (from 1:k) indicating the cluster to which each point is allocated.
- `centers`: A matrix of cluster centres.
- `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.

```
kmean_results$cluster  #group labels for each observation
```

```
##  [1] 3 2 4 3 2 1 4 2 4 1
```

```
kmean_results$centers  #centers
```

```
##         x1       x2       x3
```

```
## 1 120.5000 19.00000 44.50000
## 2 121.6667 18.33333 43.00000
## 3 124.5000 20.00000 44.50000
## 4 121.0000 17.00000 41.66667
```

```
kmean_results$totss  # total variance
kmean_results$withinss  # within-group variance
kmean_results$tot.withinss # sum of within SS
kmean_results$betweenss  #between-group variance
```

```
## [1] 51.6
## [1] 1.000000 1.333333 1.000000 2.666667
## [1] 6
## [1] 45.6
```

```
kmean_results$size  # how many cases in each cluster
```

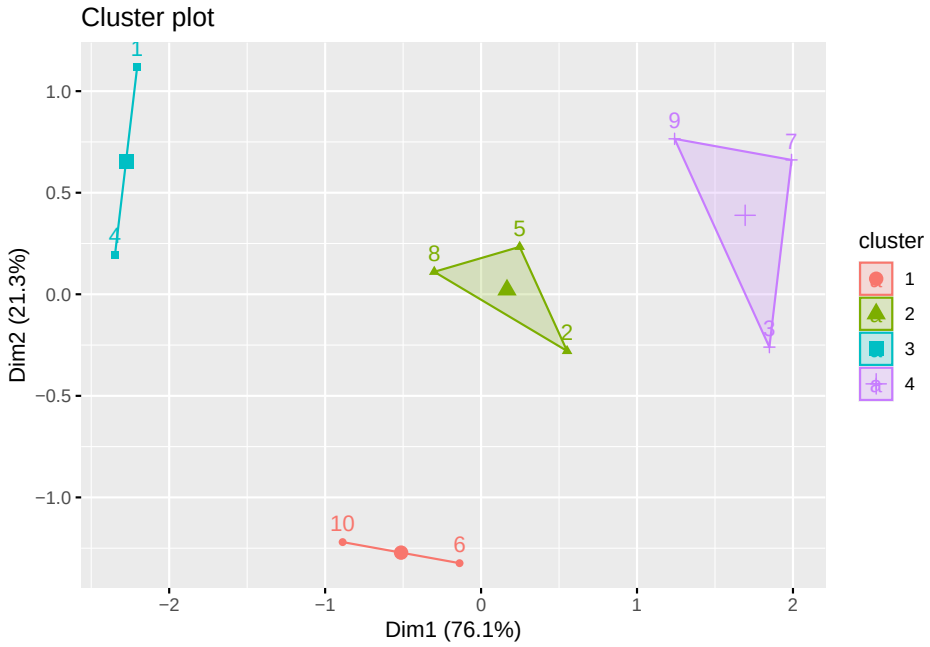
```
## [1] 2 3 2 3
```

2.2 Visualization

Use `factoextra::fviz_cluster(object, data)`. For more details about useful parameters, see the help document. Remember that `object` and `data` are two most indispensable parameters.

```
fviz_cluster(kmean_results, data = athlete)
```

```
## Registered S3 methods overwritten by 'broom':
##   method            from
##   tidy.glht         jtools
##   tidy.summary.glht jtools
```



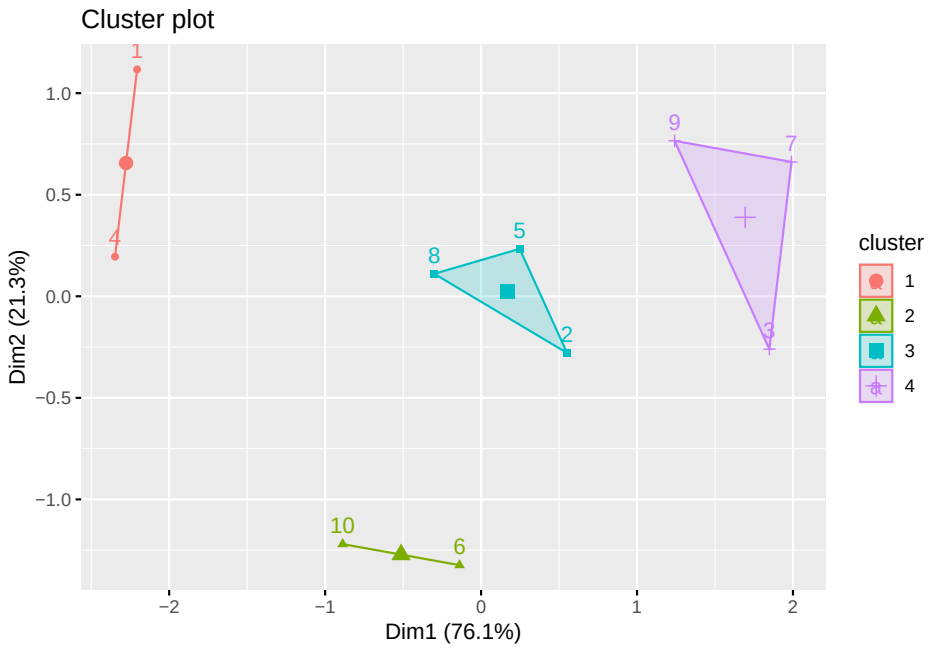
One way to start is from random centers; another way is to specify a few sets of centers and start from them.

Clearly, starting from different centers, either random or specified, the results may vary, though not significant. If the iteration will converge eventually, then given sufficiently-large iteration time, the center will be determined and fixed.

```
initial_centers = import("data13-02a.xlsx") %>% select(-1)
```

```
##   Successfully imported: 4 obs. of 4 variables
```

```
kmean_results_initialized = kmeans(athlete, initial_centers ,
                                   iter.max = 10, nstart = 1)
fviz_cluster(kmean_results_initialized, data = athlete)
```



3 Hierachical Clustering

```
automobile = import("data13-01.xlsx") %>%
  select(c('type','price','engine_s','horsepow',
           'wheelbas','width','length','curb_wgt',
           'fuel_cap','mpg')) %>% drop_na() %>% scale()
```

```
##   Successfully imported: 157 obs. of 27 variables
```

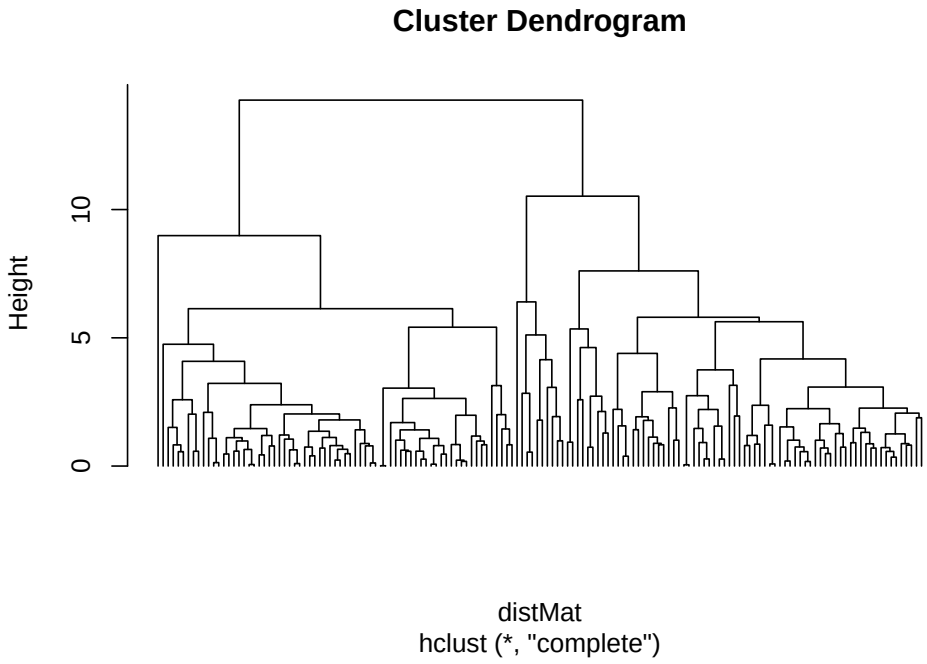
Use "`hclust()`" from base R and transmit a **dissimilarity** structure produced by "`dist()`".

```
distMat = dist(automobile, method = "euclidean") # Euclidean distance matrix
hc = hclust(distMat)  # hierarchical clustering, using the default method
```

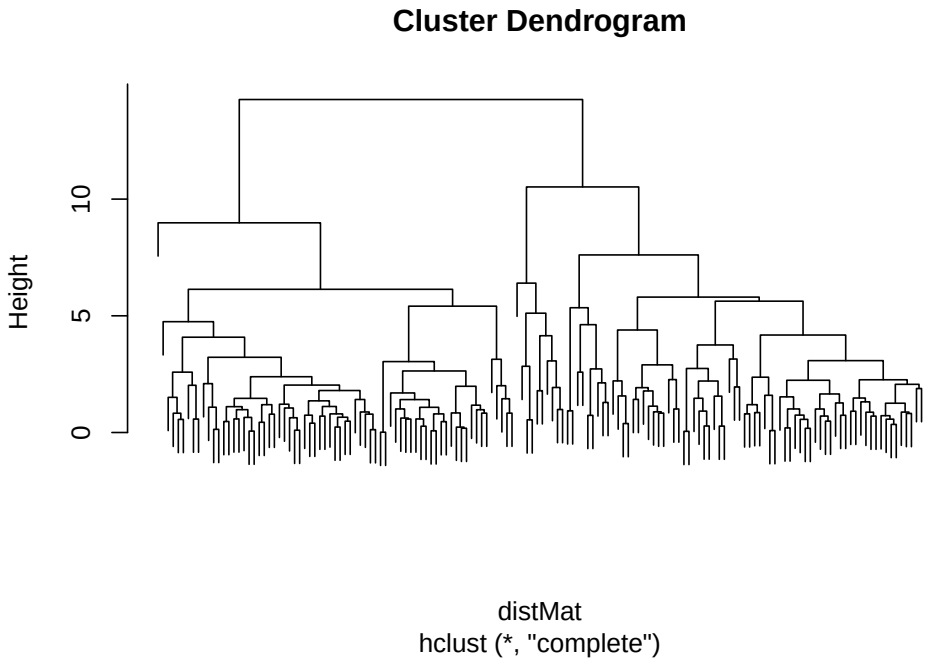
Use "`plot(hclust_obj)`" to visualize the hierarchical clustering. Note that you should specify "`hang = -1`" to ensure all nodes touch the same horizontal line and looks decent. If the sample isn't that large, set "`labels =`

TRUE" (by default) to see the corresponding observations.

```
plot(hc,labels=FALSE,hang = -1)
```

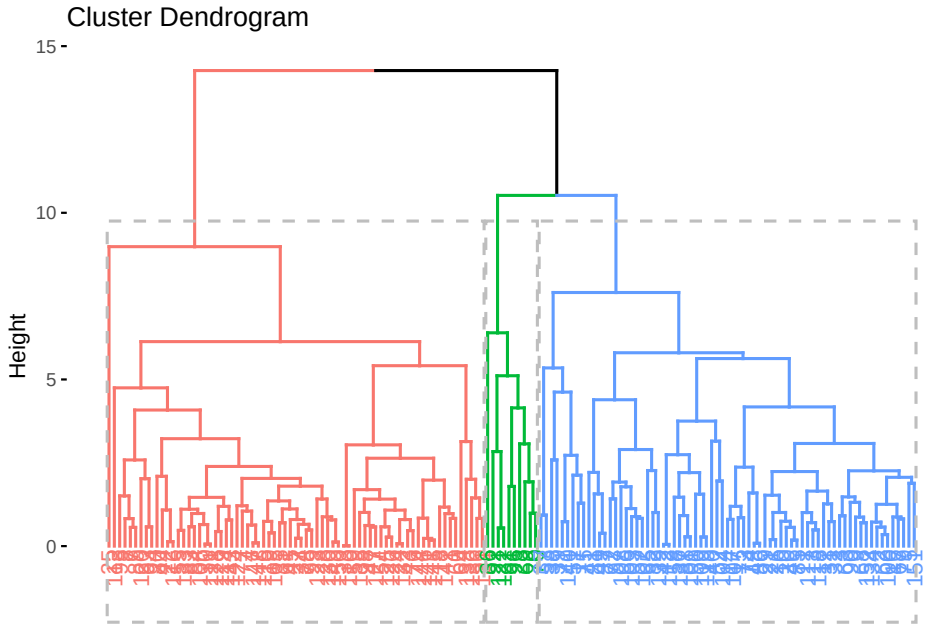



```
plot(hc,labels=FALSE)
```



Moreover, you can draw dendrogram with grouping.

```
fviz_dend(hc,k=3, rect = T)
```



Also, you can also check how many members for each cluster and make a frequency table.

Note that the grouping might change if we change cluster number. And this in essence is the pros of hierarchical clustering.

```
memberLabels = cutree(hc,3)
table(memberLabels)
```

```
## memberLabels
##  1  2  3
## 71 71 10
```

Then, we can get the centers of cluters by "`mean`" summary statistics. Use "`aggregate()`" to achieve that.

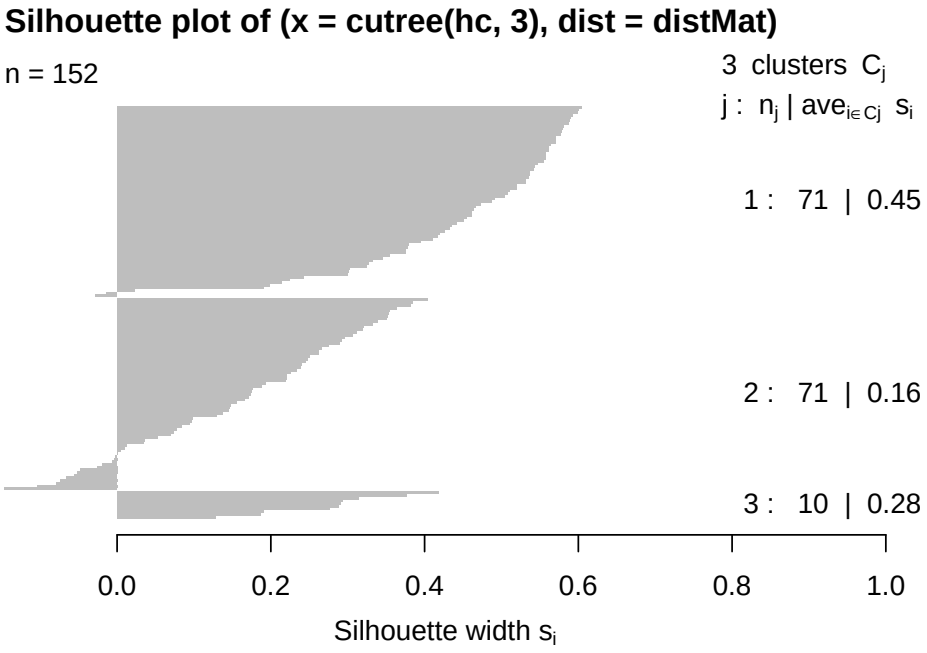
- `x`: usually a data frame.
- `by`: a list of grouping elements, each **as long** as the variables in the data frame x. The elements are coerced to factors before use.
- `FUN`: a function to compute the summary statistics which can be applied to all data subsets.

```
aggregate(automobile,list(memberLabels),mean)
```

```
##   Group.1       type      price   engine_s   horsepow   wheelbas      width
## 1       1 -0.4362472 -0.4985325 -0.7231739 -0.6579479 -0.6259158 -0.6459818
## 2       2  0.5201409  0.1193177  0.5017194  0.3293009  0.6601116  0.5979645
## 3       3 -0.5956452  2.6924245  1.5723270  2.3333938 -0.2427902  0.3409229
##        length   curb_wgt   fuel_cap        mpg
## 1 -0.65105171 -0.7853229 -0.7200528  0.6620751
## 2  0.65553921  0.7236353  0.6270241 -0.5592928
## 3 -0.03186123  0.4379815  0.6605042 -0.7297543
```

Moreover, we can use Silhouette plot to check whether some cases are outliers.

```
library(cluster)
plot(silhouette(cutree(hc,3), distMat))
```

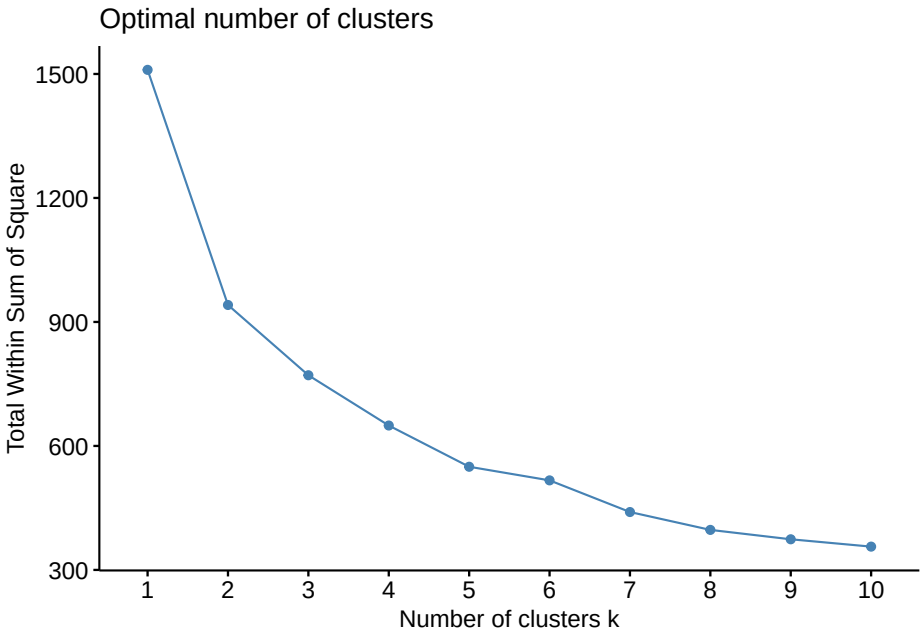


4 No. of Clusters

4.1 Within-Cluster SS

Looking for the "elbow" where the SS starts to slow down its drop.

```
fviz_nbclust(automobile, kmeans, method = "wss", print.summary=TRUE)
```

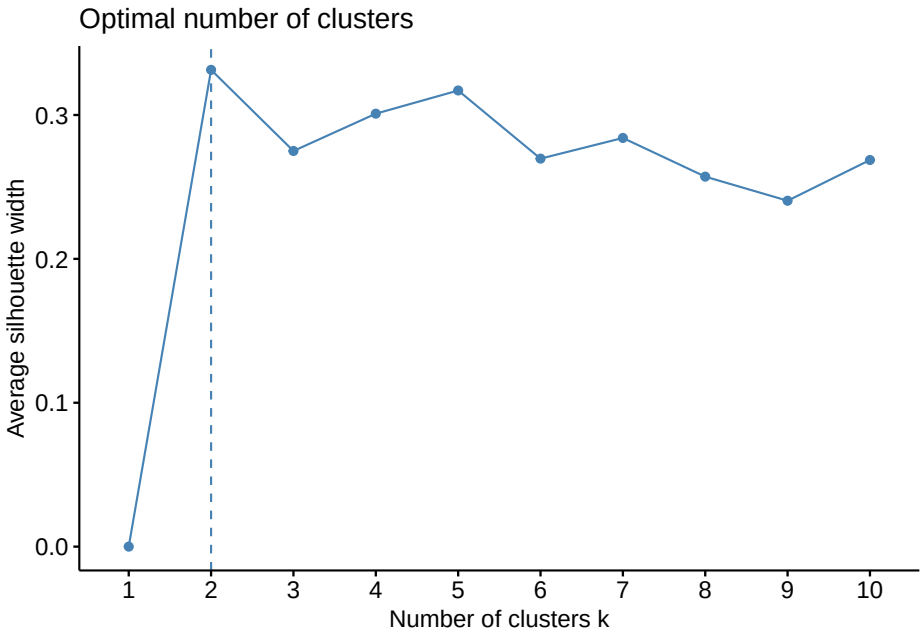


here suggests 2 or 3.

4.2 Silhouette Method

Looking for the value that sets the highest average silhouette width, which indicates the optimal number of clusters.

```
fviz_nbclust(automobile, kmeans, method = "silhouette",print.summary=TRUE)
```



4.3 Majority Rule

`n_clusters()` is the main function to find out the optimal numbers of clusters present in the data based on the maximum consensus of a large number of methods.

Essentially, there exist many methods to determine the optimal number of clusters, each with pros and cons, benefits and limitations. The main `n_clusters()` function proposes to run all of them, and find out the number of clusters that is suggested by the **majority** of methods (in case of ties, it will select the most parsimonious solution with fewer clusters).

Note that we also implement some specific, commonly used methods, like the Elbow or the Gap method, with their own visualization functionalities.

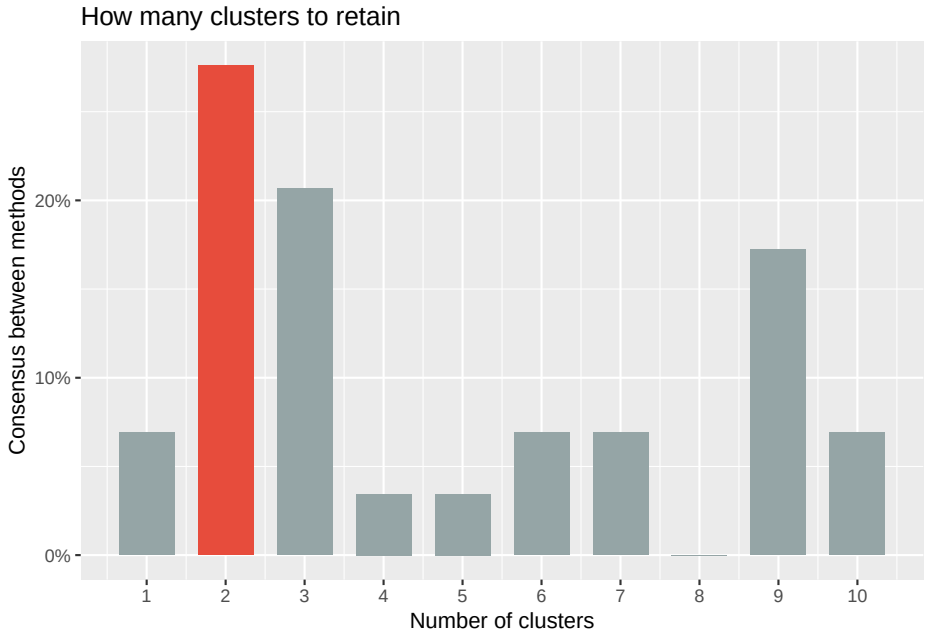
Also note that the `n_clusters()` function will do the normalization automatically (by default).

```
n_clusters(data.frame(automobile))
```

5 Method Agreement Procedure:

The choice of 2 clusters is supported by 8 (27.59%) methods out of 29 (Elbow, Silhouette, Ch, Ratkowsky, Mcclain, SDindex, Mixture (VEV), Mixture (EEV)).

```
plot(n_clusters(data.frame(automobile)))
```



6 Two Step Clustering

Some people prefer using hierarchical methods first to obtain the number of clusters and their centroids. Then use these as inputs to run non-hierarchical analysis. This approach is called Two-Step Clustering.

```
# two step clustering in R, not required
library(prcr)
d <- pisaUSA15
m3 <- create_profiles_cluster(d,
                              broad_interest, enjoyment, instrumental_mot, self_efficac
                              n_profiles = 3)
summary(m3)
```

7 Clustering for Cases and Variables

Infrastructure construction includes various dimensions such as transportation, postal and telecommunications, water supply and power supply. "construction.xlsx" records the development level of 16 infrastructure construction aspects in 7 countries. Please answer the following questions:

1. Please use hierarchical methods to determine how many different types of infrastructure construction are included in these 16 indicators, and briefly explain the possible meanings of each type.

2. Calculate the average value of each type of indicator for each country in the previous question as the overall measure of infrastructure construction for that type (assuming this operation is reasonable). Use the k-means method to cluster the 7 countries based on their different overall indicators, distinguish between developed and developing countries, and determine which countries belong to each category.

7.1 Q1

```
library(factoextra)
library(showtext)
showtext_auto()
```

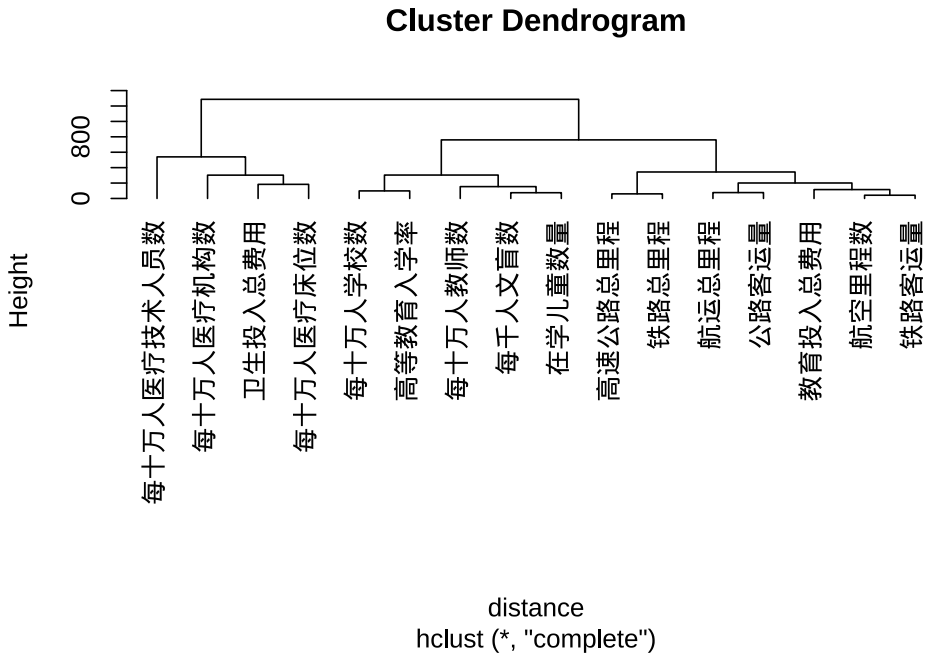
```
cons = import('construction.xlsx')
```

```
##   Successfully imported: 16 obs. of 8 variables
```

```
rownames(cons) = cons[,1]
cons = cons %>% select(-1)
attach(cons)
```

```
distance = dist(cons)
hc = hclust(distance)
```

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



From the cluster dendrogram above, it's suggested to classify all 16 indexes into 3 grand categories. From indexes contained in each category, we can infer that the three dimensions are:

- Medical resources
- Education level
- Traffic capacity

7.2 Q2

```
grouplabel = cutree(hc,k = 3)
```

```
km_country = aggregate(cons,list(grouplabel),mean) %>%
  select(-1) %>% t() %>%
  kmeans(centers = 2)
country_label = km_country$cluster
country_label
```

```
## A B C D E F G
## 1 1 1 2 2 2 1
```

From the k-means method, we can see that the seven countries are classified into two grand categories, developed and developing countries, where the first group consists of D, E & F, and all else as the second group. However, clustering analysis doesn't directly tell us the attributes of two groups, therefore, we have to "label" the groups by checking the data.

Note that here D, E & F are labeled as group "2".

```
cons %>% t() %>% as.data.frame() %>% mutate(
  develop = country_label
) %>% group_by(develop) %>% summarise_all(mean) %>%
  stargazer::stargazer(header = FALSE,
                       title = 'Index Comparison Between Groups',
                       flip = T,summary = FALSE)
```

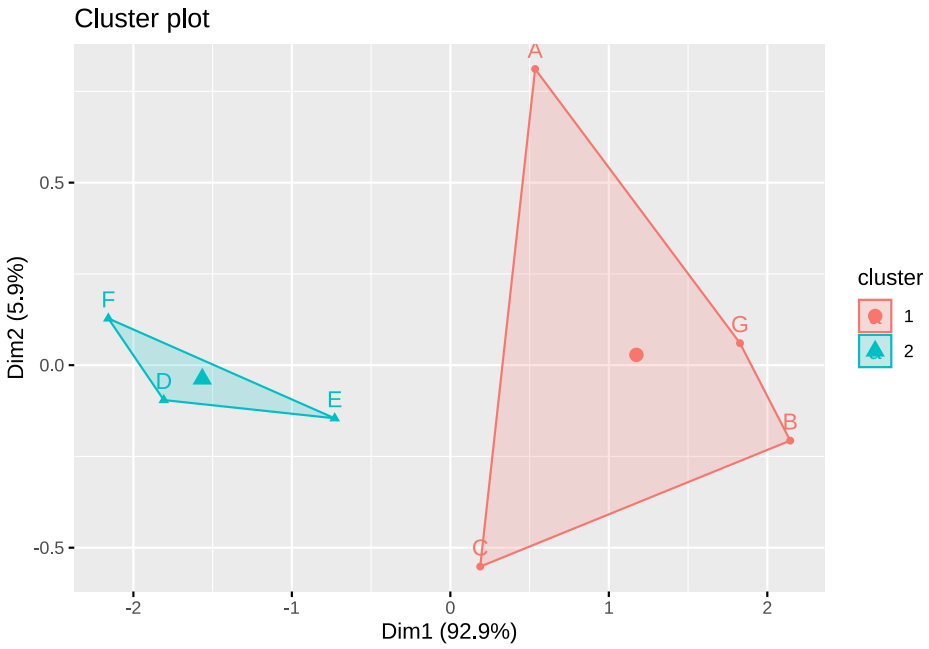
From the summarization table, apparently that among all the 16 indexes, group 1 outbeats group 2 (i.e., D, E & F). Therefore, we can say

- Developing countries: D, E, F;
- Developed countries: A, B, C, G.

```
fviz_cluster(km_country,
             data = aggregate(cons,list(grouplabel),mean) %>%
               select(-1) %>% t())
```

表 1: Index Comparison Between Groups

	1	2
develop	1	2
每十万人医疗机构数	361.035	214.980
每十万人医疗技术人员数	617.005	244.910
高速公路总里程	351.375	112.713
铁路总里程	374.023	119.360
每十万人教师数	158.267	78.060
每十万人学校数	63.805	26.087
每千人文盲数	115.995	63.923
高等教育入学率	26.145	6.663
在学儿童数量	129.545	62.687
航空里程数	244.132	103.263
航运总里程	290.225	135.253
铁路客运量	260.318	95.403
公路客运量	296.095	113.263
教育投入总费用	220.273	75.360
卫生投入总费用	489.022	229.947
每十万人医疗床位数	495.518	182.137



All done. Finally,

```
detach(cons)
```

8 容易踩的坑

聚类最常见的几个误区

1. **不标准化就跑欧氏距离**：当变量量纲差异大（如「年龄」0–100 vs「月收入」1000–50000），月收入的差异会主导距离计算，年龄几乎被忽略。解法：聚类前先 `scale()` 或者用相关距离 `1 - cor()`。
2. **结果稳定 = 聚类正确**：聚类没有「对错」，只有「在某种距离 + 算法下的最稳定划分」。同一份数据用 k-means 和 hierarchical 可能得出截然不同的分组——所以要交叉验证：bootstrap 重抽 + 比较结果是否一致（`fpc::clusterboot`）。
3. **把聚类当作"自动的因子分析"**：聚类是把**样本**分群，因子分析是把**变量**降维。要给被试分人格类型，是聚类；要把 50 道人格量表题归到 5 个潜在维度，是 EFA。不要混了。

4. **聚类结果直接当作组别变量做 ANOVA**：聚类是数据驱动出来的「组」，组内方差被最小化、组间方差被最大化——再拿这些组去跑 ANOVA 几乎一定显著（循环论证）。要验证聚类有效性，应该用**未参与聚类的外部变量**做组间比较。