Cluster Analysis(CA)

Clustering is a method of unsupervised(no response variable to predict) learning, which is used to group data points into several clusters based on similarity measures without pre-lableled outcomes. The objective is to have data points that are more similar to each other within the same cluster and dissimilar to the data points in other clusters.

5 different types of clustering methods

1. Partitioning methods
2. Hierarchical clustering
3. Fuzzy clustering
4. Density-based clustering
5. Model-based clustering

```
library(cluster)
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(magrittr)

data(wine,package="rattle")
head(wine)
```

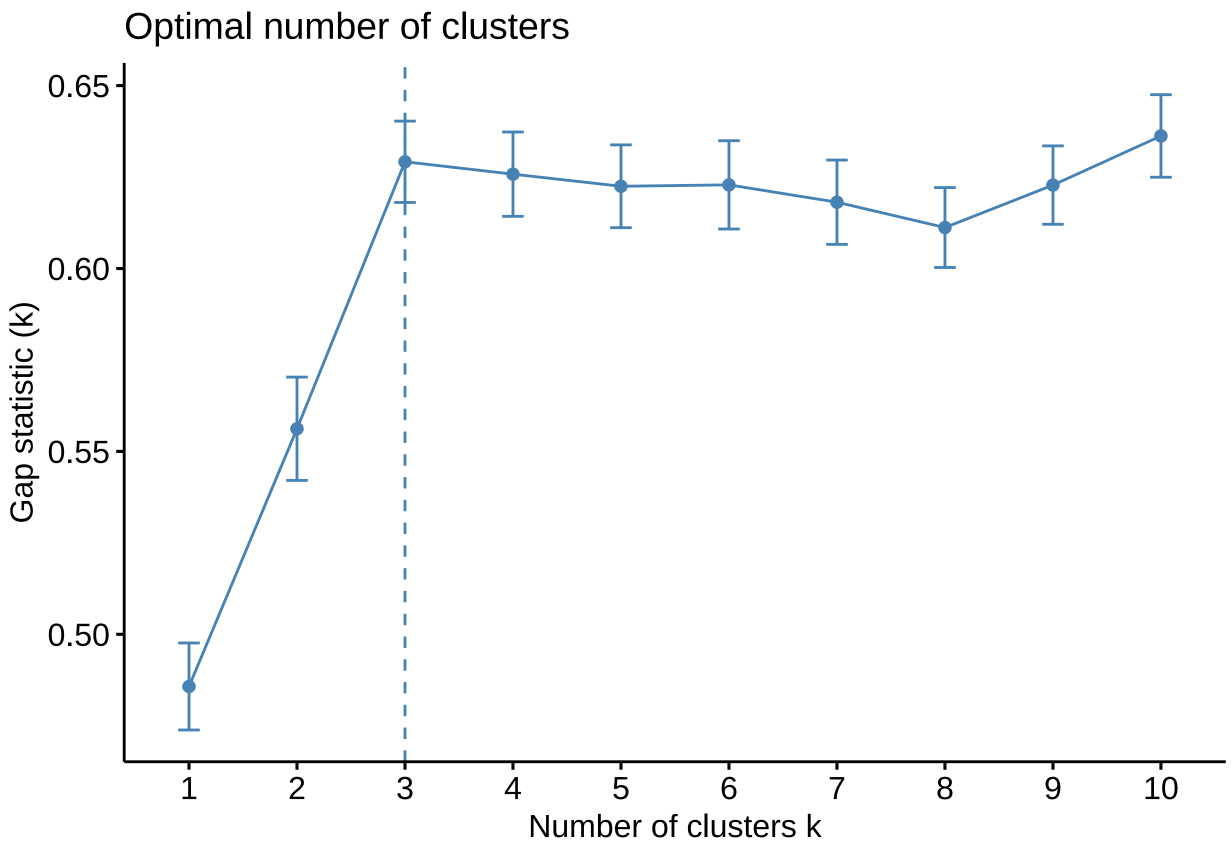
```
##   Type Alcohol Malic  Ash Alcalinity Magnesium Phenols Flavanoids Nonflavanoids
## 1    1   14.23  1.71 2.43       15.6       127    2.80       3.06          0.28
## 2    1   13.20  1.78 2.14       11.2       100    2.65       2.76          0.26
## 3    1   13.16  2.36 2.67       18.6       101    2.80       3.24          0.30
## 4    1   14.37  1.95 2.50       16.8       113    3.85       3.49          0.24
## 5    1   13.24  2.59 2.87       21.0       118    2.80       2.69          0.39
## 6    1   14.20  1.76 2.45       15.2       112    3.27       3.39          0.34
##   Proanthocyanins Color  Hue Dilution Proline
## 1            2.29  5.64 1.04     3.92    1065
## 2            1.28  4.38 1.05     3.40    1050
## 3            2.81  5.68 1.03     3.17    1185
## 4            2.18  7.80 0.86     3.45    1480
## 5            1.82  4.32 1.04     2.93     735
## 6            1.97  6.75 1.05     2.85    1450
```

```
data1 <- scale(wine[-1])
data2 <- scale(mtcars)
```

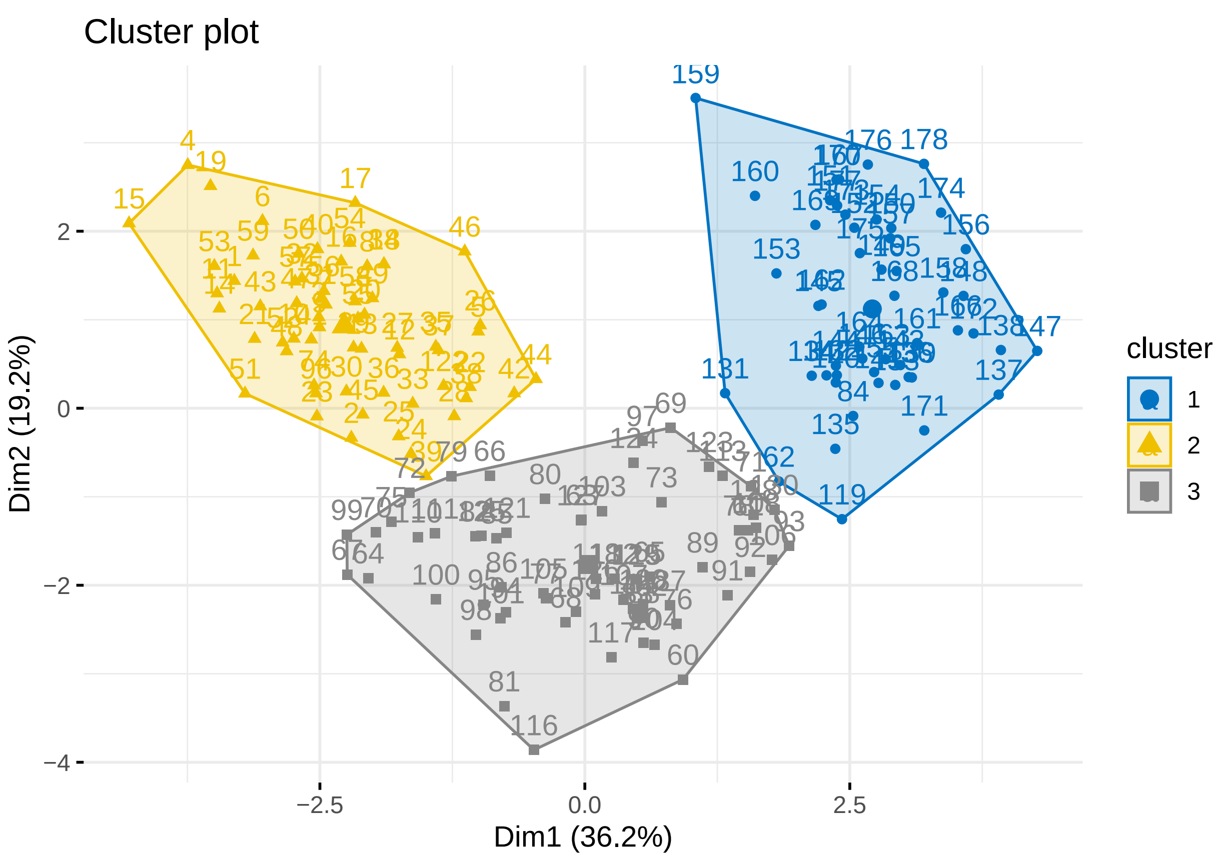
1. Partitioning clustering

Partitioning methods divide the dataset into a set of **k groups or partitions**, where k is a parameter spceified by the user. The most well-known partitioning clustering method is the **K-means algorithm**. It starts with an initial set of k centroids, where each point is assigne to closest centroid, forming k clusters. The centroids are then updated based on the points assigned to each cluster. This process repeats until the positions of the centroids no longer change significantly. Partitioning methods are efficient and are best suited for finding spherical-shaped clusters in large datasets.

```
#Firstly, determining the optimal number of clusters
fviz_nbclust(data1,kmeans,method="gap_stat")
```



```
#From the plot, the suggested number of cluster is 3.
#The gap statistic: compares the total within intra-cluster variation for different numbers of clusters

# Compute and visuaize k-means clustering
set.seed(123)
km.res <- kmeans(data1,3,nstart=25)
# Visualize
library(factoextra)
fviz_cluster(km.res,data=data1,ellipse.type="convex",palette="jco",ggtheme=theme_minimal())
```

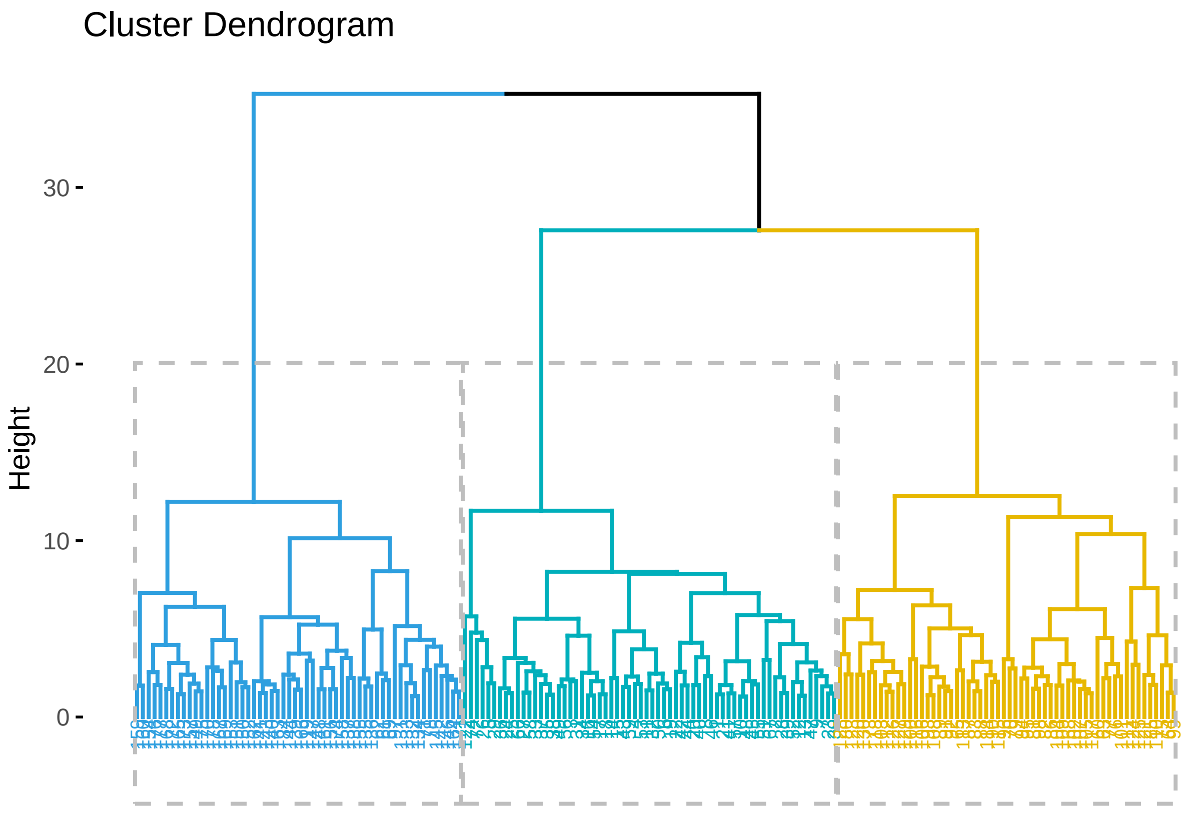


2. Hierarchical clustering

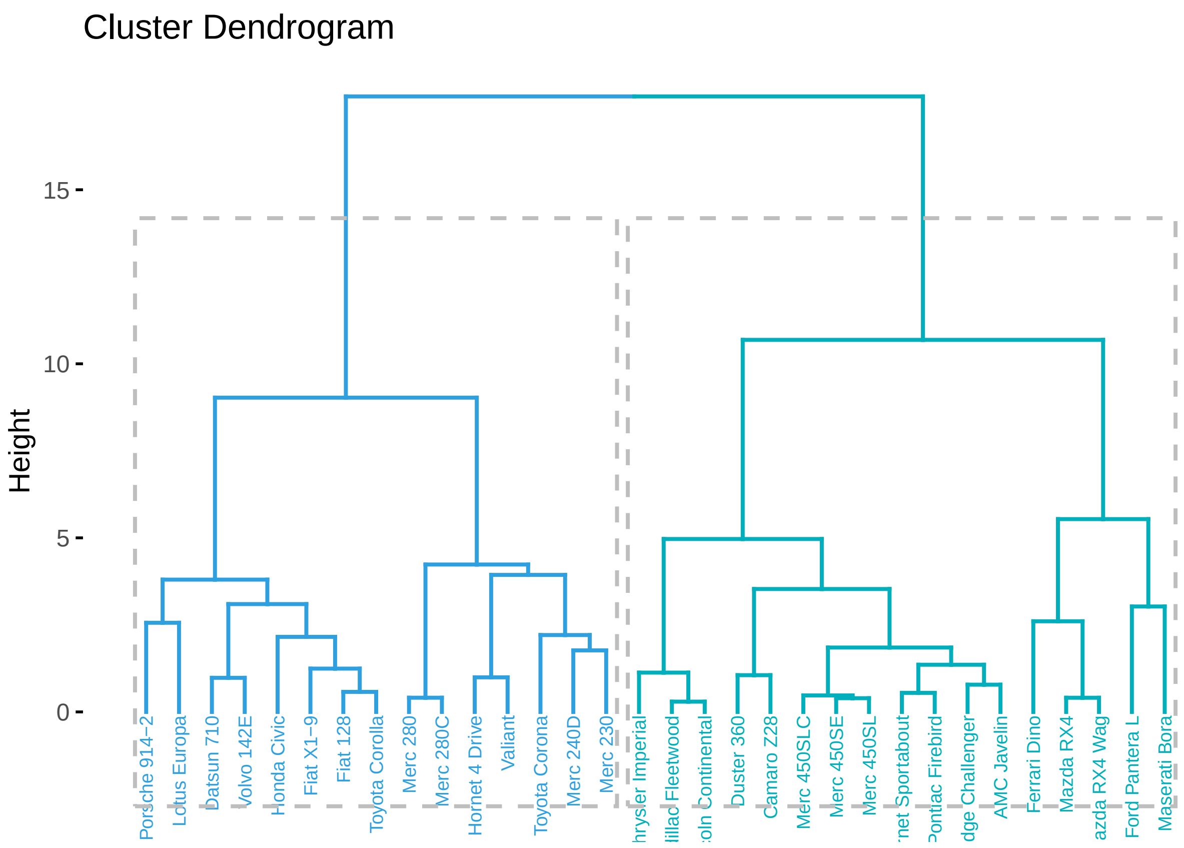
Hierarchical clustering builds a hierarchy of clusters either in a bottom-up approach (agglomerative clusterig) or a **top-down approach** (divisive clustering). In agglomerative clustering, each data point starts in its own cluster, and pairs pairs of clusters are merged as one moved up the hierarchy. In divisive clustering, **all points start in one cluster that is iteratively split into smaller clusters**. Hierarchical clustering is flexible regarding the distance metric and does not require a pre-specified number of clusters. The result is often presented in a **dendrogram**, which illustrates the series of steps taken by the algorithm to merge or split clusters.

```
# This method does not require to pre-specify the number of clusters.
# Compute hierachical clustering.
res.hc <- data1 %>%
  scale() %>% #Scale the data
  dist(method="euclidean") %>% #Compute dissimiarlity matrix
  hclust(method="ward.D2") #Compute hierachical clustering
# Ward's method minimizes the total within-cluster variance

# Visualize using factoextra
# Cut in 3 groups and color by groups
fviz_dend(res.hc, k=3, # cut in four groups
          cex=0.5,# label size
          k_colors=c("#2E9FDF","#00AFBB","#E7B800"),
          color_labels_by_k=TRUE, # color labels by groups
          rect=TRUE # add rectangle around groups
          )
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## i The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



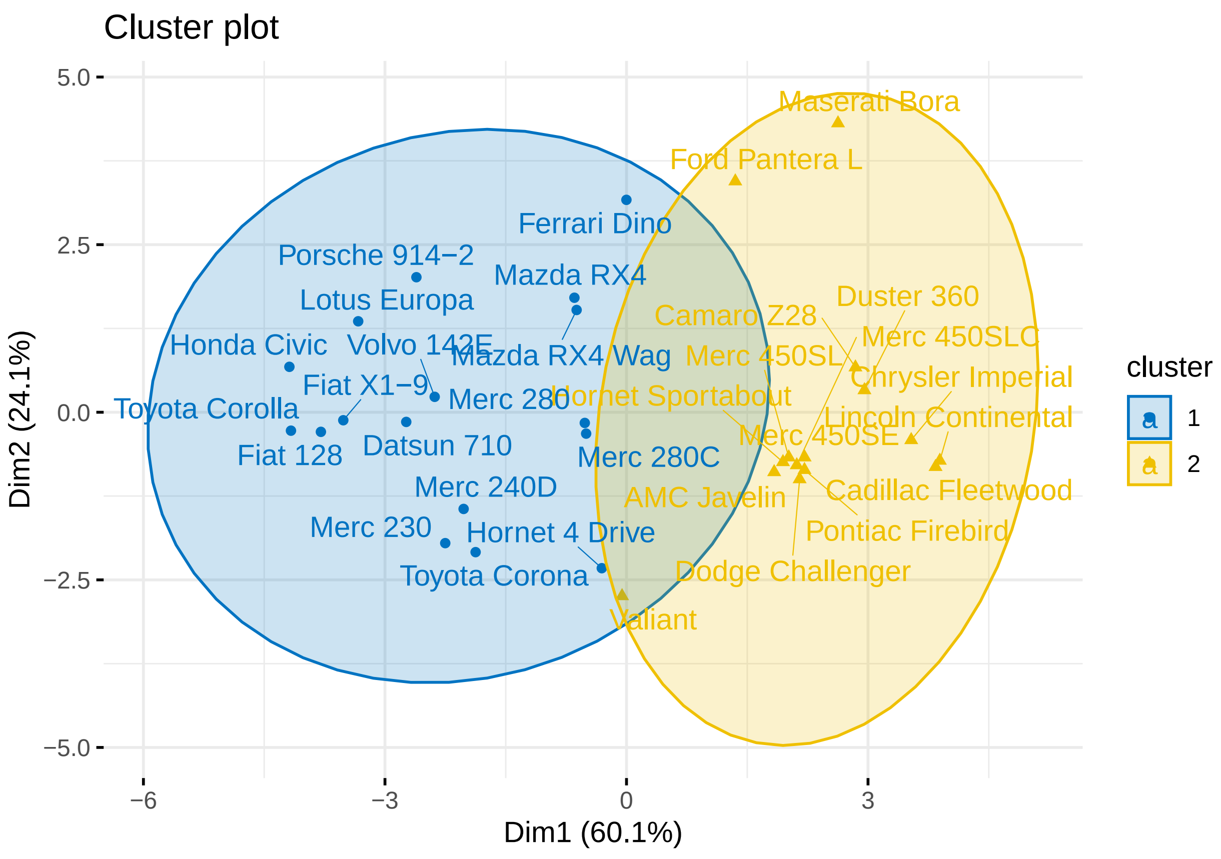
```
#dataset2: mtcars
res.hc <- data2 %>%
  scale() %>%
  dist(method="euclidean") %>%
  hclust(method="ward.D2")
fviz_dend(res.hc,k=2,cex=0.5,k_colors=c("#2E9FDF","#00AFBB"),color_labels_by_k=TRUE,rect=TRUE)
```



3. Fuzzy clustering

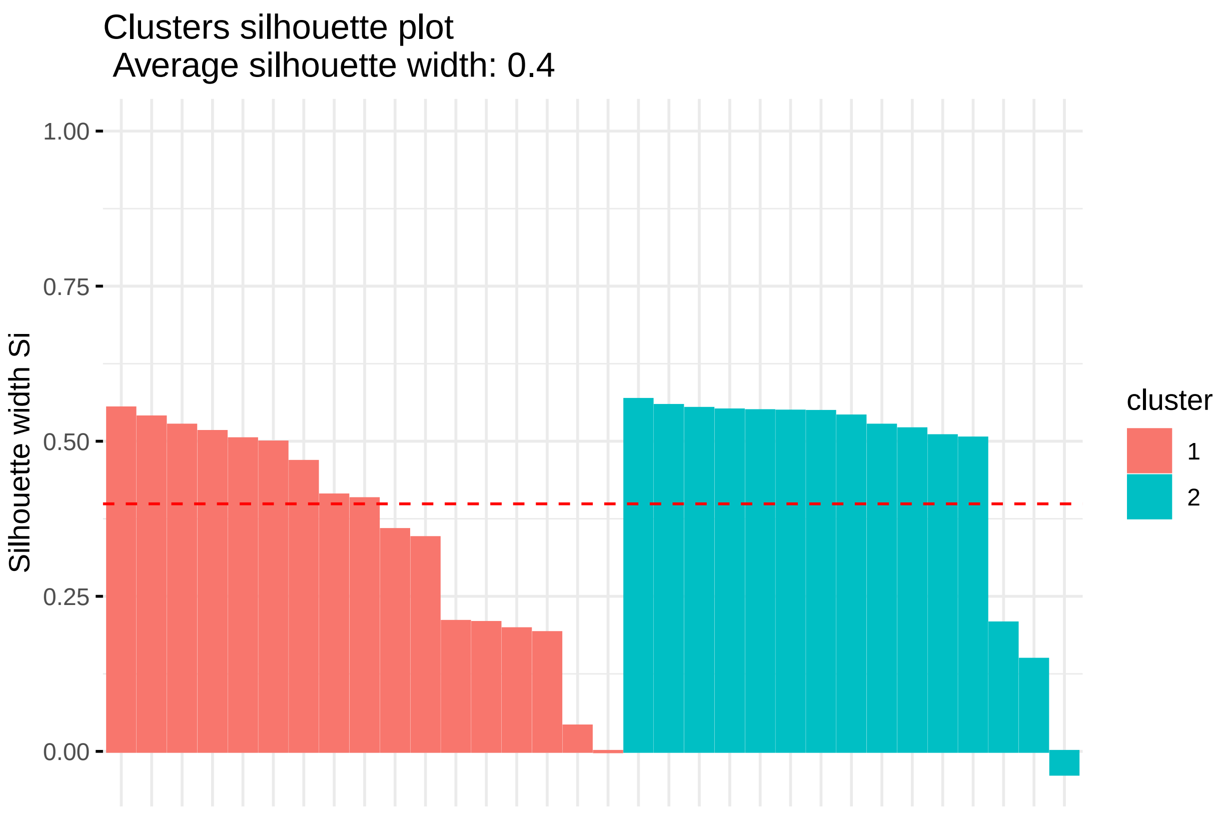
Fuzzy clustering, or soft clustering, **allows points to belong to multiple clusters** with varying degrees of membership. This contrasts with hard clustering method like K-means, where each point belongs to exactly one cluster. The most popular method of fuzzy clustering is the Fuzzy C-Means algorithm, which assigns each point a probability or membership level for each cluster. The algorithm iteratively updates the cluster centers and the membership levels for each point until the membership levels do not change significantly. Fuzzy clustering is useful when the boundaries between clusters are not clear-cut.

```
# An alternative way to k-means
library(cluster)
library(factoextra)
res.fanny <- fanny(data2,2) # Compute fuzzy clustering with k=3
fviz_cluster(res.fanny, ellipse.type="norm",repel=TRUE,palette="jco",ggtheme=theme_minimal(),legend="ri
```



```
fviz_silhouette(res.fanny,palette="ico",ggtheme=theme_minimal())
```

```
##   cluster size ave.sil.width
## 1       1   17          0.35
## 2       2   15          0.45
```



4. Density-based clustering

Density-based clustring methods **identify clusters as areas of higher density** within the data space, surrounded by areas of lower density, DBSCAN(Desntiy-Based Spatial Clustering of Applications with Noise) is well-known example of this approach. It defines a cluster as a maximal set of density-connected points. DBSCAN can find arbitrarily shaped clusters and is good at **separating noise from the clusters**. It requires two parameters: "eps", which specifies how close points should be to each other to be considered part of cluster, and "minPts", the minimum number of points required to form a dense region.

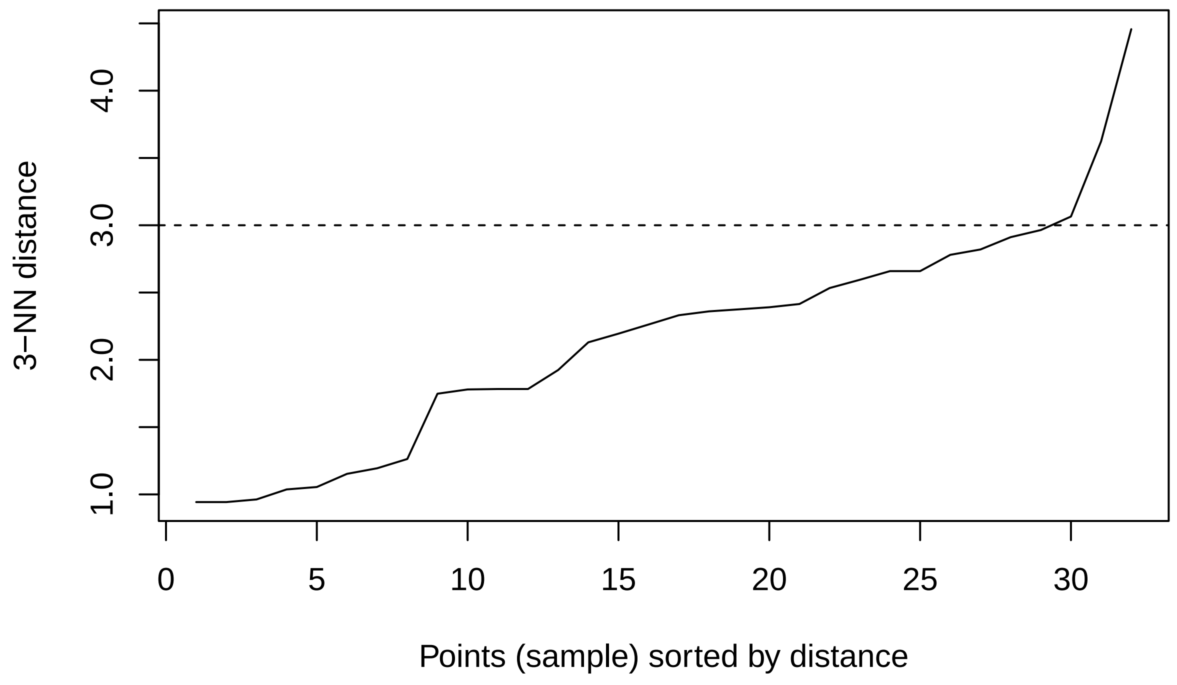
```
# load the data
data("multishapes",package="factoextra")
df <- multishapes[,1:2]

# find out an eps value
library(dbscan)
```

```
## Warning: package 'dbscan' was built under R version 4.2.3
```

```
##
## Attaching package: 'dbscan'
```

```
## The following object is masked from 'package:stats':
##
##     as.dendrogram
```

```
dbscan::kNNdistplot(data2,k=3)
abline(h=3,lty=2)
```



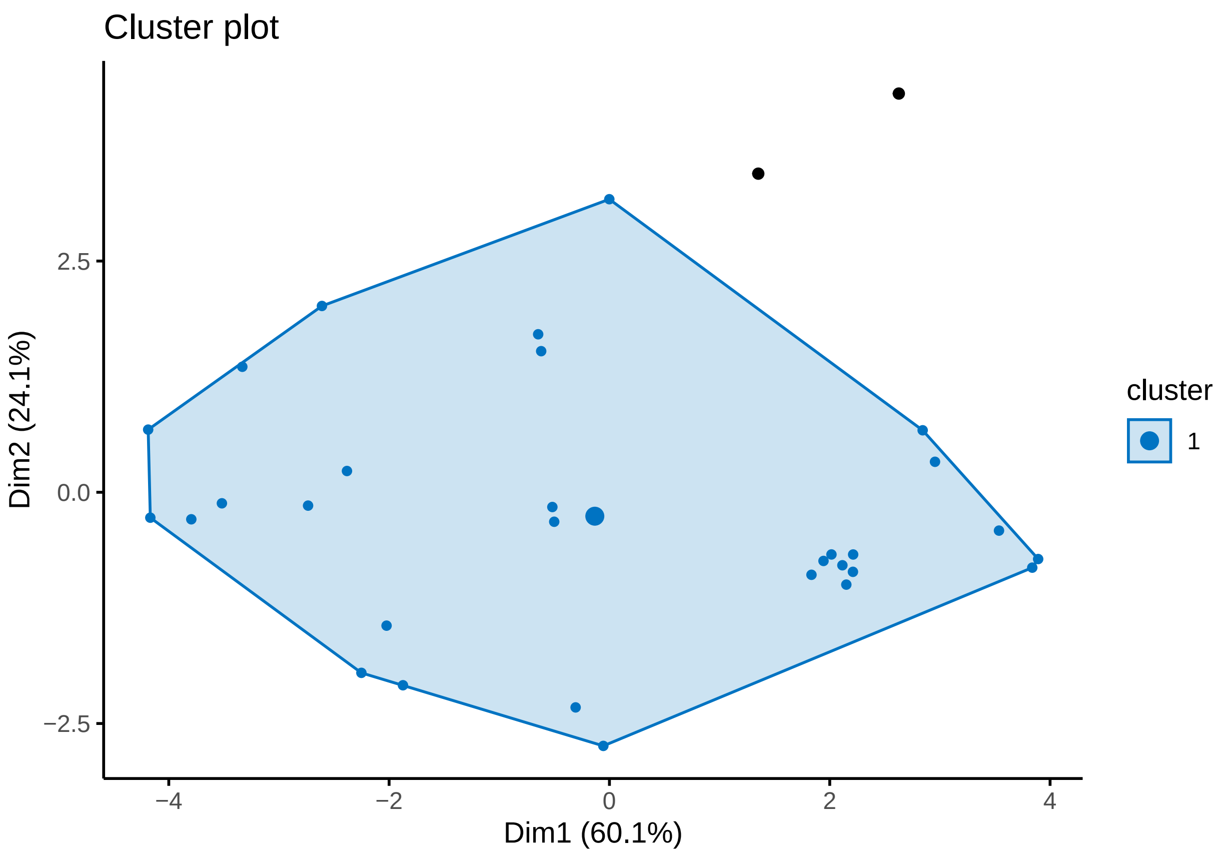
```r
# comopute DBSCAN using fpc package
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 4.2.3
```

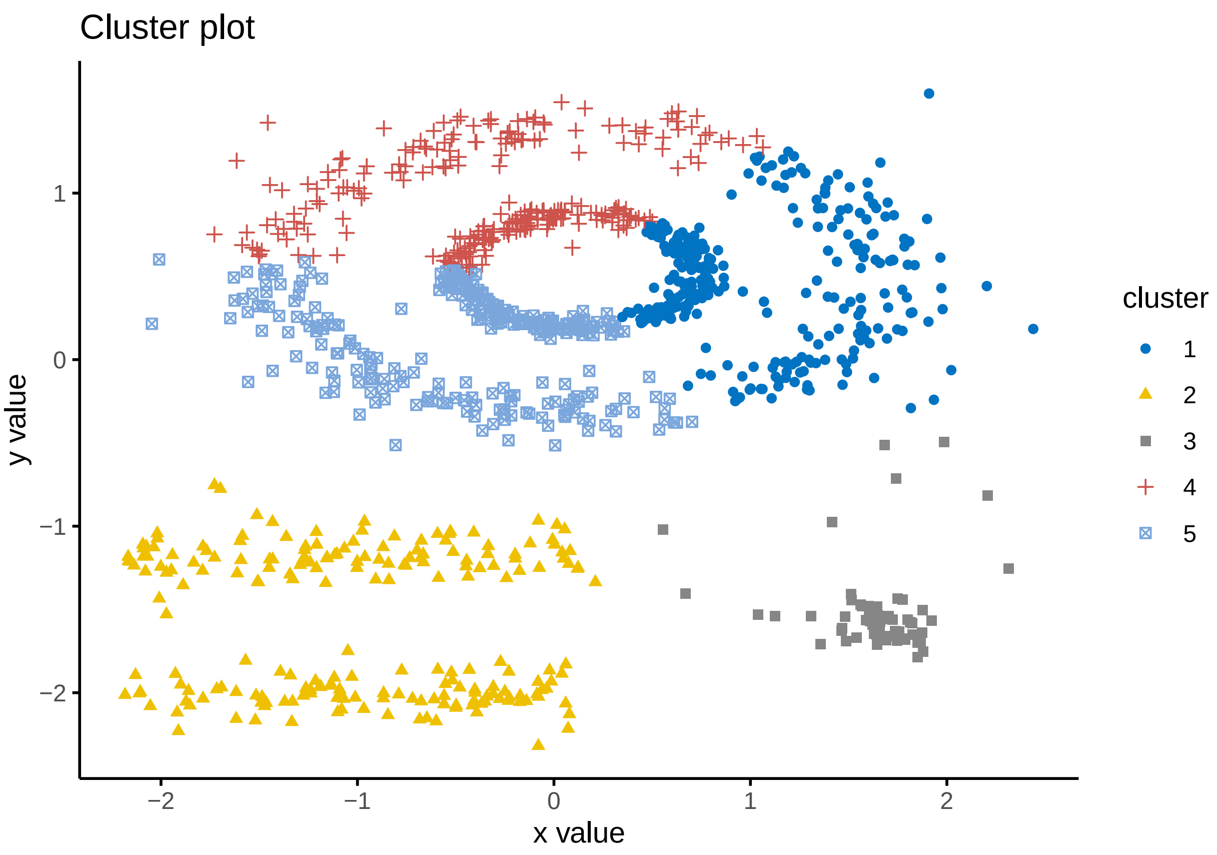
```
##
## Attaching package: 'fpc'
```

```
## The following object is masked from 'package:dbscan':
##
##     dbscan
```

```r
set.seed(123)
db <- fpc::dbscan(data2,eps=3,MinPts=5)

# Plot DBSCAN results
library(factoextra)
fviz_cluster(db, data=data2, stand=FALSE,geom="point",palette="jco",ggtheme=theme_classic())
```



```
# Not good

# A better example
data("multishapes")
df <- multishapes[,1:2]
set.seed(123)
km.res <- kmeans(df,5,nstart=25)
fviz_cluster(km.res,df,geom="point",ellipse=FALSE,show.clust.cent = FALSE,palette="jco",ggtheme=theme_cl
```



5. Model-based clustering

Model-based clustering approaches assume that the data is generated from a mixture of finite distributions, where each distribution represents a cluster. A common technique is the Gaussian Mixture Model(GMM), where it is assumed that the data is generated from a mixture of several Gaussian distributions with unknow parameters. The Expectation-Maximization (EM) algorithm is typically used to estimate the parameters of the Gaussian distributions. Model-based clustering can infer the number of clusters based on the data, and it accommodates clusters of different shapes and sizes.

```
# data1: wine data
library(mclust)
```

```
## Warning: package 'mclust' was built under R version 4.2.3

## Package 'mclust' version 6.1
## Type 'citation("mclust")' for citing this R package in publications.
```

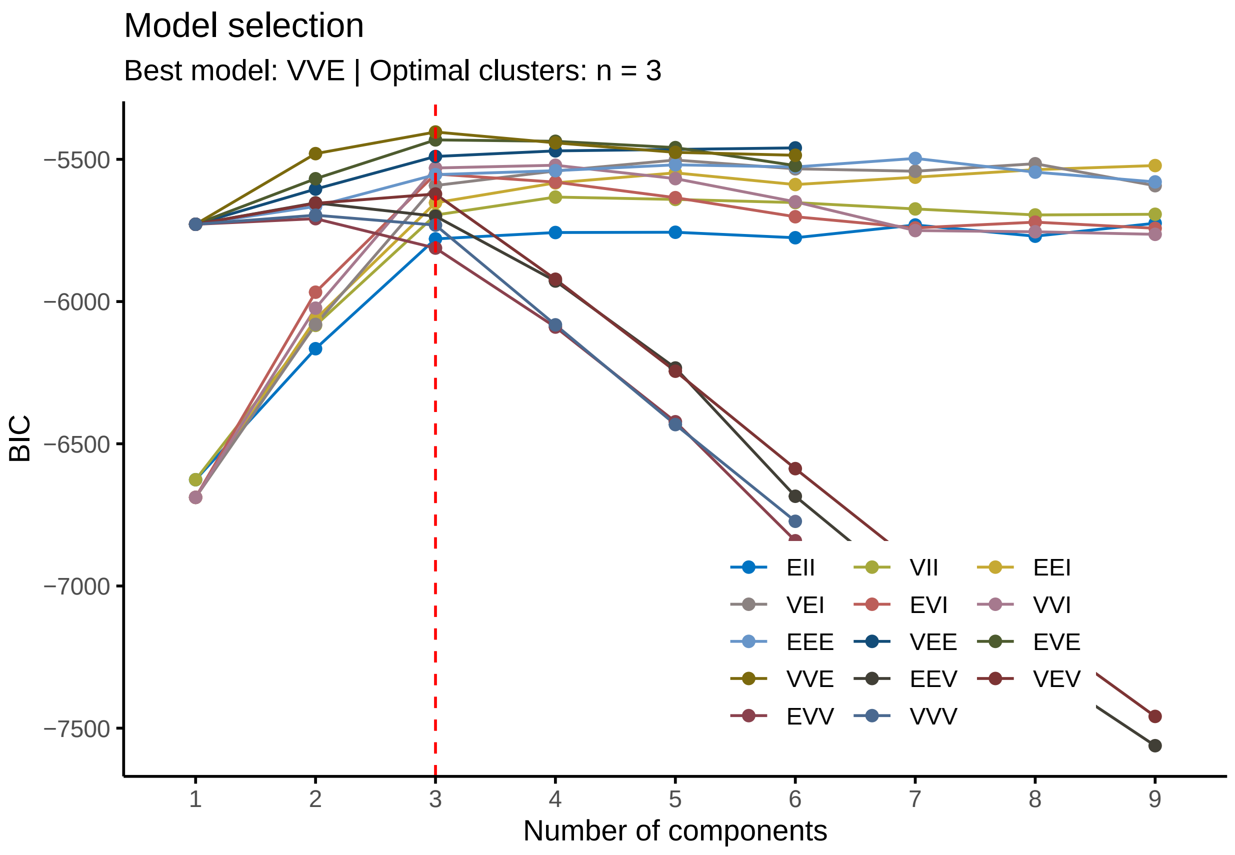
```
mc1 <- Mclust(data1) #Model-based clustering
summary(mc1)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVE (ellipsoidal, equal orientation) model with 3 components:
##
##  log-likelihood   n  df       BIC       ICL
##       -2292.553 178 158 -5403.829 -5404.793
##
## Clustering table:
```

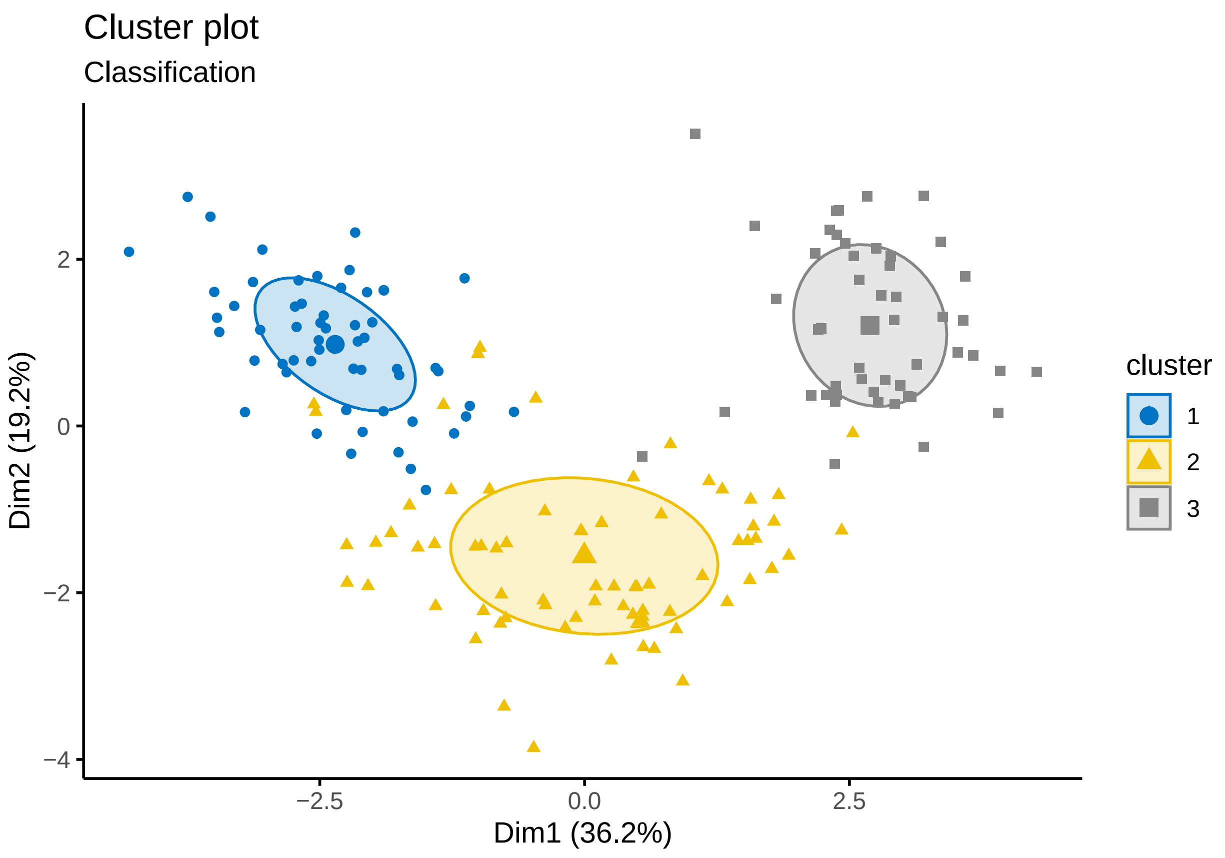
```
##  1  2  3
## 56 73 49
```

```
# the number of cluster: 3

library(factoextra)
# BIC values used for choosing the number of clusters
fviz_mclust(mc1,"BIC",palette="jco")
```

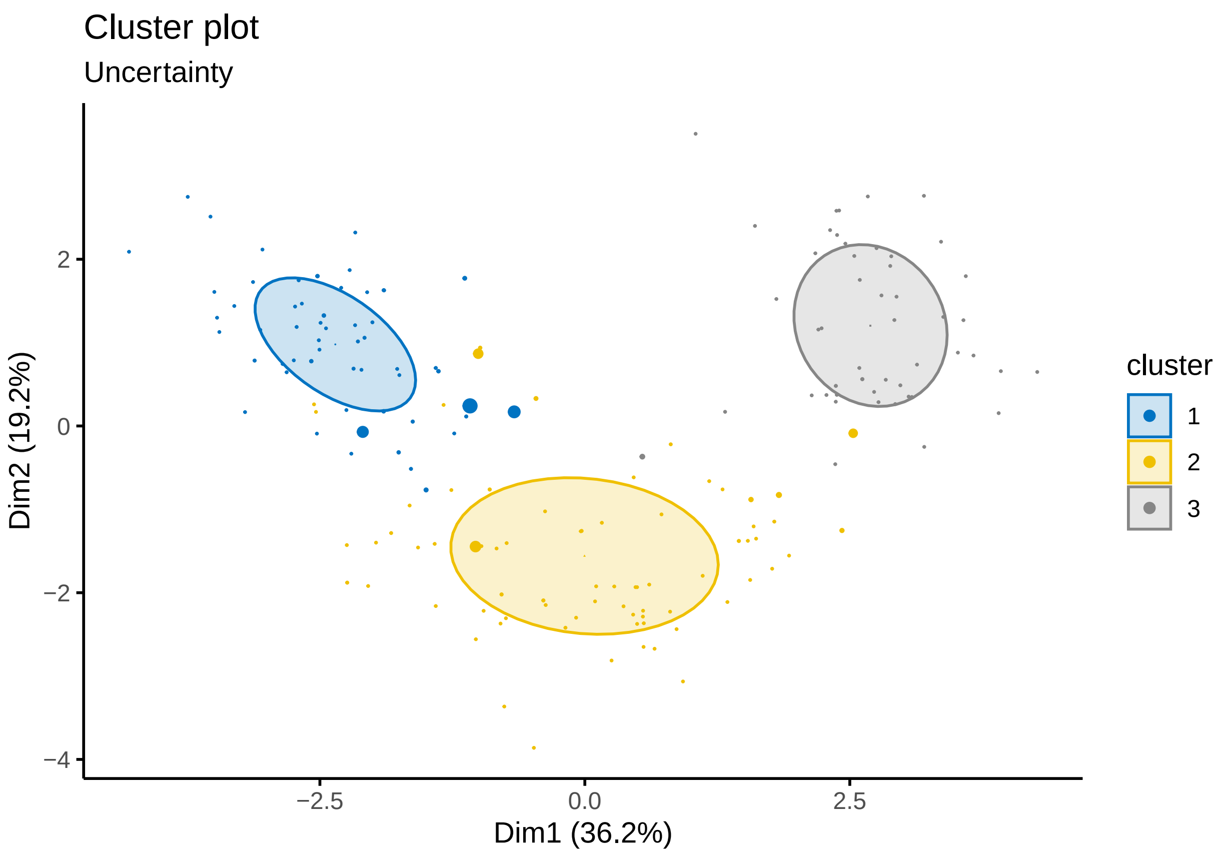
```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## i Please use `gather()` instead.
## i The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
# Classification: plot showing the clustering
# names of the Gaussian Mixture models, e.g.
# EII: Indicates that all clusters have equal volume, shape, and orientation
# VII: Suggests that clusters can have different volumes but must have the same shape and orientation
# VVE: Implies that clusters can have different volumes and shapes but share the same orientation
fviz_mclust(mc1, "classification", geom="point",pointsize=1.5,palette="jco") # Plots the classification
```



```r
# Classification uncertainty
fviz_mclust(mc1, "uncertainty", palette="jco")
```



Cluster plot
Uncertainty
Dim2 (19.2%)
Dim1 (36.2%)
2
0
−2
−4
−2.5
0.0
2.5
cluster
1
2
3