Unsupervised clustering

Hierarchical, k-means clustering

Presenter: Xuan Tran

Outline

1. Introduction
2. Distance
3. Dendrogram
4. Hierarchical clustering
5. K-means clustering

Machine learning algorithms

	Supervised learning	Unsupervised learning
Discrete outcome	Classification	**Clustering**
Continuous outcome	Regression	Dimensionality reduction

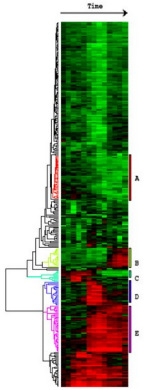
Application of Clustering

Image Segmentation

http://people.cs.uchicago.edu/~pff/segment/

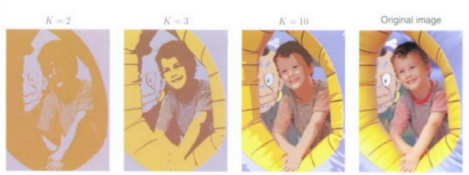
Clustering Search Results

Clustering gene expression data



Eisen et al, PNAS 1998

Vector quantization to compress images

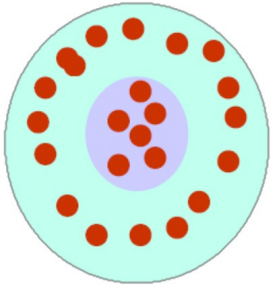


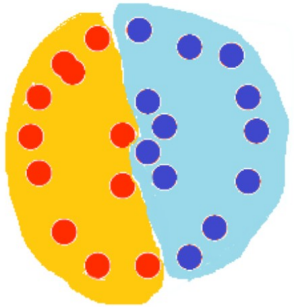
Bishop, PRML

Cluster analysis - Definition

The organization of unlabeled data into similarity groups called clusters.

A cluster is a collection of data items which are “similar” between them, and “dissimilar” to data items in other clusters.





Cluster analysis - Characteristics

Multivariate statistics (multiple Y variables)

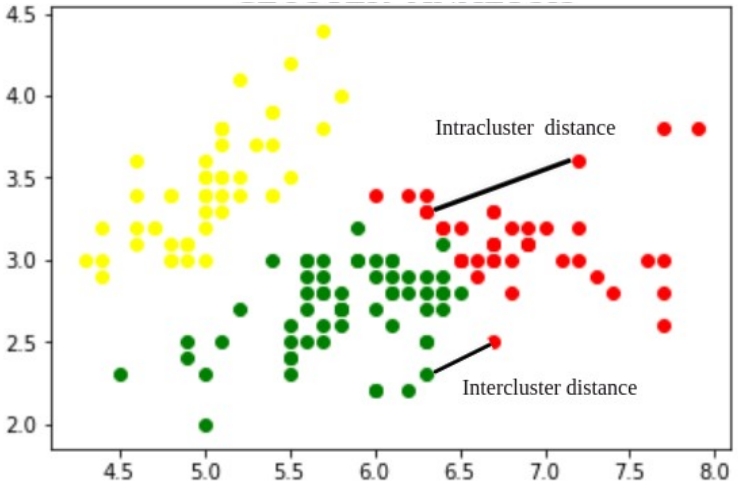
Using multiple variables to classify subjects (objects) into clusters

Search for similarities (for discovery rather than prediction)

Main idea: to find the “natural” groups that exist in a dataset

The analysis requires the measurement of similarities (or dissimilarities) between observations which is strongly dependent on the Distance metric

Distance


Intracluster distance
Intercluster distance
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

Euclidean distance

The most used distance function is the Euclidean distance, defined as the square root of the sum of the squared differences of n features between two patterns x and y:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

Euclidean distance - An example

Individual	Height	Weight
Mary	128	54
Bob	158	86
Julie	177	82
Mark	131	59

Distance between Mary and Bob:

$$\sqrt{(128-158)^2+(54-86)^2} = 43.863424$$

Distance between Mary and Julie:

$$\sqrt{(128-177)^2+(54-82)^2} = 56.435804$$

Euclidean distance - An example

Individual	Height	Weight	zHeight	zWeight
Mary	128	54	-0.88	-1.00
Bob	158	86	0.40	0.98
Julie	177	82	1.22	0.73
Mark	131	59	-0.75	-0.70
Mean ($\bar{x}$)	148.5	70.3		
SD (s)	23.3	16.1		

$$z = \frac{x_i - \bar{x}}{s}$$

```
name = c("Mary", "Bob", "Julie", "Mark")
height = c(128, 158, 177, 131)
weight = c(54, 86, 82, 59)
zheight = scale(height)
zweight = scale(weight)
```

Euclidean distance - An example

Individual	Height	Weight	zHeight	zWeight
Mary	128	54	-0.88	-1.00
Bob	158	86	0.40	0.98
Julie	177	82	1.22	0.73
Mark	131	59	-0.75	-0.70
Mean ($\bar{x}$)	148.5	70.3		
SD (s)	23.3	16.1		

$$z = \frac{x_i - \bar{x}}{s}$$

```
> height=c(128,158,177,131)
> weight=c(54,86,82,59)
> df=data.frame(height,weight)
> rownames(df) <- c("Mary", "Bob", "Julie", "mark")
> df
      height weight
Mary     128     54
Bob      158     86
Julie    177     82
mark     131     59
```

```
> scale_df <- matrix(data=NA, ncol=ncol(df), nrow=nrow(df))
> rownames(scale_df) <- rownames(df)
> colnames(scale_df) <- colnames(df)
>
> for (i in colnames(df)) {
+ mu <- mean(df[,i])
+ sdev <- sd(df[,i])
+ for (j in rownames(df)) {
+ scale_df[j,i] <- (df[j,i] - mu)/sdev
+ }
+ }
>
> scale_df
          height     weight
Mary   -0.8797392 -1.0098883
Bob     0.4076840  0.9788149
Julie   1.2230521  0.7302270
mark   -0.7509969 -0.6991535
>
> scale(df)
          height     weight
Mary   -0.8797392 -1.0098883
Bob     0.4076840  0.9788149
Julie   1.2230521  0.7302270
mark   -0.7509969 -0.6991535
```

Euclidean distance - An example

Individual	Height	Weight
Mary	128	54
Bob	158	86
Julie	177	82
Mark	131	59

```
> dist_df <- matrix(data=NA, ncol=nrow(scale_df), nrow=nrow(scale_df))
> colnames(dist_df) <- rownames(scale_df)[1:nrow(scale_df)]
> rownames(dist_df) <- rownames(scale_df)[1:nrow(scale_df)]
>
> for (i in rownames(dist_df)) {
+ for (j in colnames(dist_df)) {
+ s <- 0
+ for (k in colnames(scale_df)) {
+ delta <- (scale_df[i,k] - scale_df[j,k])^2
+ s <- s + delta
+ }
+ dist_df[i,j] <- sqrt(s)
+ }
+ }
>
> dist_df
           Mary       Bob     Julie      Mark
Mary  0.0000000 2.3690502 2.7294198 0.3363491
Bob   2.3690502 0.0000000 0.8524207 2.0391467
Julie 2.7294198 0.8524207 0.0000000 2.4372110
Mark  0.3363491 2.0391467 2.4372110 0.0000000
>
> dist(scale(df), method="euclidean")
           Mary       Bob     Julie
Bob   2.3690502
Julie 2.7294198 0.8524207
Mark  0.3363491 2.0391467 2.4372110
```

Other distance methods

Manhattan distance

Minkowski distance

Pearson’s correlation distance

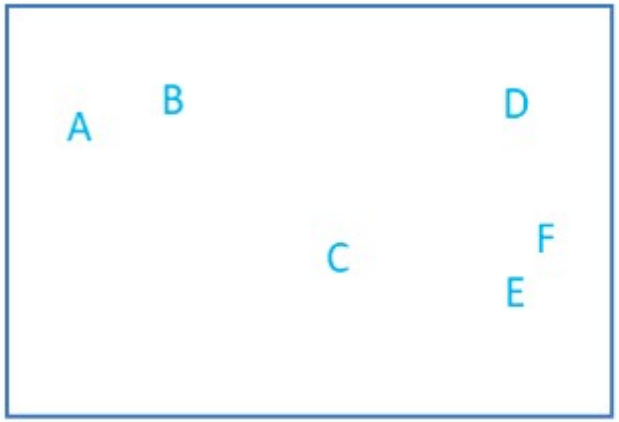
...

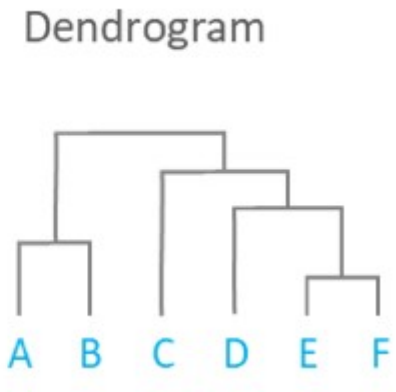
```
> dist(scale(df), method="manhattan")
           Mary       Bob     Julie
Bob   3.2761264
Julie 3.8429066 1.0639559
Mark  0.4394772 2.8366492 3.4034294

> dist(scale(df), method="minkowski")
           Mary       Bob     Julie
Bob   2.3690502
Julie 2.7294198 0.8524207
Mark  0.3363491 2.0391467 2.4372110
```

Dendrogram

Definition: a tree diagram, especially one showing taxonomic relationships

A prefered way to represent a hierarchical clustering





Dendrogram - Example dataset

Raw data

ID	X, Y
A	18, 0
B	22, 0
C	43, 0
D	42, 0
E	27, 0
F	25, 0

Euclidean distance matrix data

ID	A	B	C	D	E	F
A	0	4	25	24	9	7
B		0	21	20	5	3
C			0	1	16	18
D				0	15	17
E					0	2
F						0

Dendrogram - Example dataset

ID	A	B	C	D	E	F
A	0	4	25	24	9	7
B		0	21	20	5	3
C			0	1	16	18
D				0	15	17
E					0	2
F						0

The shortest distance is 1 (C and D). Cluster 1 is CD

ID	A	B	CD	E	F
A	0				
B	4	0			
CD	25	21	0		
E	9	5	16	0	
F	7	3	18	2	0

Dendrogram - Example dataset

ID	A	B	CD	E	F
A	0				
B	4	0			
CD	25	21	0		
E	9	5	16	0	
F	7	3	18	2	0

The shortest distance is 2 (EF). Cluster 2 is EF

ID	A	B	CD	EF
A	0			
B	4	0		
CD	25	21	0	
EF	9	5	18	0

Dendrogram - Example dataset

ID	A	B	CD	EF
A	0			
B	4	0		
CD	25	21	0	
EF	9	5	18	0

The shortest distance is 4 (AB). Cluster 3 is AB

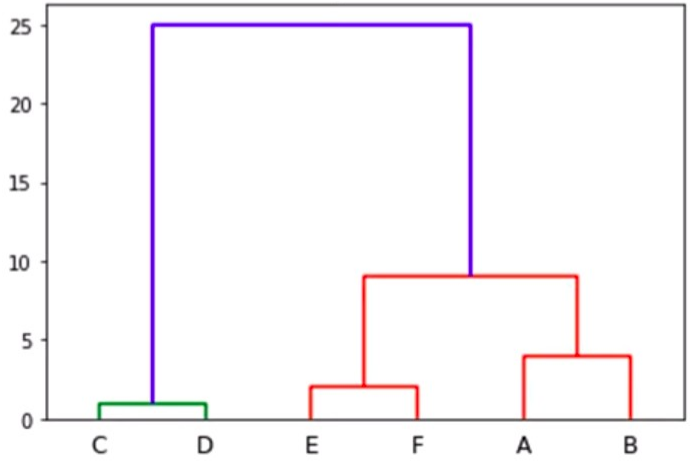
ID	AB	CD	EF
AB	0		
CD	25	0	
EF	9	18	0

Dendrogram - Example dataset

ID	AB	CD	EF
AB	0		
CD	25	0	
EF	9	18	0

The shortest distance is 9 (AB EF). Cluster 5 is ABvEF

ID	ABEF	CD
ABEF	0	
CD	25	0



Hierarchical clustering

Hierarchical clustering

-Hierarchical clustering seeks to build a hierarchy of clusters based on a proximity measure

-Strategies:

(1) Divisive (top down) clustering: start with all data points in one cluster, the root, then

+ splits the root into a set of child clusters. Each child cluster is recursively divided further

+ stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

(2) Agglomerative (bottom up) clustering: the dendrogram is built from the bottom by

+ merging the most similar (or nearest) pair of clusters

+ stopping when all the data points are merged into a single cluster (i.e., the root cluster)

Linkage

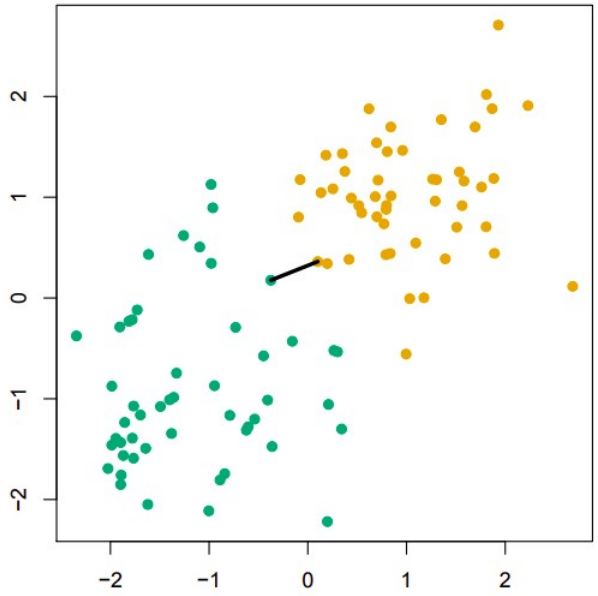
- Let $d_{ij} = d(x_i, x_j)$ denote the dissimilarity[1] (distance) between observation x_i and x_j
- At our first step, each cluster is a single point, so we start by merging the two observations that have the *lowest dissimilarity*
- But after that…we need to think about distances not between points, but between sets (clusters)
- The dissimilarity between two clusters is called the linkage
- i.e., Given two sets of points, G and H, a linkage is a dissimilarity measure $d(G, H)$ telling us how different the points in these sets are

Single linkage

In single linkage (i.e., nearest-neighbor linkage), the dissimilarity between G, H is the smallest dissimilarity between two points in different groups:

$$d_{\text{single}}(G, H) = \min_{i \in G,\, j \in H} d(x_i, x_j)$$

Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{\text{single}}(G, H)$ is the distance of the closest pair

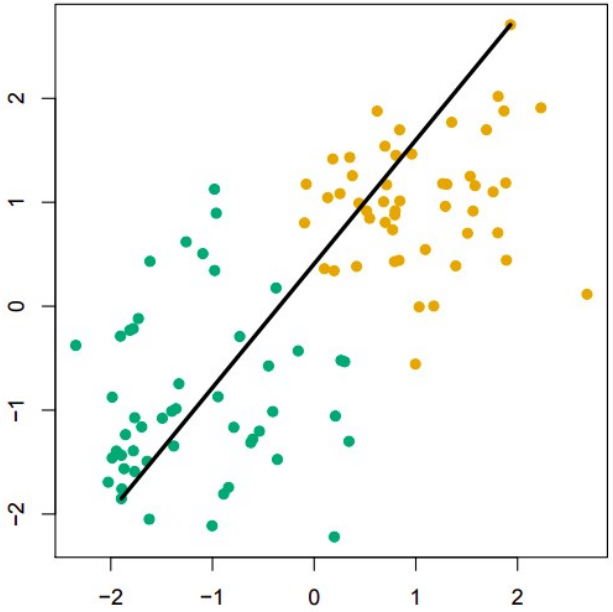


Complete linkage

In complete linkage (i.e., furthest-neighbor linkage), dissimilarity between G, H is the largest dissimilarity between two points in different groups:

$$d_{\text{complete}}(G, H) = \max_{i \in G,\, j \in H} d(x_i, x_j)$$

Example (dissimilarities d_{ij} are distances, groups are marked by colors): complete linkage score $d_{\text{complete}}(G, H)$ is the distance of the furthest pair



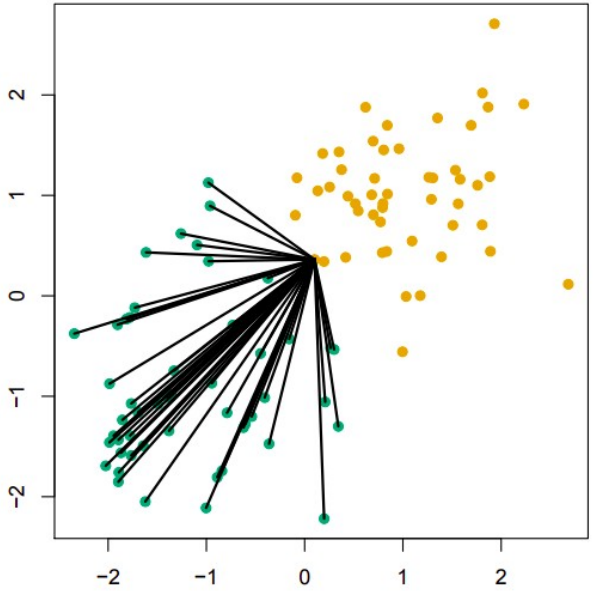
Average linkage

In average linkage, the dissimilarity between G, H is the average dissimilarity over all points in opposite groups:

$$d_{\text{average}}(G, H) = \frac{1}{|G| \cdot |H|} \sum_{i \in G,\, j \in H} d(x_i, x_j)$$

Example (dissimilarities d_{ij} are distances, groups are marked by colors): average linkage score $d_{\text{average}}(G, H)$ is the average distance across all pairs

(Plot here only shows distances between the green points and one orange point)



Linkage - other methods

What type of algorithm should be used to cluster points and define groups

" ward.D" = Ward's minimum variance method

" ward.D2" = Ward's minimum variance method – however dissimilarities are squared before clustering

"single" = Nearest neighbours method

"complete" = distance between two clusters is defined as the maximum distance between an observation in one cluster and an observation in the other cluster

"average" = distance between two clusters is defined as the mean distance between an observation in one cluster and an observation in the other cluster

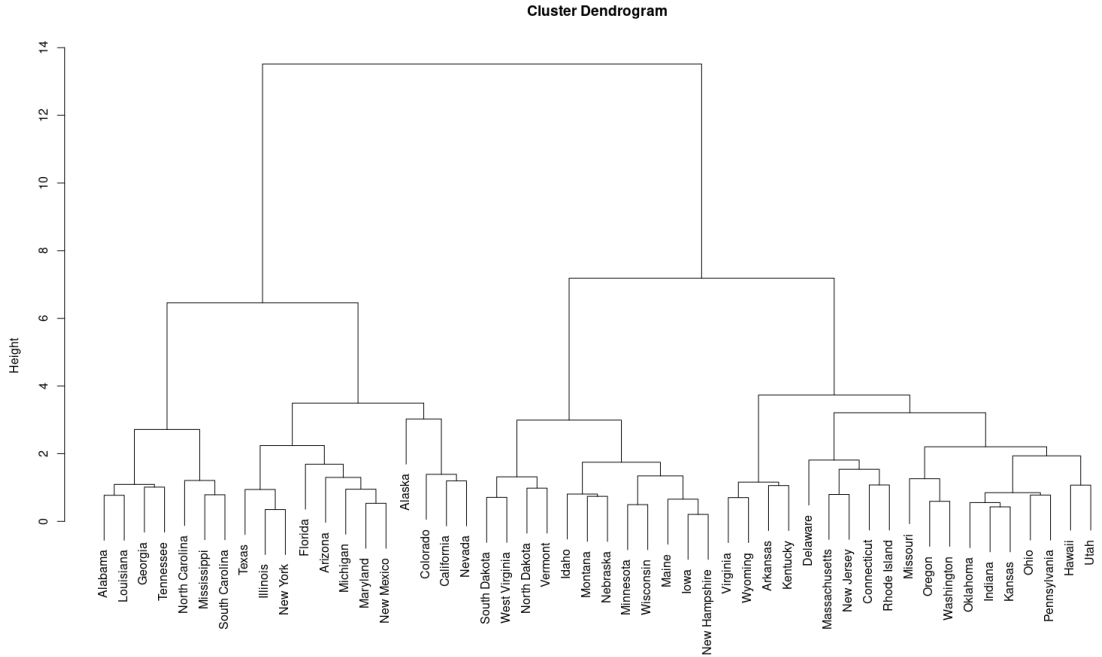
"mcquitty " = when two clusters are be joined, the distance of the new cluster to any other cluster is calculated as the average of the distances of the soon to be joined clusters to that other cluster

"median" = uses group median

"centroid" = uses group centroid

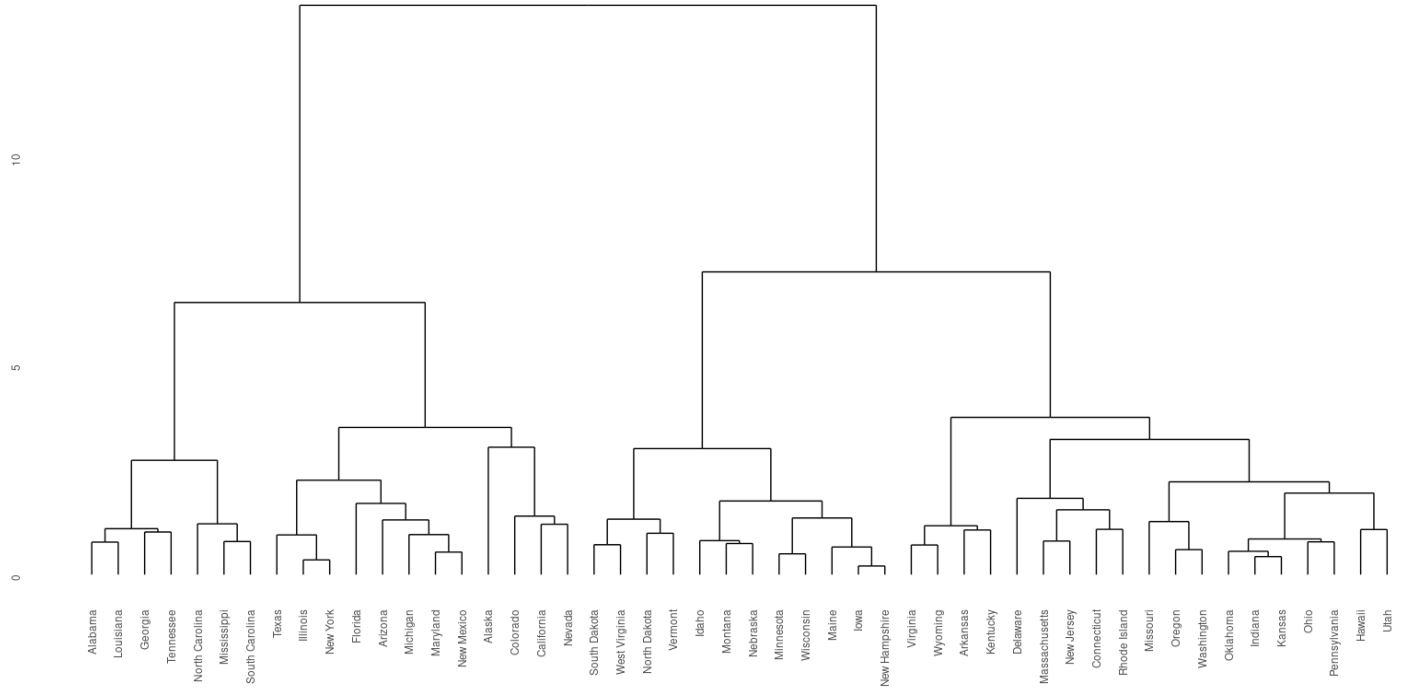
Hierarchical clustering - Example

```
dist <- dist(scale(USArrests), method="euclidean")
hc <- hclust(dist, method="ward.D2")
plot(hc)
```



Hierarchical clustering - Example

```
dist <- dist(scale(USArrests), method="euclidean")
hc <- hclust(dist, method="ward.D2")
ggdendrogram(hc)
```



Clustering Performance Evaluation Metrics

- No any labels in clustering
- The goal is to create clusters that have similar observations clubbed together and dissimilar observations kept as far as possible.
- Based on some similarity or dissimilarity measure such as the distance between cluster points
- If the clustering algorithm separates dissimilar observations apart and similar observations together, then it has performed well.
- The two most popular evaluation metrics: Silhouette coefficient and **Dunn's Index**

https://www.analyticsvidhya.com/blog/2020/10/quick-guide-to-evaluation-metrics-for-supervised-and-unsupervised-machine-learning/#:~:text=Clustering%20Performance%20Evaluation%20Metrics&text=Here%20clusters%20are%20evaluated%20based,then%20it%20has%20performed%20well

Dunn Index

Dunn's Index is equal to the minimum inter-cluster distance divided by the maximum cluster size.

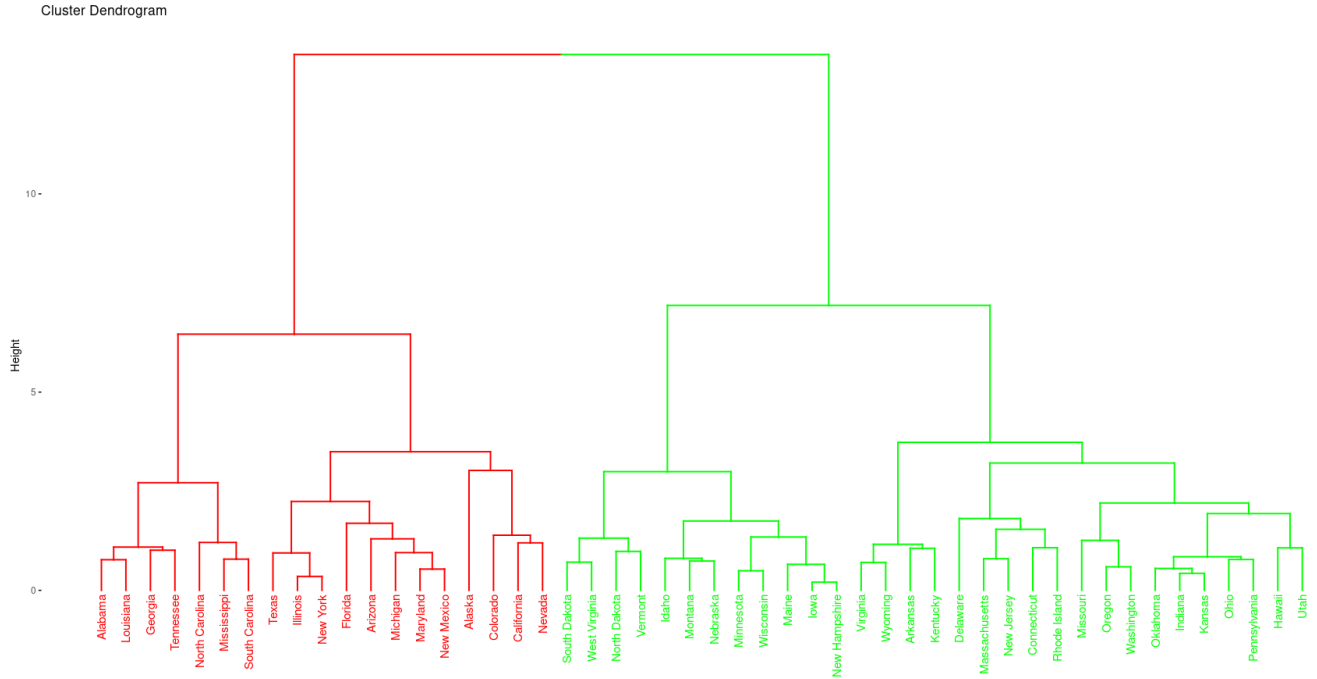
Note that large inter-cluster distances (better separation) and smaller cluster sizes (more compact clusters) lead to a higher DI value → higher DI implies better clustering

```
> for (i in 2:10) {
+ cluster <- cutree(hc, i)
+ DI <- dunn(dist, cluster)
+ print(paste0("Dunn index for ", i, " cluster is: ", DI))
+ }
[1] "Dunn index for 2 cluster is: 0.263664639274641"
[1] "Dunn index for 3 cluster is: 0.119434484503072"
[1] "Dunn index for 4 cluster is: 0.160440346650323"
[1] "Dunn index for 5 cluster is: 0.160440346650323"
[1] "Dunn index for 6 cluster is: 0.175237157740891"
[1] "Dunn index for 7 cluster is: 0.175237157740891"
[1] "Dunn index for 8 cluster is: 0.21977971625531"
[1] "Dunn index for 9 cluster is: 0.225846863303564"
[1] "Dunn index for 10 cluster is: 0.241799928809642"
```

https://www.analyticsvidhya.com/blog/2020/10/quick-guide-to-evaluation-metrics-for-supervised-and-unsupervised-machine-learning/#:~:text=Clustering%20Performance%20Evaluation%20Metrics&text=Here%20clusters%20are%20evaluated%20based,then%20it%20has%20performed%20well

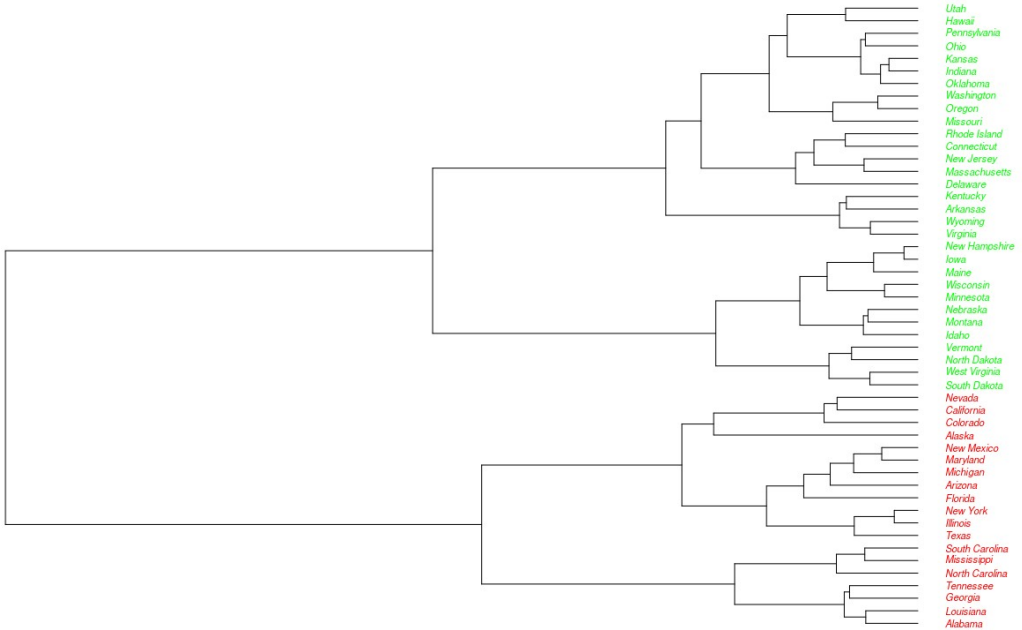
Hierarchical clustering - Example

```
dist <- dist(scale(USArrests), method="euclidean")
hc <- hclust(dist, method="ward.D2")
library(factoextra)
fviz_dend(hc, k=2, k_color=c("red", "green"))
```



Hierarchical clustering - Example

```
dist <- dist(scale(USArrests), method="euclidean")
hc <- hclust(dist, method="ward.D2")
nclust <- cutree(hc,2)
colors <- c("red", "green")
plot(as.phylo(hc), label.offset=0.5, cex=0.7, tip.color=colors[nclust], type="phylogram")
```



Strengths and Weaknesses

Strengths:

The math of hierarchical clustering is the easiest to understand compared to other clustering algorithms.

It is also relatively straightforward to program.

Its main output, the dendrogram, is also the most appealing of the outputs of these algorithms.

Weaknesses:

When using hierarchical clustering it is necessary to specify both the distance metric and the linkage criteria. There is rarely any strong theoretical basis for such decisions. Thus, it rarely provides the best solution.

K-mean clustering

K-means: Idea

- Represent the data set in terms of K clusters, each of which is summarized by a prototype μ_k
- Each data is assigned to one of K clusters
  - Represented by responsibilities $r_{ik} \in \{0, 1\}$ such that $\sum_{k=1}^{K} r_{ik} = 1$ for all data indices *i*
- Example: 4 data points and 3 clusters

$$(r_{ik}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

K-means: Idea

- Loss function: the sum-of-squared distances from each data point to its assigned prototype (is equivalent to the within-cluster scatter).

data

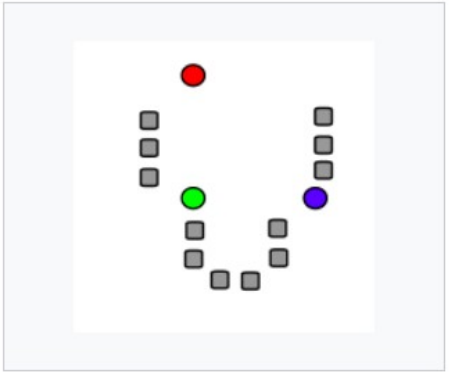
$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|x_i - \mu_k\|^2$$

responsibilities

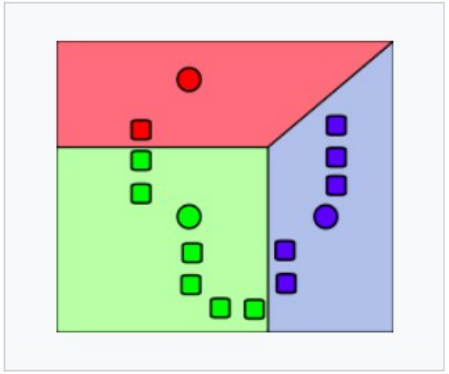
prototypes (centroids/ means)

K-means clustering algorithm

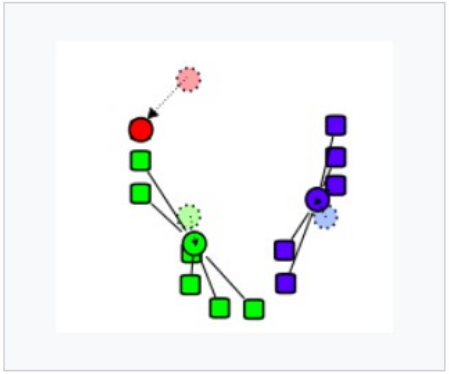
Demonstration of the standard algorithm



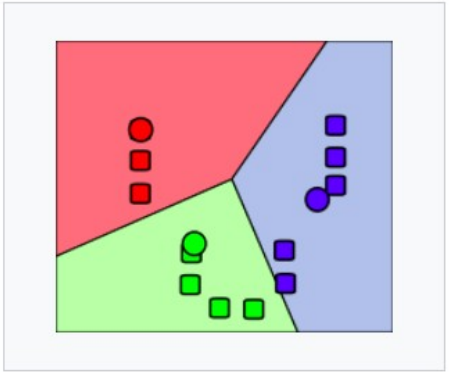
1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3. The centroid of each of the *k* clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

K-means clustering algorithm

State the number of clusters (k)

Randomly select k objects from the dataset as initial centers for clusters

Assign each observation to their closest centroid based on distance

For each of the k clusters, compute the new mean value ("centroid update")

Iteratively minimize the total within sum of squares

How many clusters (k)?

No definitive solution (solution may be subjective)

More than 30 methods

Common methods:

- Elbow method
- Average silhouette method
- Gap statistic method

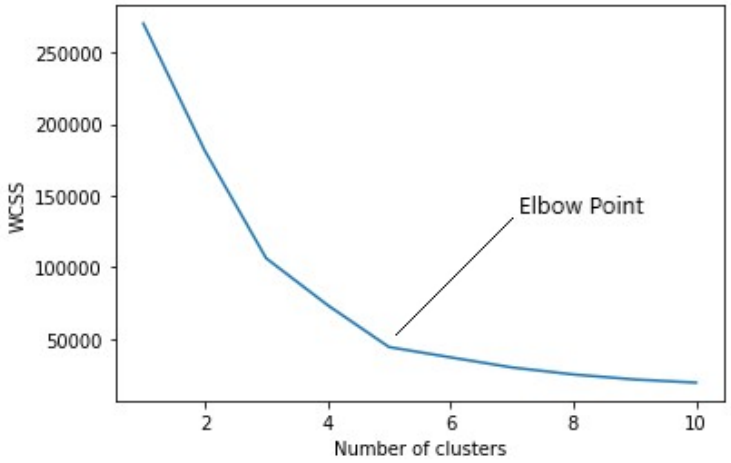
Elbow method

Elbow method Criterion : within cluster sum of squares (WCSS, loss function)

Select k so that adding another cluster doesn't increase the total WCSS significantly.

Algorithms:

- Using k-means algorithm
- For each k, calculate WCSS
- The point of location of a bend (knee) in the plot is considered an optimal k

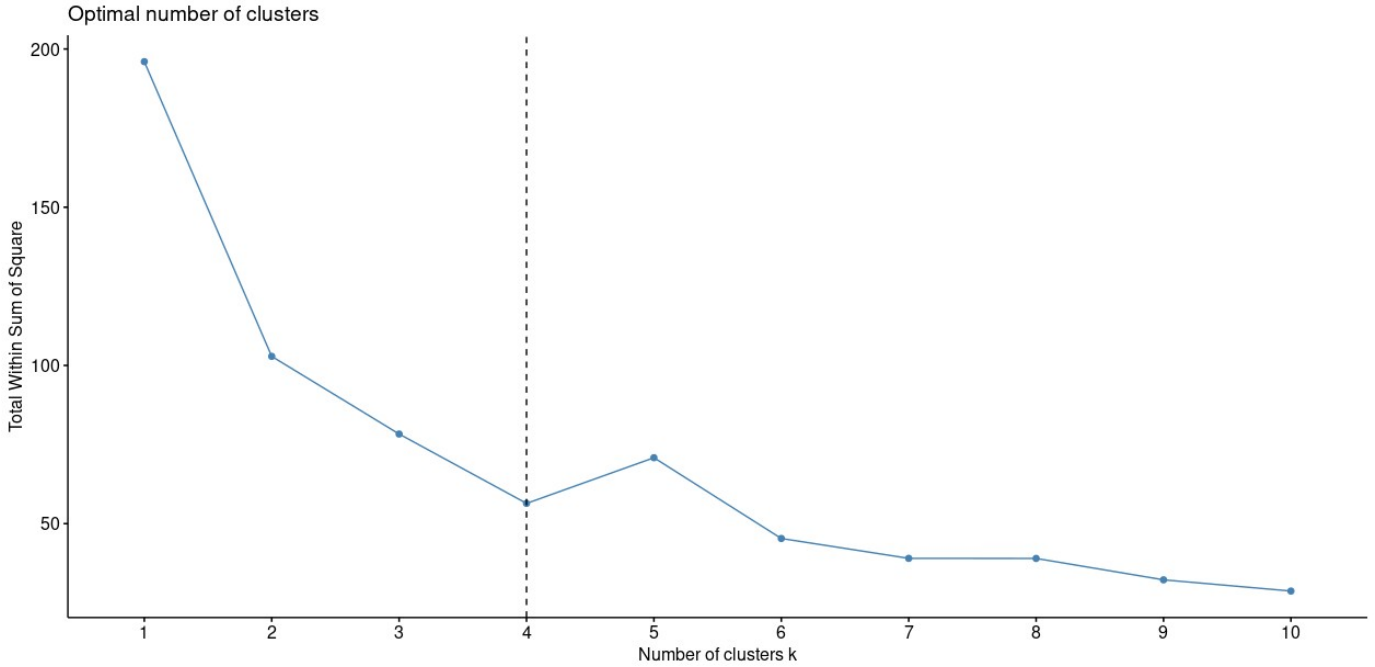


Elbow method

```
dt = scale(USArrests)
head(dt)
               Murder   Assault   UrbanPop         Rape
Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473
Alaska     0.50786248 1.1068225 -1.2117642  2.484202941
Arizona    0.07163341 1.4788032  0.9989801  1.042878388
Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602
California 0.27826823 1.2628144  1.7589234  2.067820292
Colorado   0.02571456 0.3988593  0.8608085  1.864967207
```

Elbow method

```
> library(NbClust); library(factoextra)
> fviz_nbclust(dt, kmeans, method="wss") + geom_vline(xintercept=4, linetype=2)
```



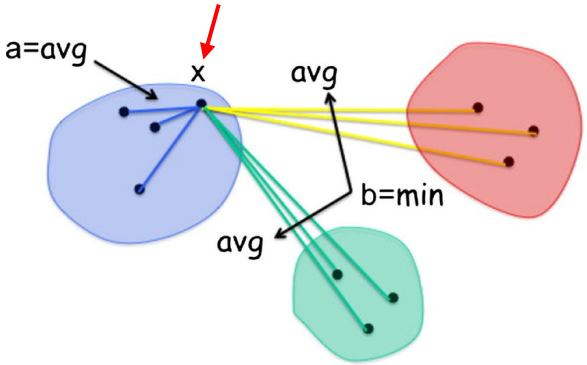
Silhouette method

Silhouette : "a measure of how similar an object is to its own cluster"

Optimal k is the one that maximize the average silhouette over a range of possible values for k (Kaufman and Rousseeuw 1990).

Algorithm:

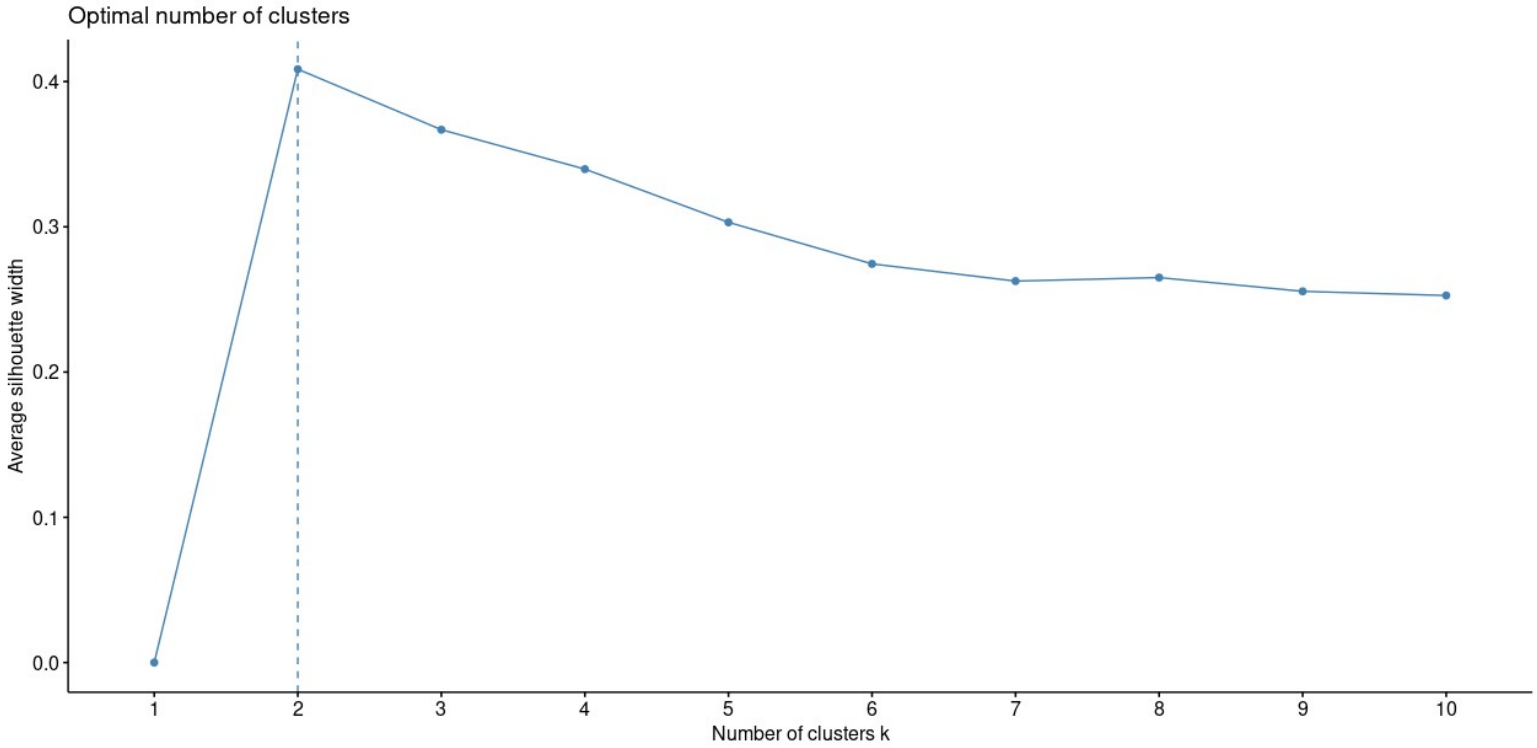
- Using k-means algorithm
- For each k, calculate silhouette score (s)
- Plot of k vs silhouette score (s)
- The point of maximum s is considered the optimal k



$$s = \frac{b - a}{\max(a, b)}$$

Silhouette method

```
fviz_nbclust(dt, kmeans, method="silhouette")
```



Gap statistic method

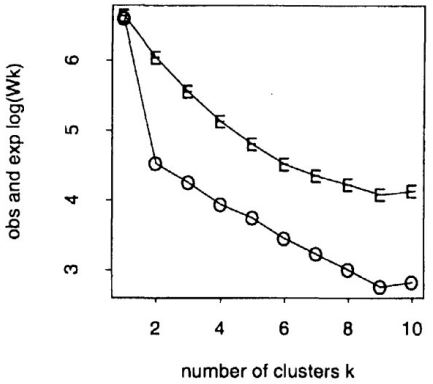
Gap statistic: Tibshirani, Walther, Hastie (2001)

Idea: standardize the graph of $\log(W_k)$ by comparing it with its expectation under an appropriate null reference distribution of the data

The estimate of the optimal number of clusters is the value of *k* for which $\log(W_k)$ falls the farthest below the reference curve. Hence we define:

$$\mathrm{Gap}_n(k) = E_n^*\{\log(W_k)\} - \log(W_k)$$

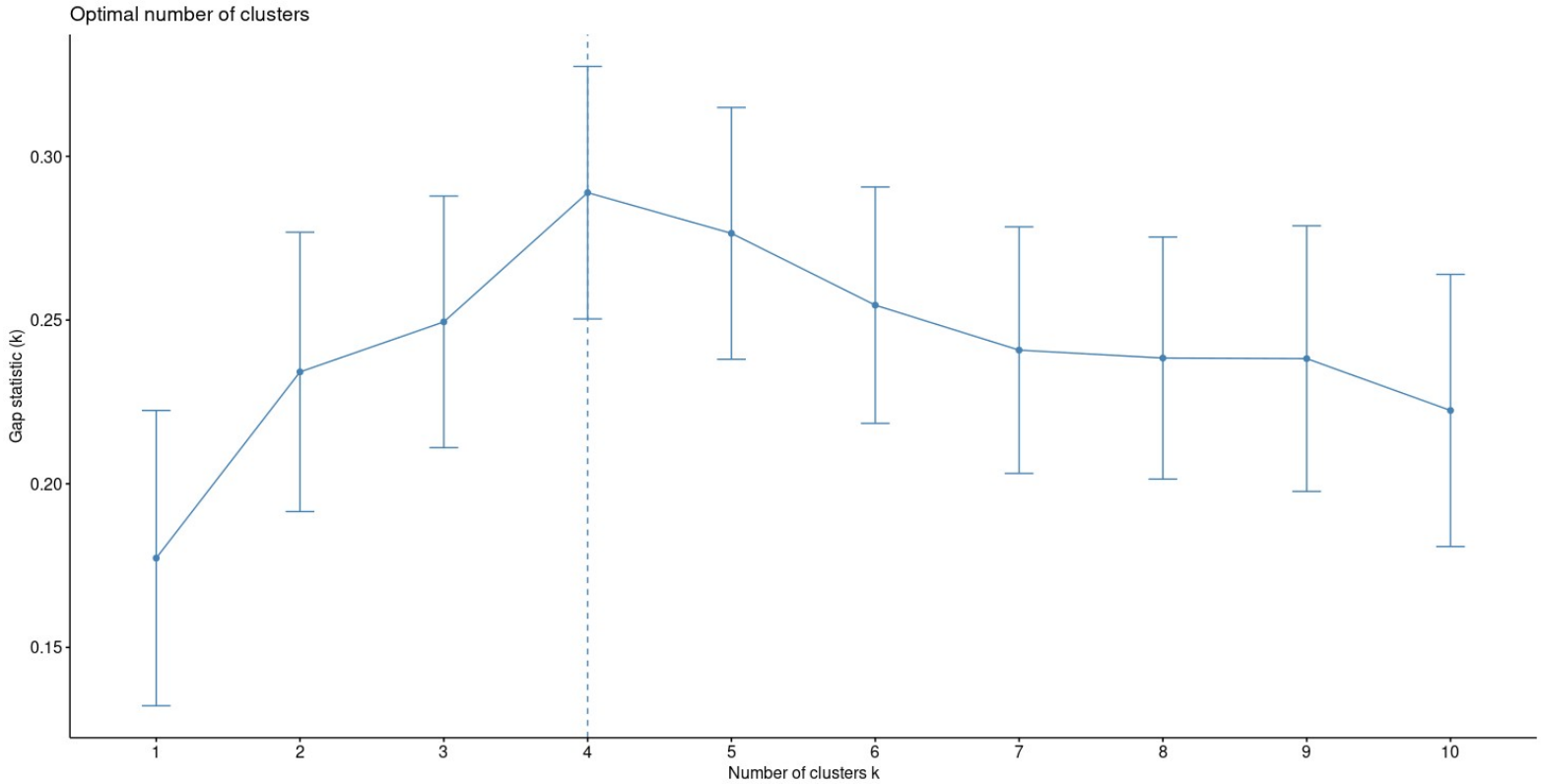
In other words, the estimate k will be the value that maximize $\mathrm{Gap}_n(k)$ after we take the sampling distribution into account



https://hastie.su.domains/Papers/gap.pdf

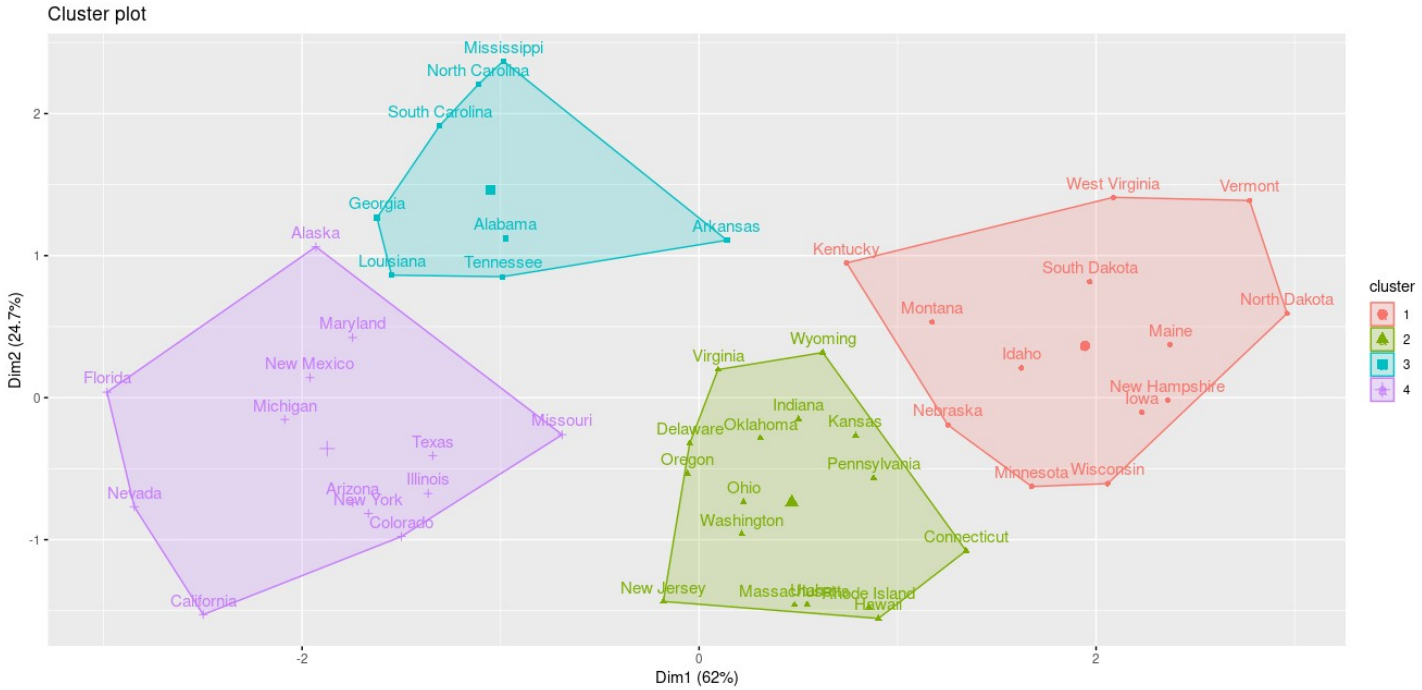
Gap statistic method

```
fviz_nbclust(dt, kmeans, nstart=10, method="gap_stat", nboot=30)
```



Visualization of k-mean clusters

```
km = kmeans(dt, center=4, nstart=30)
fviz_cluster(km, dt, ellipse.type="convex")
```



Visualization of k-mean clusters

```
km = kmeans(dt, center=2, nstart=30)
fviz_cluster(km, dt, ellipse.type="convex")
```

Hierarchical and k-mean clustering - Summary

In **k-means clustering**, we seek to partition the observations into a pre-specified number of clusters.

In **hierarchical clustering**, we do not know in advance how many clusters we want; in fact, we end up with a tree-like visual representation of the observations, called a dendrogram, that allows us to view at once the clusterings obtained for each possible number of clusters, from 1 to n.

Evaluation: based on some similarity or dissimilarity measure such as the distance between cluster points.

THE END