



**BeVera**

# CDC Data Science R Academy

---

July 2021

Yvonne Phillips, Instructor

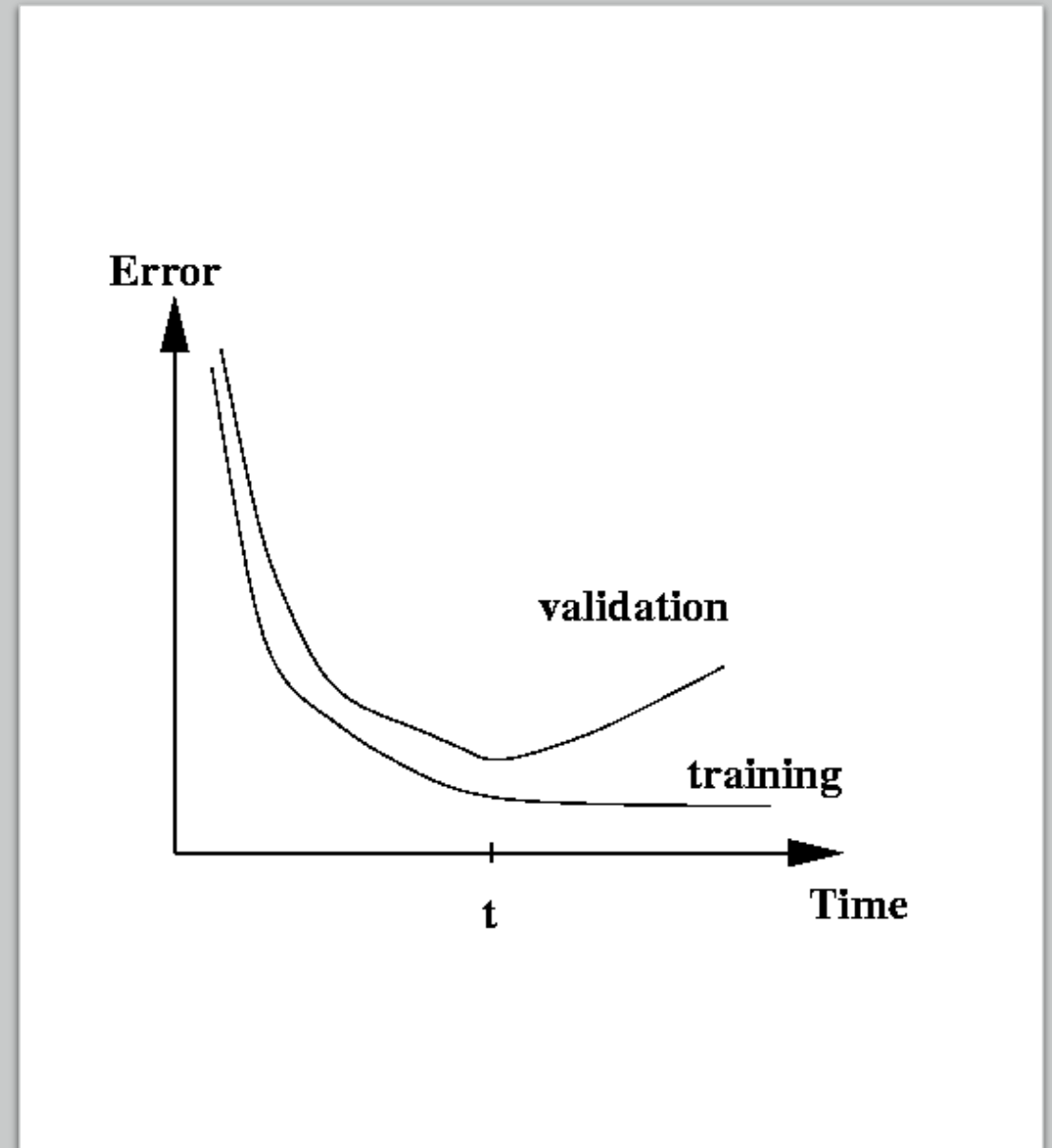
# You've prepared your data: what's next?



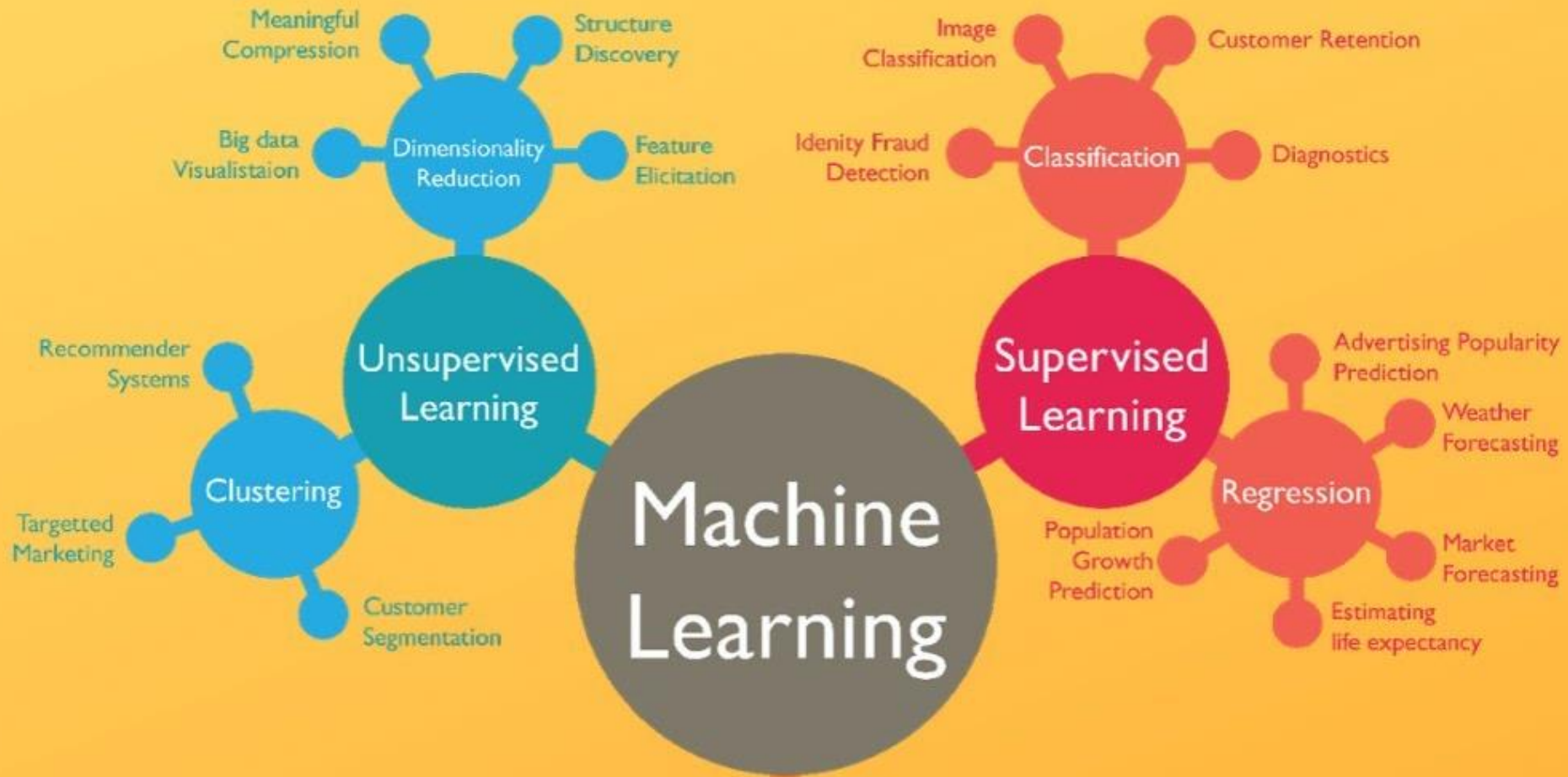
What kind of analysis do you need? Which model is more appropriate for it? ...

# Datasets

- **Training set:** a set of examples used for learning, where the target value is known.
- **Validation set:** a set of examples used to tune the architecture of a classifier and estimate the error.
- **Test set:** used only to assess the performances of a classifier. It is never used during the training process so that the error on the test set provides an unbiased estimate of the generalization error.



# Supervised Vs Unsupervised



# Supervised Learning

- Training data includes both the input and the desired results.
- For some examples the correct results (targets) are known and are given in input to the model during the learning process.
- The construction of a proper training, validation and test set is crucial.
- These methods are usually fast and accurate.
- Have to be able to **generalize**: give the correct results when new data are given in input without knowing a priori the target.

# Unsupervised Learning

- The model is not provided with the correct results during the training.
- Can be used to cluster the input data in classes on the basis of their statistical properties only.
- Cluster significance and labeling.
- The labeling can be carried out even if the labels are only available for a small number of objects representative of the desired classes

# Theory and Methods

- Examine analytic needs and select an appropriate technique based on business objective initial hypothesis; and the data's structure and volume
- Apply some of the more commonly used methods in Analytics solutions
- Explain the algorithms and the technical foundations for the commonly used methods
- Explain the environment (use case) in which each technique can provide the most value
- Use appropriate diagnostic methods to validate the models created
- Use R to fit, score and evaluate models

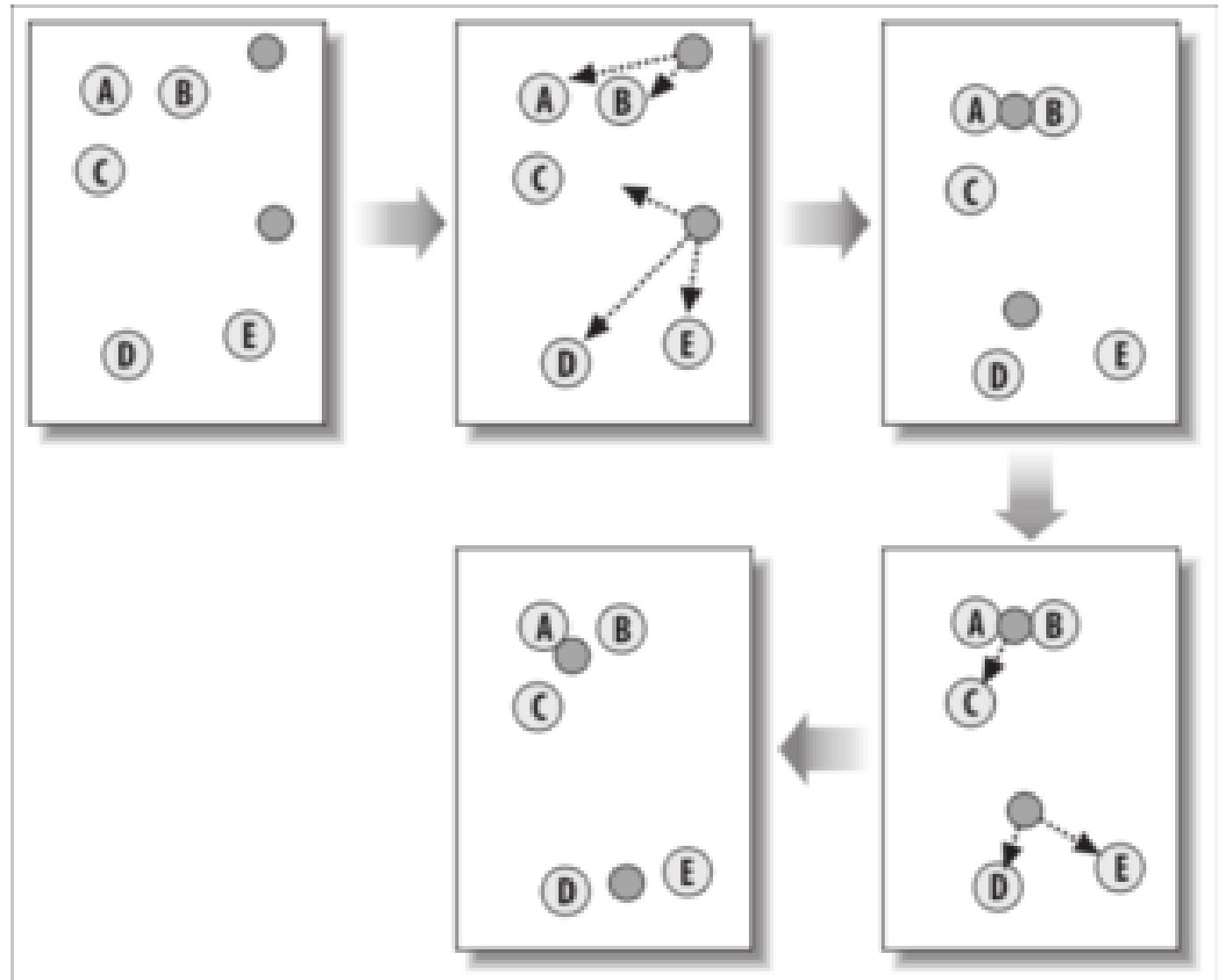
# What kind of problem do I need to solve? How do I solve it?

The Problem to Solve	The Category of Techniques	Analytical Method
I want to group items by similarity. I want to find structure (commonalities) in the data.	Clustering	K-means clustering
I want to discover relationships between actions or items	Association Rules	Apriori
I want to determine the relationships, between the outcome and the input variables.	Regression	Linear Regression Logistic Regression
I want to assign (known) labels to objects.	Classification	Naïve Bayes Decision Trees
I want to find the structure in a temporal process. I want to forecast the behavior of a temporal process.	Time Series Analysis	ACF, PACF, ARIMA
I want to analyze my text data.	Text Analysis	Regular expressions, Document representation (Bag of Words), TF-IDF



# Cluster Analysis

CDC Data Science R Academy  
July 2021



This machine learning course using the R programming language will get you started with hands-on examples. The course is ideal for professionals who need to use cluster analysis, unsupervised machine learning and R in their field.

In this course, you will :

- Understand unsupervised learning and clustering using R-programming language.
- How cluster algorithms works in general. Get an intuitive explanation with graphics that are easy to understand.
- Course covers both theoretical background of K-means clustering analysis as well as practical examples in R-Studio.
- How to implement K-Means and Hierarchical clustering very fast with R coding: examples of real data will be provided.
- Evaluate Model Performance & Learn The Best Practices For Evaluating Machine Learning Model Accuracy



# Agenda

- What is clustering analysis?
- Partitioning clustering
  - K-means
  - K-medoids (PAM)
  - Evaluation Methods
- Implementing Partitioning clustering in R
- Hierarchical clustering
  - Dendrogram
  - Agglomerative clustering
  - Divisive clustering
- Clustering linkage comparison
- Implementing hierarchical clustering in R programming language
- Dimensionality Reduction - Review PCA in R

# Background

Clustering is an **unsupervised learning** problem.

In **some sense** the **objective** is **similar** to **classification** BUT **we do not have a training set.**

**Objective:** a collection of objects which are *similar* between them and are dissimilar to the objects belonging to another cluster.

What does similar mean?

Clustering is an **ill-posed problem** under specific axiomatic framework

Consequence: different algorithms provide different clustering

## When to cluster?

In which of these scenarios would clustering methods likely be appropriate?

- 1) Using consumer behavior data to identify distinct segments within a market.
- 2) Predicting whether a given user will click on a pharmacy ad.
- 3) Identifying distinct groups of medications that follow similar healing patterns.
- 4) Modeling & predicting exponential growth.

- a. 1
- b. 2
- c. 4
- d. 1 & 3
- e. 2 & 4

# Application of clustering analysis

- Medical Science – Medicine and health industries make use of clustering algorithms to *facilitate efficient diagnosis and treatment of their patients as well as the discovery of new medicines*. Based on the age, group, genetic coding of the patients, these organizations are better capable to understand diagnosis through robust clustering.
- Sociology – Clustering is used in Data Mining operations to *divide people based on their demographics, lifestyle, socioeconomic status*, etc. This can help the law enforcement agencies to group potential criminals and even identify them with an efficient implementation of the clustering algorithm.
- Clustering in medicine is known as *nosology*.

# Motivations

Does replications cluster together?

Does similar conditions, time points, tissue types cluster together?

Cluster genes -> Prediction of functions of unknown genes by known ones

Cluster samples -> Discover clinical characteristics (e.g. survival, marker status) shared by samples

# Clustering Algorithms

## Partitioning-based

K-means

PAM

CLARA

FCM

## Hierarchical-based

Agglomerative / Divisive

BIRCH

ROCK

CURE

## Density-based

DBSCAN

OPTICS

DBCLASD

DENCLUE

## Grid-based

Wavecluster

STING

CLIQUE

OptiGrid

## Model-based

GMM

COBWEB

CLASSIT

SOMs



Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large <code>n_samples</code> , medium <code>n_clusters</code> with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with <code>n_samples</code>	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with <code>n_samples</code>	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium <code>n_samples</code> , small <code>n_clusters</code>	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large <code>n_samples</code> and <code>n_clusters</code>	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large <code>n_samples</code> and <code>n_clusters</code>	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large <code>n_samples</code> , medium <code>n_clusters</code>	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large <code>n_clusters</code> and <code>n_samples</code>	Large dataset, outlier removal, data reduction.	Euclidean distance between points

See more: <http://scikit-learn.org/stable/modules/clustering.html>

# What is Good Clustering?



A good clustering method will produce high quality clusters.

High intra-class similarity: cohesive within clusters  
Low intra-class similarity: distinctive within clusters



The quality of a clustering method depends on

The similarity measure used by the method  
Its implementation and  
Its ability to discover some or all the hidden patterns.

# Measure the quality of clustering

Dissimilarity/Similarity metric

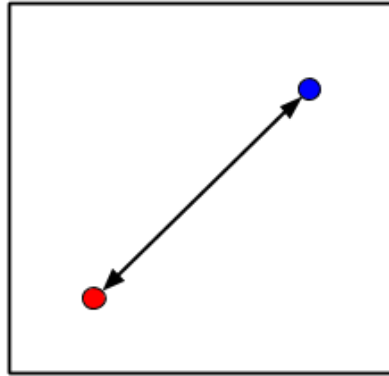
Distance functions

Quality of clustering:

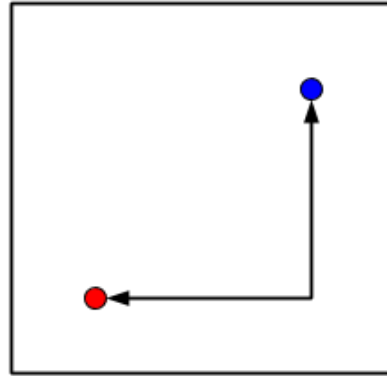
Hard to define “similar enough” or “good enough”

## Methods in measuring distances: distance matrix

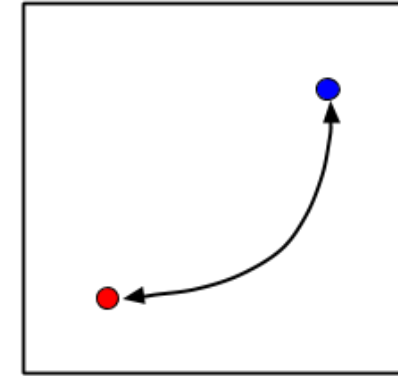
Euclidean



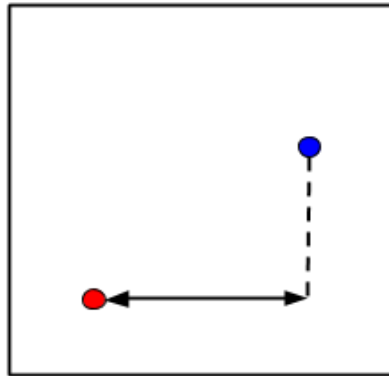
Manhattan



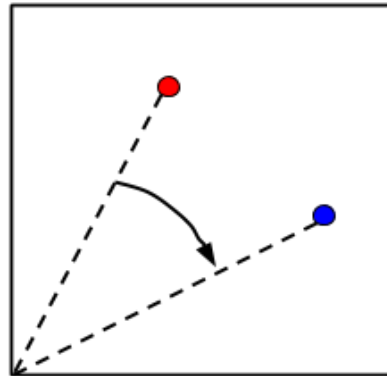
Minkowski



Chebychev



Cosine Similarity



Hamming



# Distance vs Similarity

Distance = 1 – Similarity

<u>Measure</u>	<u>method=""</u>	<u>Best used for...</u>
Euclidean distance	Euclidean	continuous data
Hamming distance	Hamming	categorical data between the two vectors
Canberra distance	Canberra	non-negative values (e.g., counts)
Jaccard asymmetric	Binary	binary data

# Measuring distance for categorical data: Binary data

Var1	Var2	Var3	Var4
TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	TRUE	FALSE

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

```
Jaccard = function (x, y) {  
  M.11 = sum(x == 1 & y == 1)  
  M.10 = sum(x == 1 & y == 0)  
  M.01 = sum(x == 0 & y == 1)  
  return (M.11 / (M.11 + M.10 + M.01))  
}
```

# Partitioning Clustering



K-means

PAM

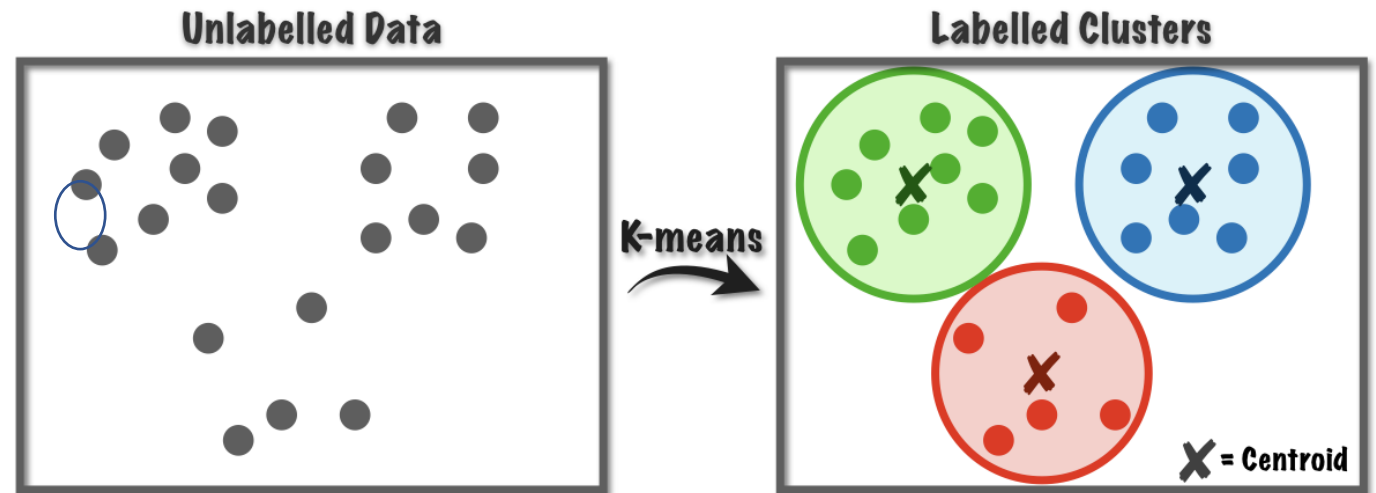
CLARA

# K-means

- k-means clustering objective is to separate the set of observations  $\mathbf{x}_i$  to  $k$  groups such that the within-cluster sum of squares is minimized

$$\min_S \sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|^2$$

The centroid of the points assigned to cluster  $S_i$





# K-means algorithms

- Lloyd algorithm (also called Lloyd-Forgy algorithm)
- MacQueen algorithm
- Hartigan-Wong algorithm

# K-means

- Lloyd's heuristic algorithm includes three steps:

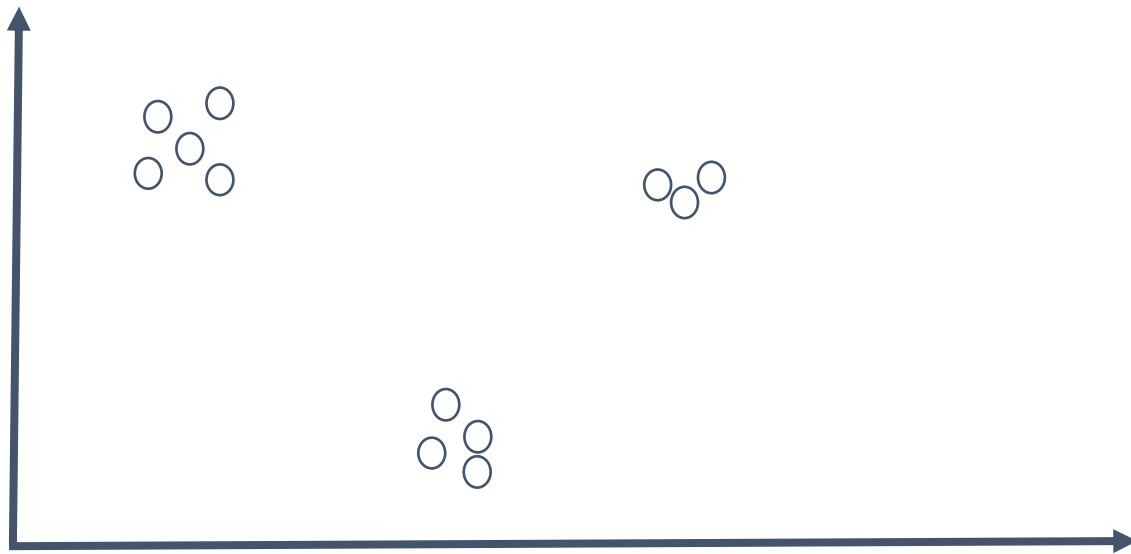
- Initialization
  - Assignment
  - Update
- } Iterative steps

- In the **initialization** phase identifies an initial set of  $k$  cluster centroids
  - Usually this is simply a randomly selected set of  $k$  data points from our observations

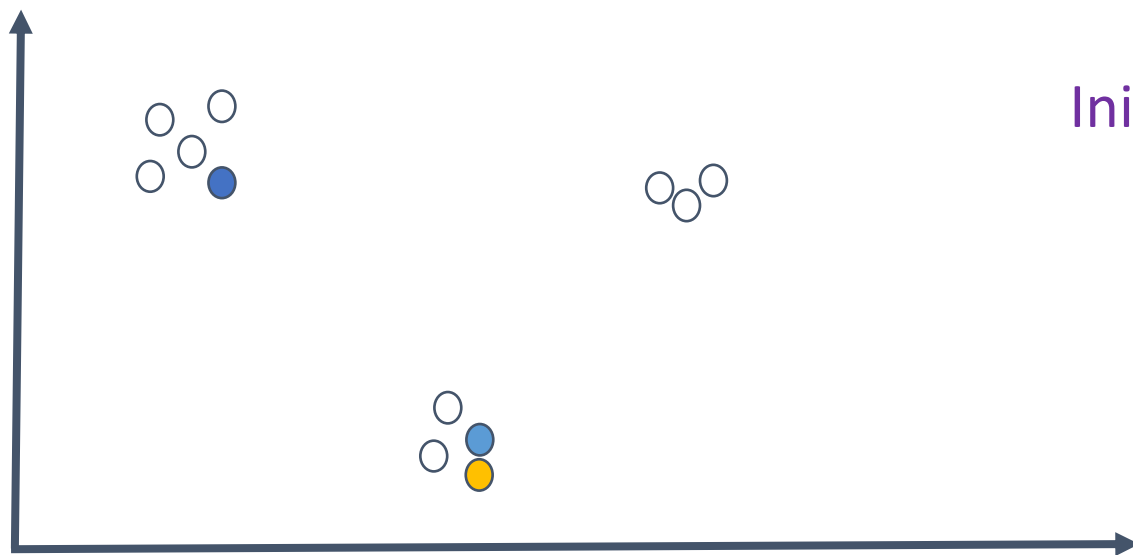
# K-means

- In the **assignment phase**, given the  $k$  centers  $(\mu_i)$  we assign each of the observations to one of the  $k$ -clusters
  - The assignment rule of point  $\mathbf{x}_j$  is simple:  $\mathcal{S}_{x_j}(t) = \{\mathcal{S}_r: \|\mathbf{x}_j - \mu_r\| \leq \|\mathbf{x}_j - \mu_r\|, \forall r \in \{1, \dots, k\}\}$ 
    - Simply put, the point  $\mathbf{x}_j$  is assigned during the  $t$ -th iteration to the cluster whose centroid  $(\mu_i)$  is the closest
- In the **update phase**, we re-calculate the new centroids of the  $k$ -clusters based on the assignments:  $\mu_i(t + 1) = \frac{1}{|\mathcal{S}_i|} \sum_{\mathbf{x}_j \in \mathcal{S}_i} \mathbf{x}_j$

# K-means

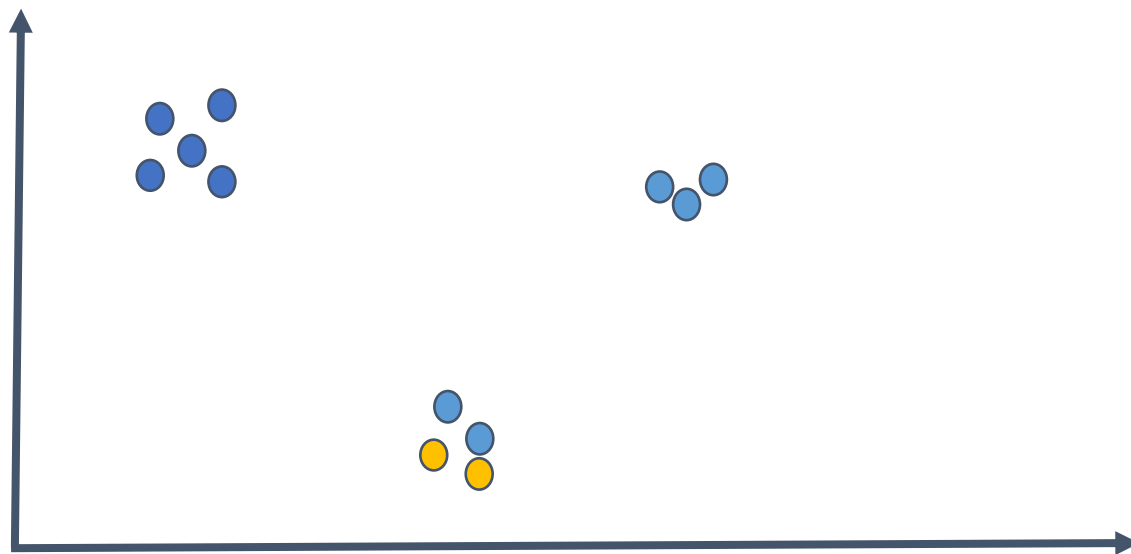


# K-means



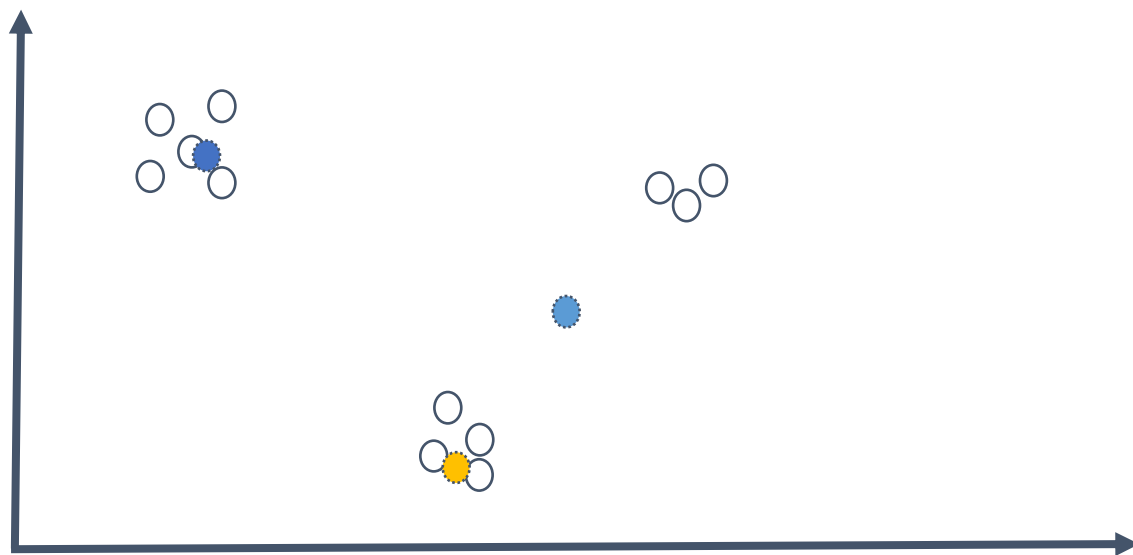
Initialization phase ( $k=3$ )

# K-means



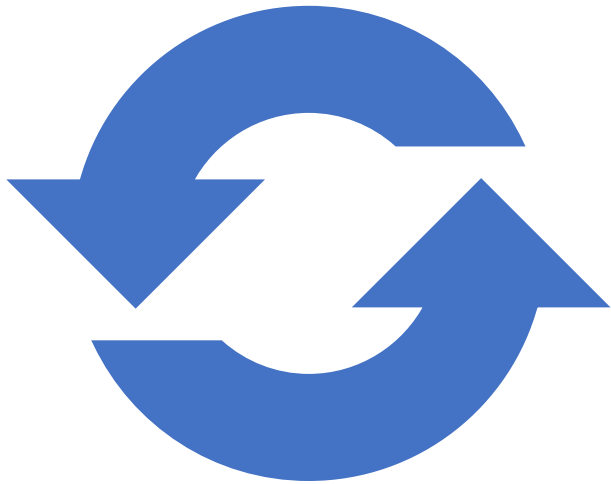
Assignment phase

# K-means



# K-means

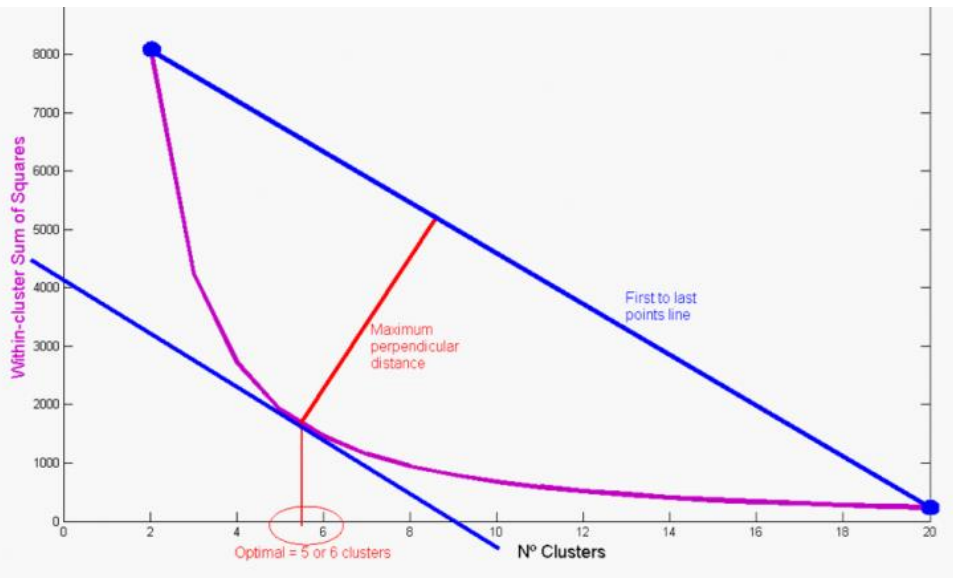
---



- When do we stop the iterations over the **assignment** and **update** phases?
  - When there are no changes between two consecutive assignment phases
- **There is no guarantee that the optimal solution has been found!!**
- **Different initializations can result into different clusters!**
- How to choose the value of  $k$ ?
  - Sometimes  $k$  is application-specific but most of the times there is not a good idea on what is a good value for  $k$



# Optimal number of clusters



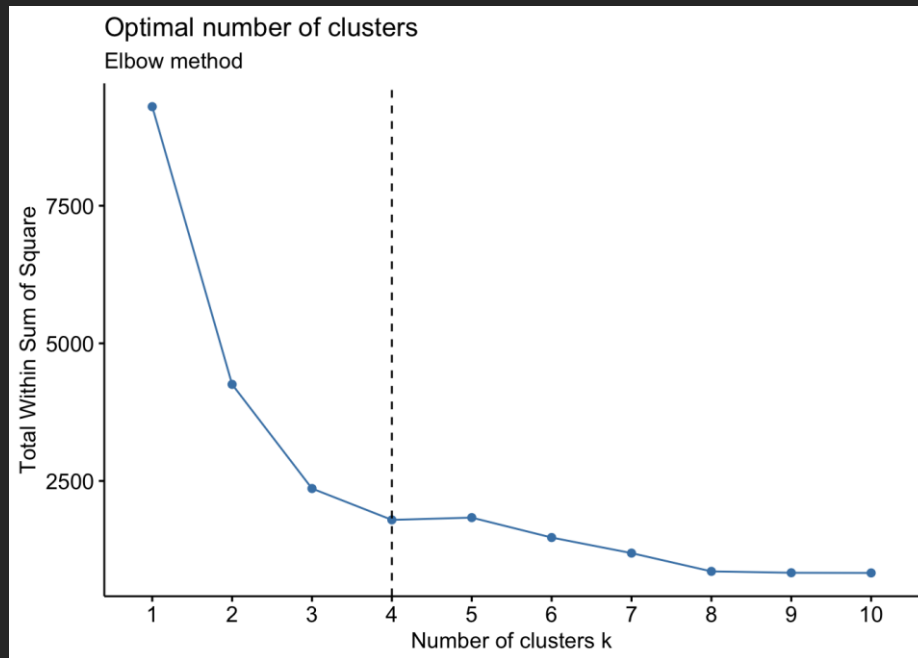
In order to find the optimal number of clusters for a k-means, it is recommended to choose it based on:

The context of the problem at hand, for instance if you know that there is a specific number of groups in your data (this is option is however subjective), or

The following four approaches:

- Elbow method (which uses the within cluster sums of squares)
- Average silhouette method
- Gap statistic method
- NbClust() function


# K-means: Elbow method



- One way to determine the number of clusters is the elbow method.
- The elbow method makes use of the percentage of variance explained
  - Ratio between-group variance to the total variance
- We can obtain different clustering for different value of k and then plot the percentage of variance explained versus k
  - The *elbow* cannot always be identified unambiguously

# K-means: silhouette analysis

Another approach in choosing the number of clusters is through silhouette analysis



Silhouette analysis calculates a silhouette coefficient for every point  $x_i$  as follows:

$A_i$  is the average distance between  $x_i$  and all the other data points that are in the same cluster with  $x_i$

$B_i$  is the smallest average distance between  $x_i$  and all the points in any other cluster that  $x_i$  is not a member

# K-means: silhouette analysis

- 
- The silhouette coefficient for  $\mathbf{x}_i$  is then:

$$s(\mathbf{x}_i) = \frac{B_i - A_i}{\max(A_i, B_i)}$$

- The silhouette coefficient is between -1 and 1
  - If  $s(\mathbf{x}_i)$  is close to 1 it means that the cluster assignment for  $\mathbf{x}_i$  is *good*
  - If  $s(\mathbf{x}_i)$  is close to -1 it means that the cluster assignment for  $\mathbf{x}_i$  is *bad*
  - If  $s(\mathbf{x}_i)$  is close to 0 it means that the  $\mathbf{x}_i$  is close to the border of the two clusters

# K-means



**One can then calculate the average silhouette coefficient  $\sigma$  over all the points in the dataset for different number of clusters  $k$**

The value of  $k$  that maximizes the average silhouette coefficient can then be chosen as the most appropriate number of clusters



**Some general (heuristic) rules:**

$\sigma > 0.7$ : a *strong* structure has been found  
 $0.5 < \sigma < 0.7$ : a *reasonable* structure has been found  
 $0.25 < \sigma < 0.5$ : a *weak* structure has been found  
 $\sigma < 0.25$ : *no* significant structure has been found

# K-means

- When using a heuristic to choose the number of clusters we need to be aware that this is exactly this, i.e., a **heuristic**
- We need to be aware of the limitations of each approach (recall clustering is an ill-posed problem to begin with and this is one of its manifestations)
- Sometimes choosing the number of clusters is more of an *art* than *science*
  - Best approach might be using (multiple) heuristics, visual inspection and domain-specific knowledge

# Evaluating cluster performance

---

The Davies-Bouldin index:

- Calculates the intracluster (within-cluster) variance and the distance between the centroids of each cluster.
- For each cluster, its nearest neighboring cluster is identified, and the sum of their intracluster variances is divided by the difference between their centroids. This value is calculated for each cluster, and the Davies-Bouldin index is the mean of these values.

The Dunn index:

- Quantifies the ratio between the smallest distance between cases in different clusters and the largest distance within a cluster.

The pseudo F-statistic:

- Ratio of the between-cluster sum of squares to the within-cluster sum of squares. The grand centroid is shown as a square in the right-side plot.

# K-medoids clustering (Partitioning Around Medoids)

- Find representative objects (medoids) in clusters.
- PAM works effectively on small data sets, but does not scale well for large data sets (due to the computational complexity)
- Efficiency improvement on PAM
  - CLARA (Kaufmann & Rousseeuw, 1990): PAM on samples
  - CLARANS (Ng & Han, 1994): Randomized re-sampling



# K-medoids clustering (Partitioning Around Medoids)

In Partitioning Around Medoids (PAM), we do the following steps to find cluster centers:

- Choose  $k$  data points from the scatter plot as starting points for cluster centers.
- Calculate their distance from all the points in the scatter plot.
- Classify each point into the cluster whose center it is closest to.
- Select a new point in each cluster that minimizes the sum of distances of all points in that cluster from itself.
- Repeat Step 2 until the centers stop changing.

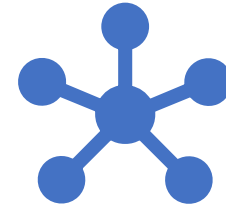
# K-means versus K-medoids clustering



Computational  
complexity



Presence of outliers



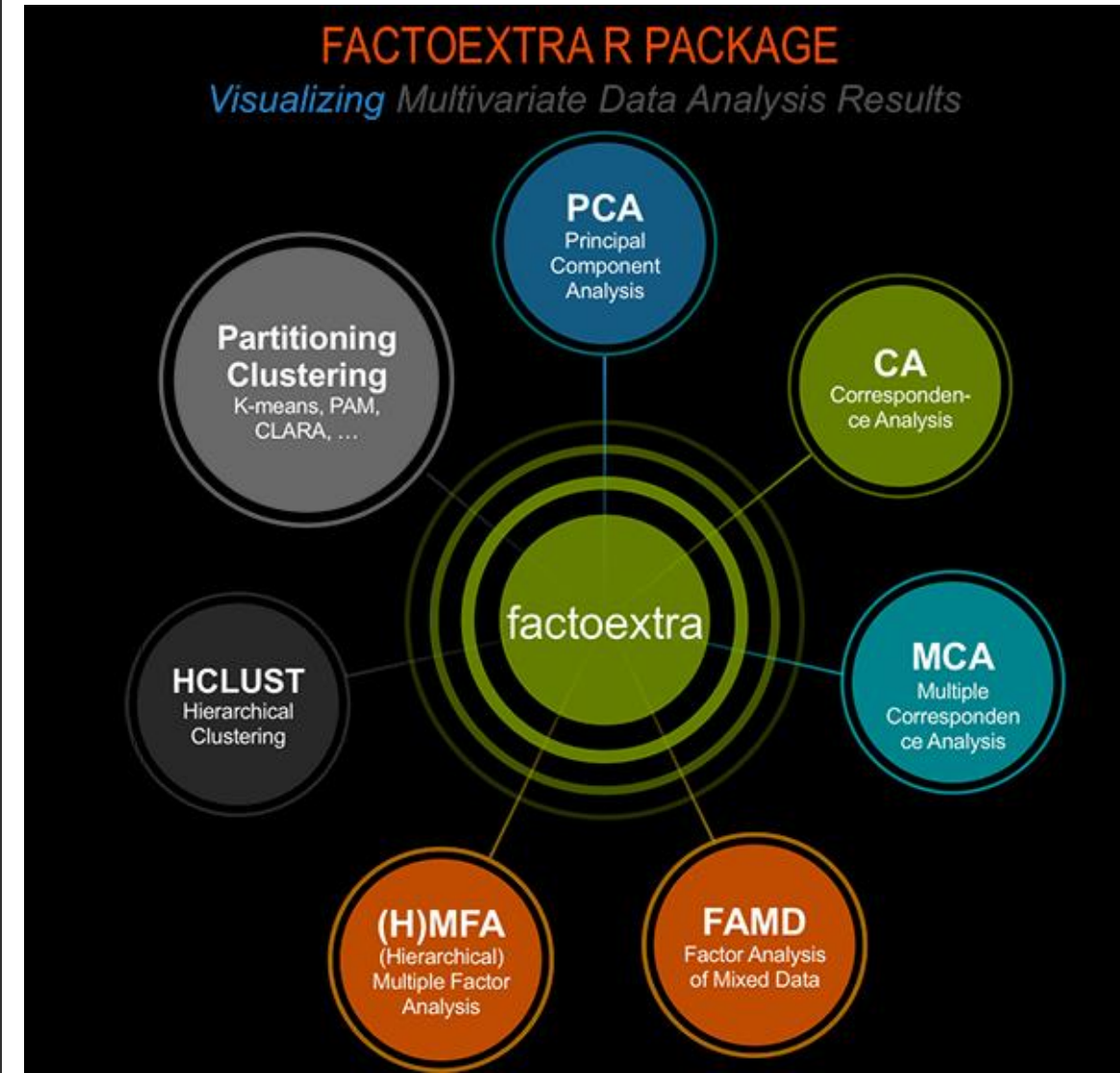
Cluster centers

# In Summary

- ✓ The goal of clustering is to discover similarities among subsets of your data.
- ✓ In a good clustering, points in the same cluster should be more similar (nearer) to each other than they are to points in other clusters.
- ✓ When clustering, the units that each variable is measured matters. Different units cause different distances and potentially different clusterings. Make sure to transform all the columns to have a mean value of 0 and a standard deviation of 1.0, for example by using the function `scale()`.
- ✓ Clustering is often used for data exploration or as a precursor to supervised learning methods.
- ✓ Like visualization, it's more iterative and interactive, and less automated than supervised methods.
- ✓ You should consider different approaches, with different numbers of clusters as different clustering algorithms will give different results.
- ✓ Consider the results from different heuristics, estimating the best number of clusters. and explore various numbers of clusters.

# factoextra contains many functions for cluster analysis and visualization

- `dist(fviz_dist, get_dist)` Distance Matrix Computation and Visualization
- `get_clust_tendency` Assessing Clustering Tendency
- `fviz_nbclust(fviz_gap_stat)` Determining the Optimal Number of Clusters
- `fviz_dend` Enhanced Visualization of Dendrogram
- `fviz_cluster` Visualize Clustering Results
- `fviz_mclust` Visualize Model-based Clustering Results
- `fviz_silhouette` Visualize Silhouette Information from Clustering
- `hcut` Computes Hierarchical Clustering and Cut the Tree
- `hkmeans` Hierarchical k-means clustering
- `eclust` Visual enhancement of clustering analysis

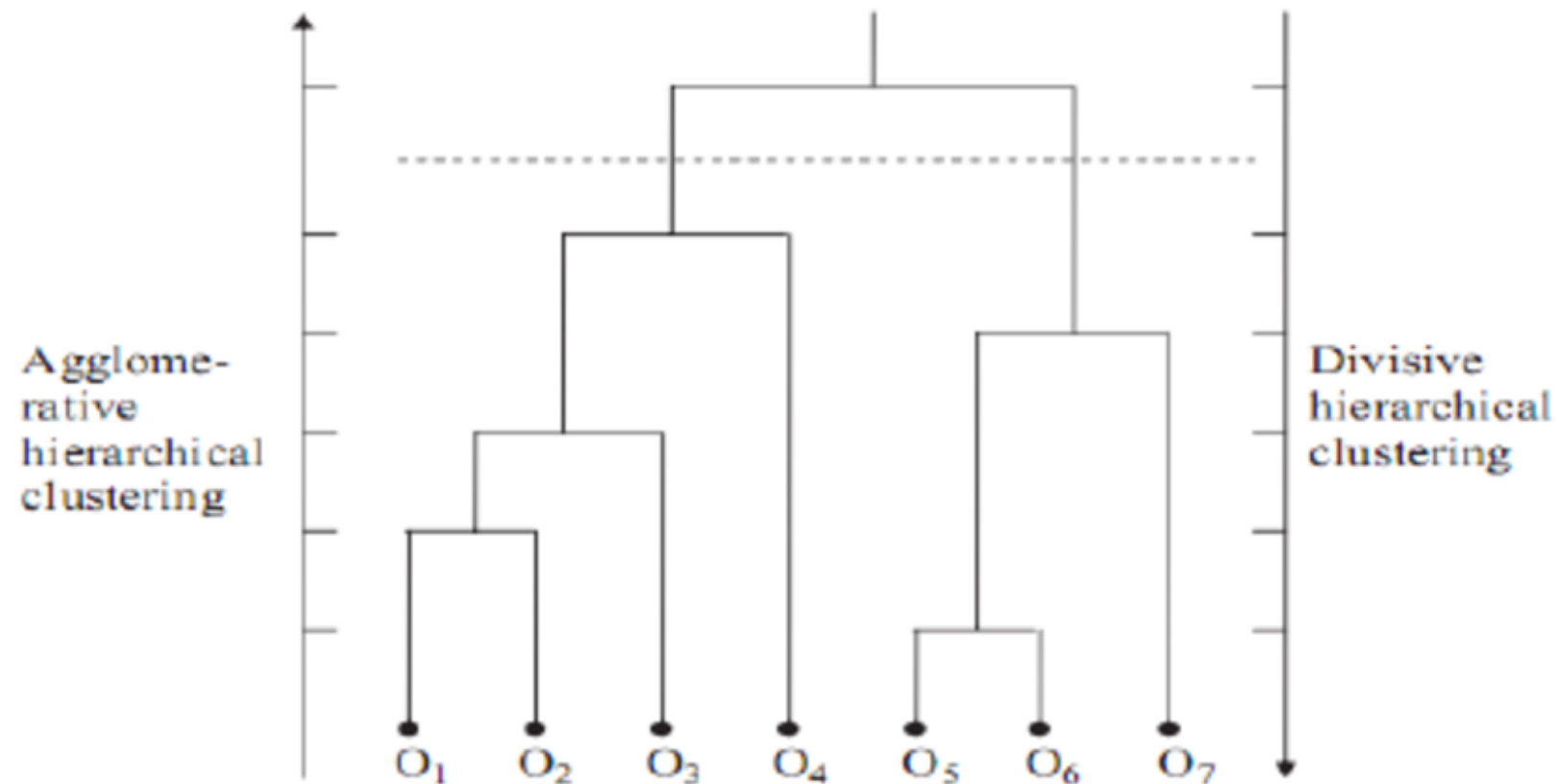


# eclust()

- **x**: numeric vector, data matrix or data frame
- **FUNcluster**: a clustering function including “kmeans”, “pam”, “clara”, “fanny”, “hclust”, “agnes” and “diana”. Abbreviation is allowed.
- **hc\_metric**: character string specifying the metric to be used for calculating dissimilarities between observations. Allowed values are those accepted by the function `dist()` [including “euclidean”, “manhattan”, “maximum”, “canberra”, “binary”, “minkowski”] and correlation based distance measures [“pearson”, “spearman” or “kendall”]. Used only when FUNcluster is a hierarchical clustering function such as one of “hclust”, “agnes” or “diana”.
- ...: other arguments to be passed to FUNcluster.

# hierarchical clustering

- Understanding hierarchical clustering
- Using linkage methods
- Measuring the stability of

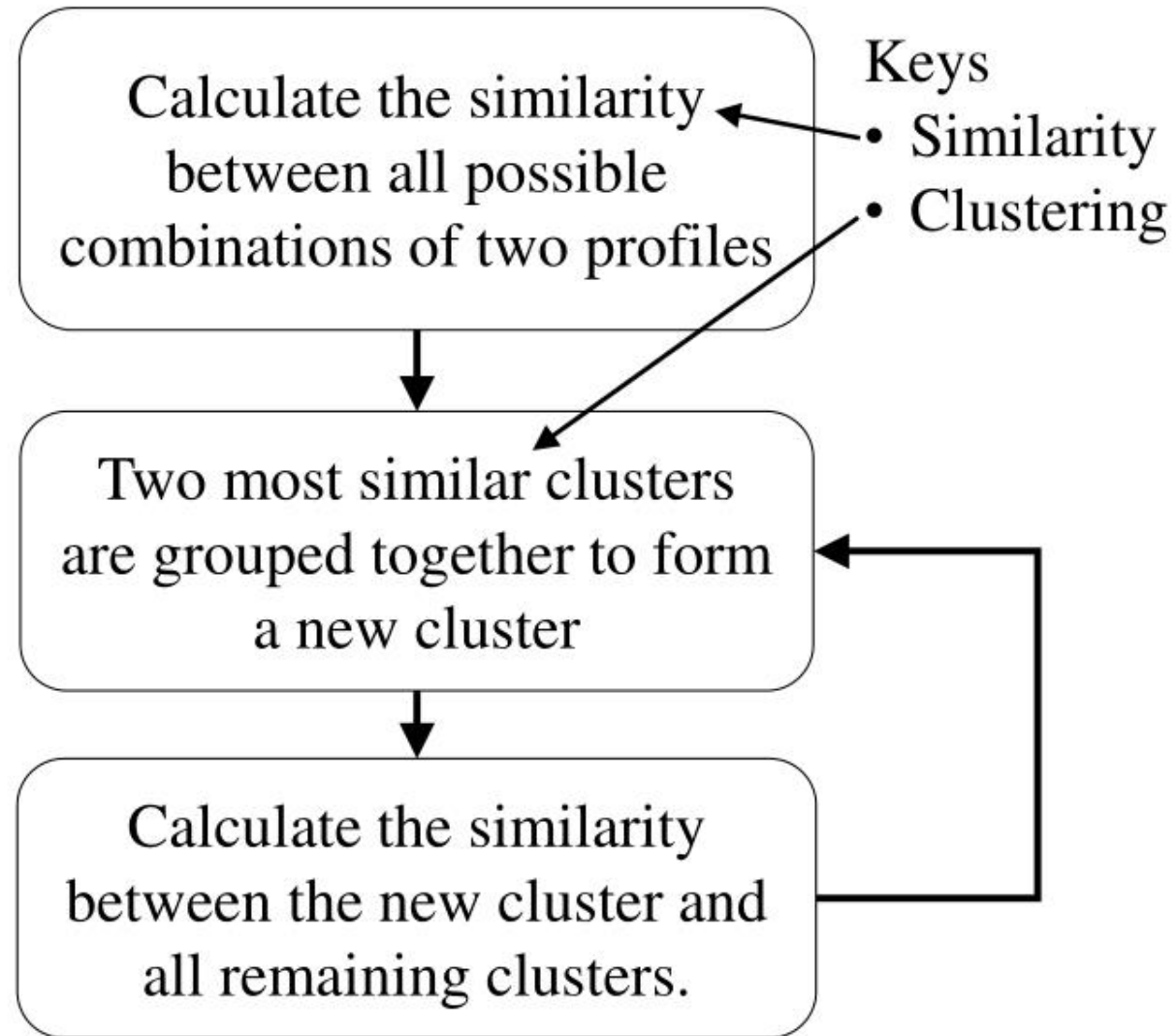


# Hierarchical Clustering

---

- ❑ Two main types of hierarchical clustering
  - Agglomerative (bottom up):
    - ❑ Start with the points as individual clusters
    - ❑ At each step, merge the closest pair of clusters until only one cluster (or  $k$  clusters) left
  - Divisive (top down):
    - ❑ Start with one, all-inclusive cluster
    - ❑ At each step, split a cluster until each cluster contains a point (or there are  $k$  clusters)
- ❑ Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

# Hierarchical Clustering





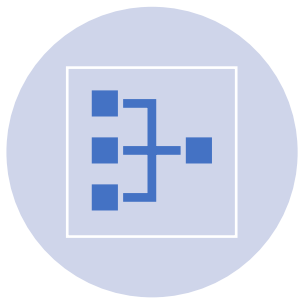
# Agglomerative clustering algorithm



This is the most common type of hierarchical clustering algorithm. It's used to group objects in clusters based on how similar they are to each other.



This is a form of bottom-up clustering, where each data point is assigned to its own cluster. Then those clusters get joined together.

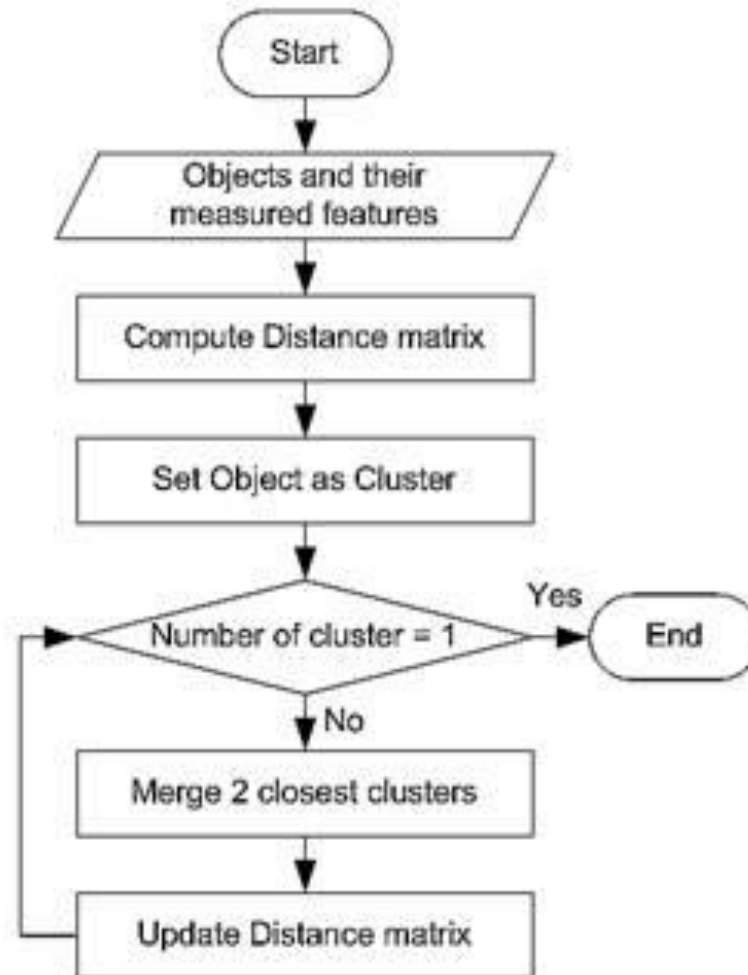


At each iteration, similar clusters are merged until all of the data points are part of one big root cluster.

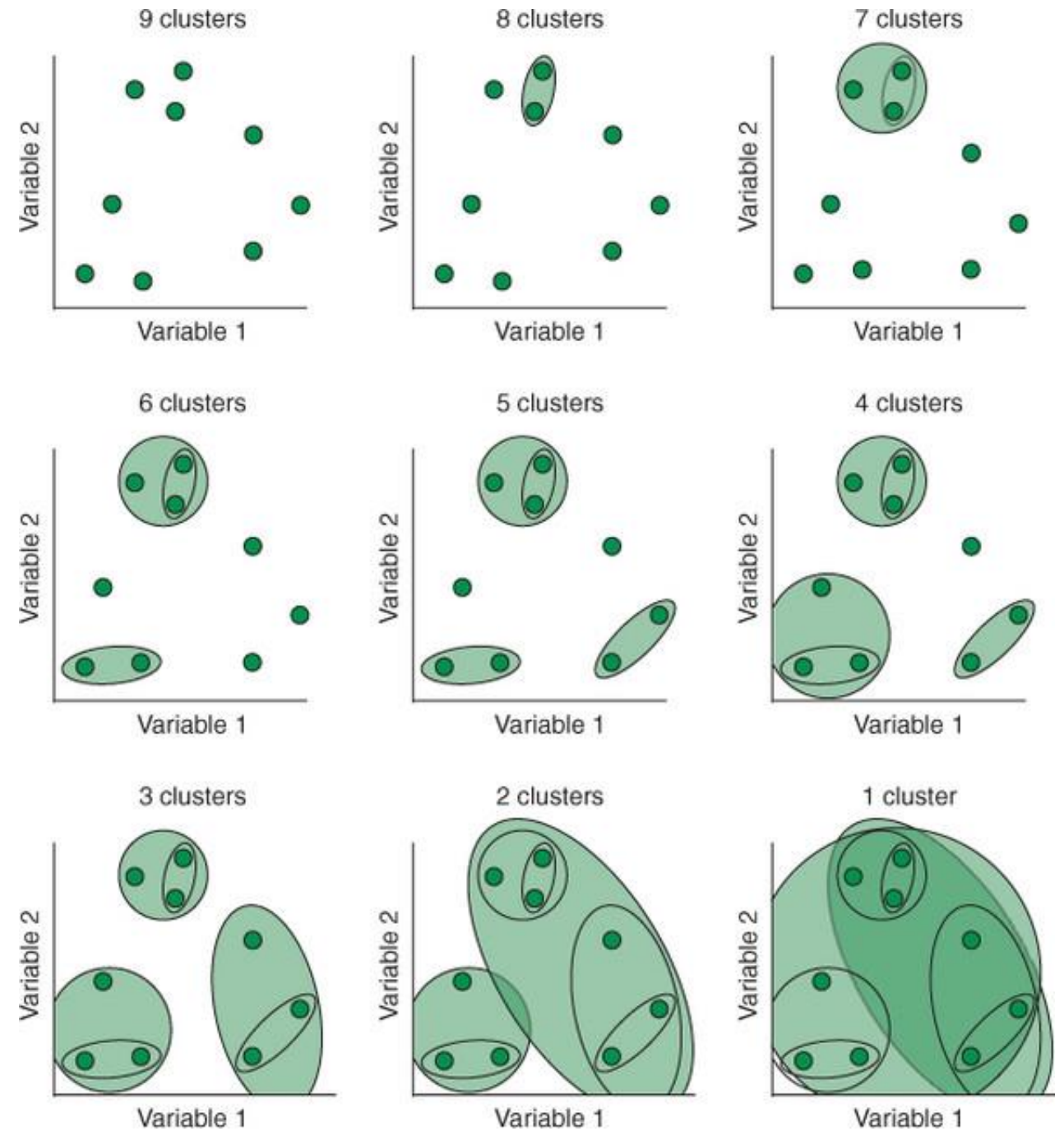


Agglomerative clustering is best at finding small clusters. The result looks like a dendrogram so that you can easily visualize the clusters when the algorithm finishes.

# Agglomerative clustering algorithm

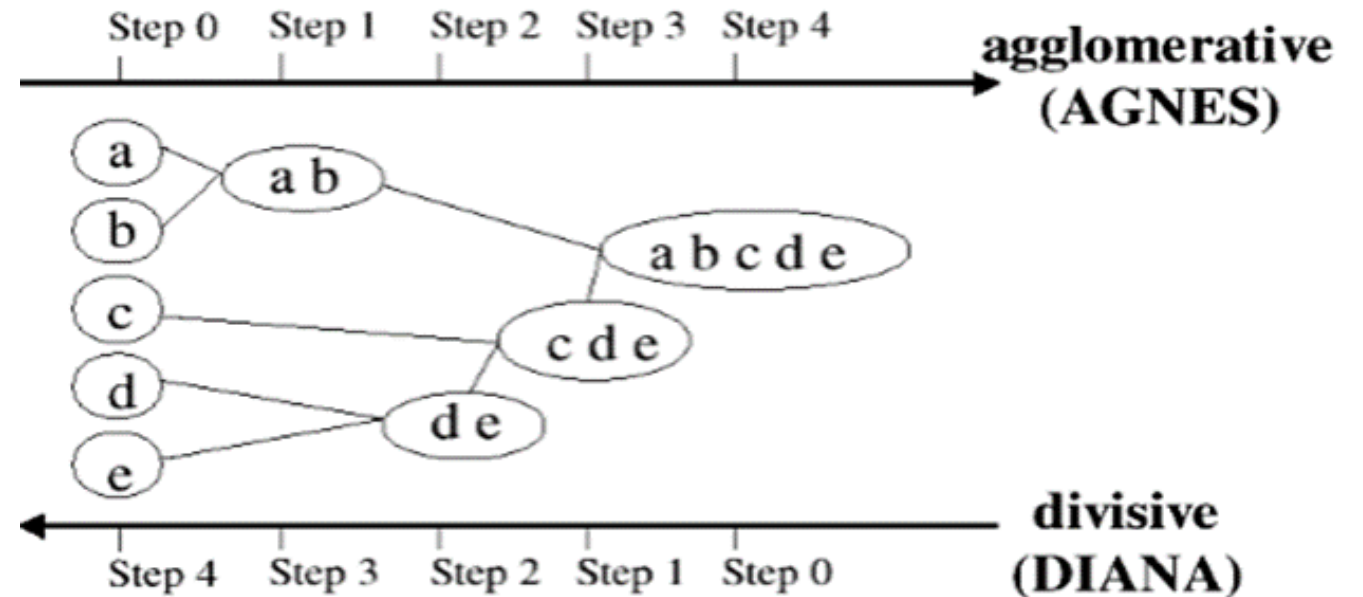


# Agglomerative

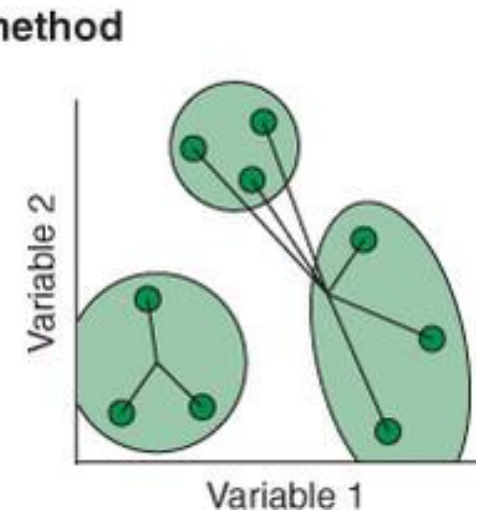
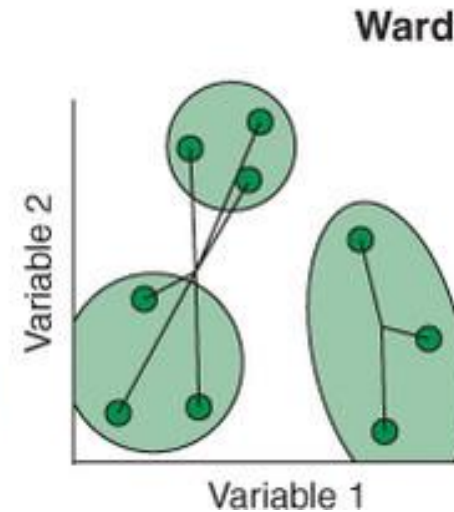
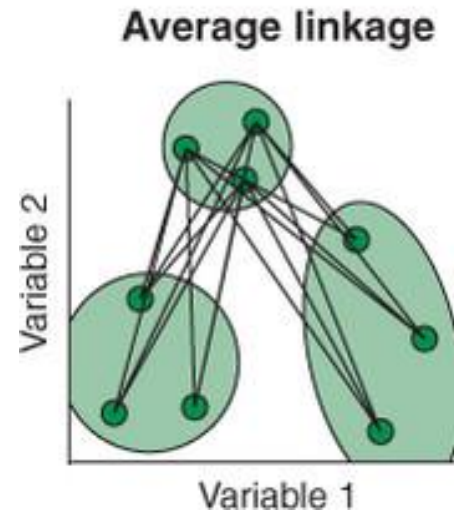
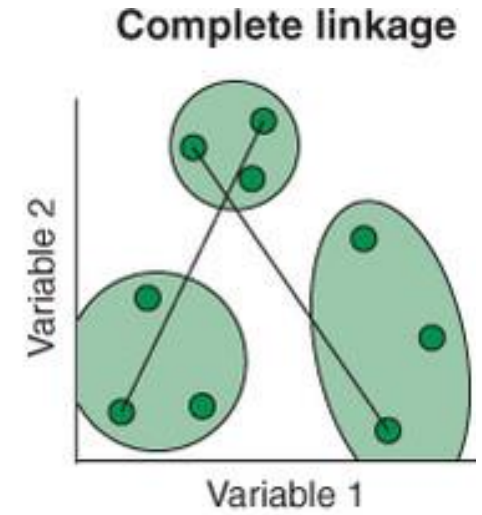
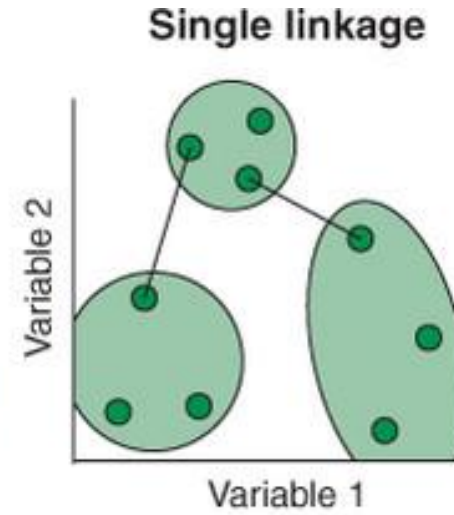
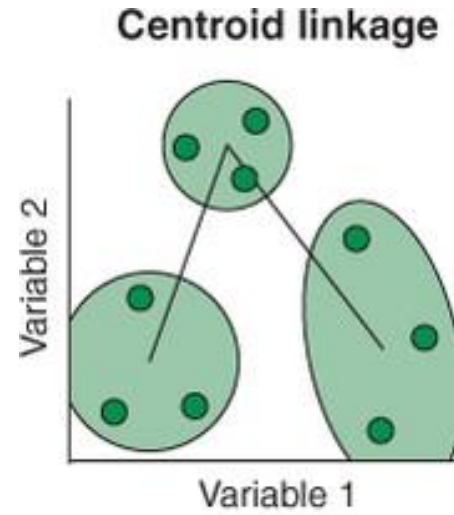


# Divisive Clustering (DIANA)

- Inverse order of Agglomerative (AGNES)
- Starts with one cluster of the data and then partitions the appropriate cluster. Eventually each node forms a cluster on its own



Different  
linkage  
methods to  
define the  
distance  
between  
clusters.



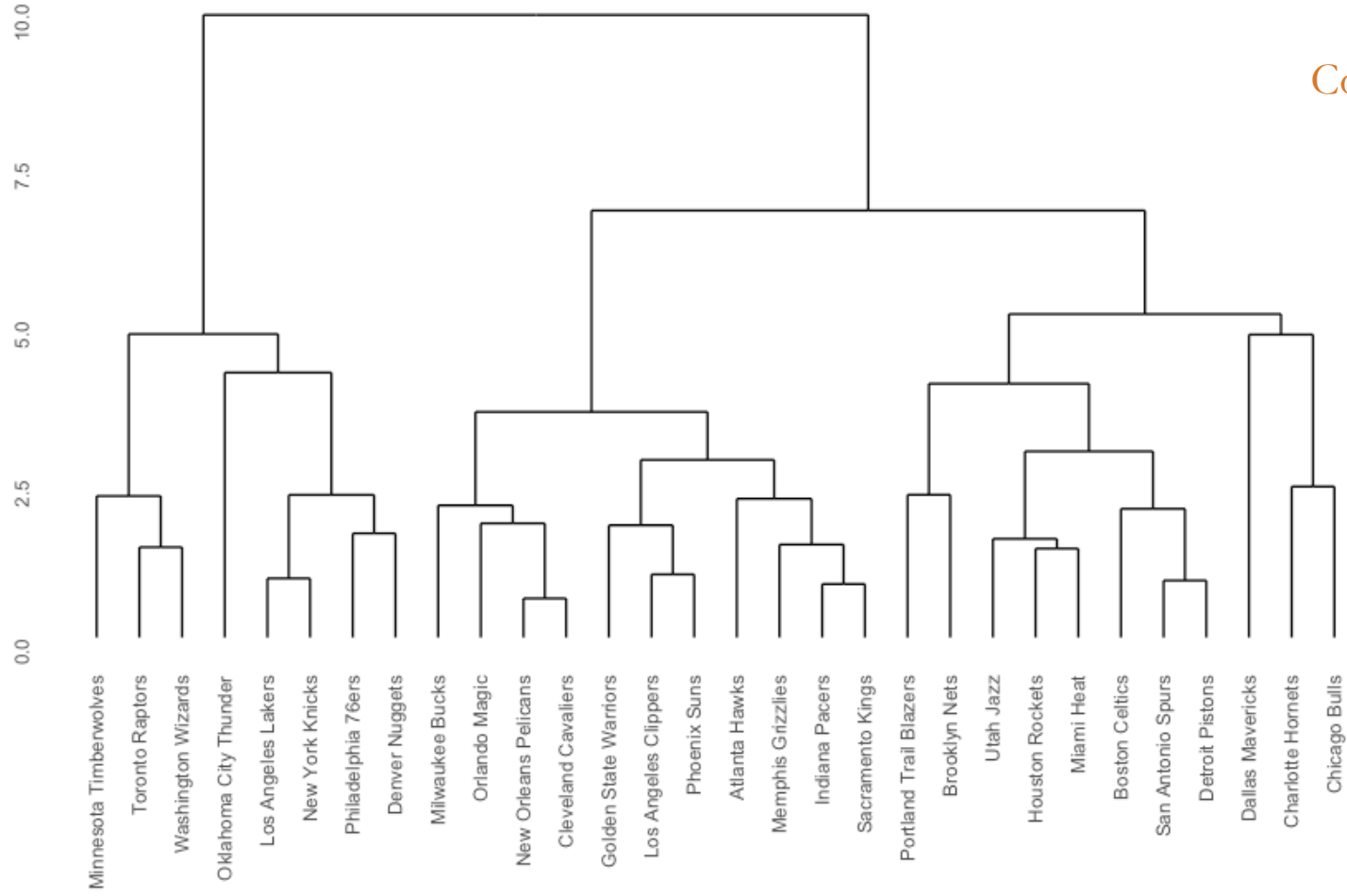
There are 5 main methods to measure the distance between clusters, referred as linkage methods:

1. Single linkage: computes the minimum distance between clusters before merging them.
2. Complete linkage: computes the maximum distance between clusters before merging them.
3. Average linkage: computes the average distance between clusters before merging them.
4. Centroid linkage: calculates centroids for both clusters, then computes the distance between the two before merging them.
5. Ward's (minimum variance) criterion: minimizes the total within-cluster variance and find the pair of clusters that leads to minimum increase in total within-cluster variance after merging.

# Hierarchical clustering

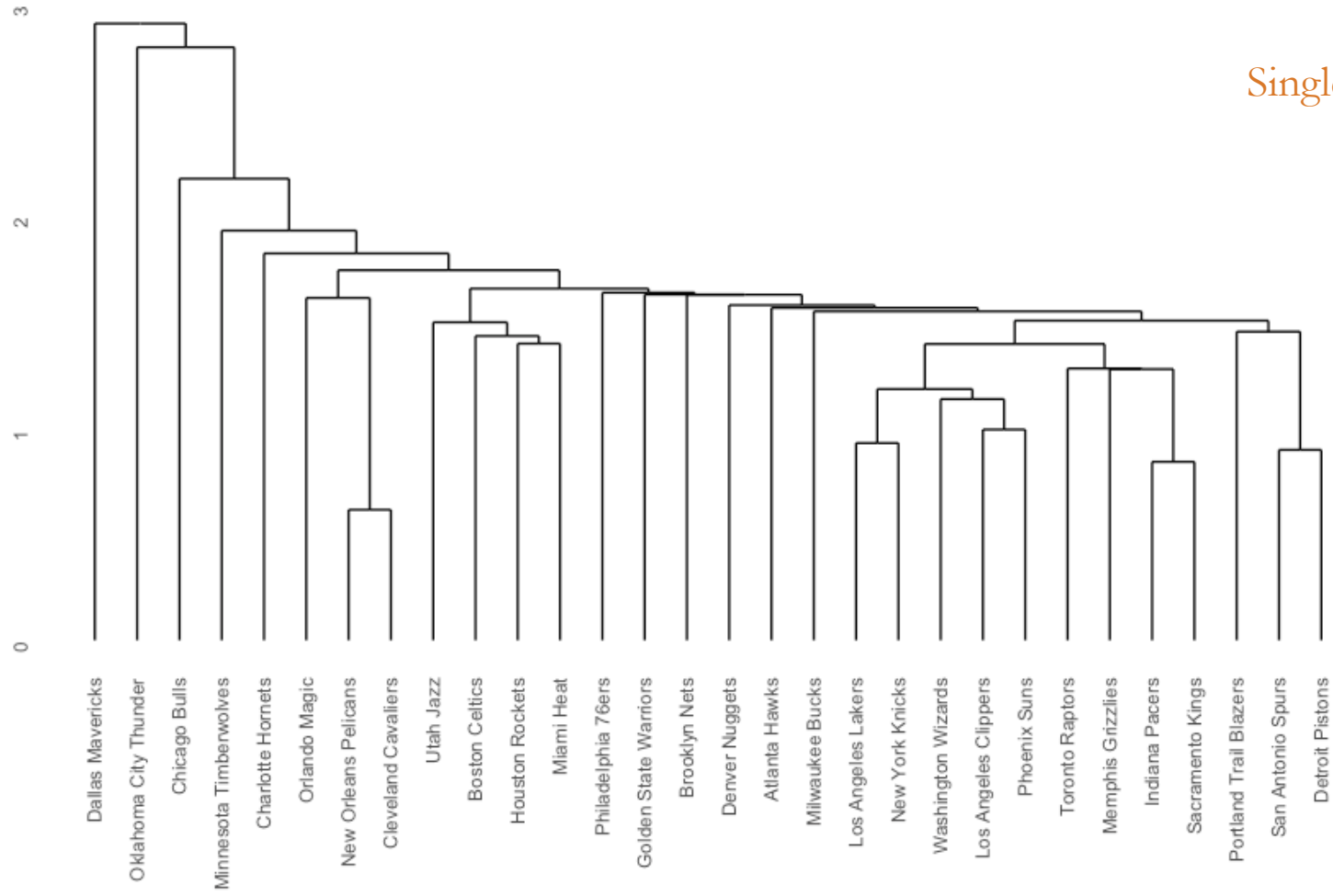
---

- Single-linkage: maximum similarity between data points in the two groups, i.e.,  $\sigma_{single}(A, B) = \max_{i \in A, j \in B} \sigma(\mathbf{x}_i, \mathbf{x}_j)$
- Complete-linkage: minimum similarity between data points in the two groups, i.e.,  $\sigma_{complete}(A, B) = \min_{i \in A, j \in B} \sigma(\mathbf{x}_i, \mathbf{x}_j)$
- Average-linkage: average similarity between data points in the two groups, i.e.,  $\sigma_{average}(A, B) = \frac{1}{n_A n_B} \sum_{i \in A} \sum_{j \in B} \sigma(\mathbf{x}_i, \mathbf{x}_j)$

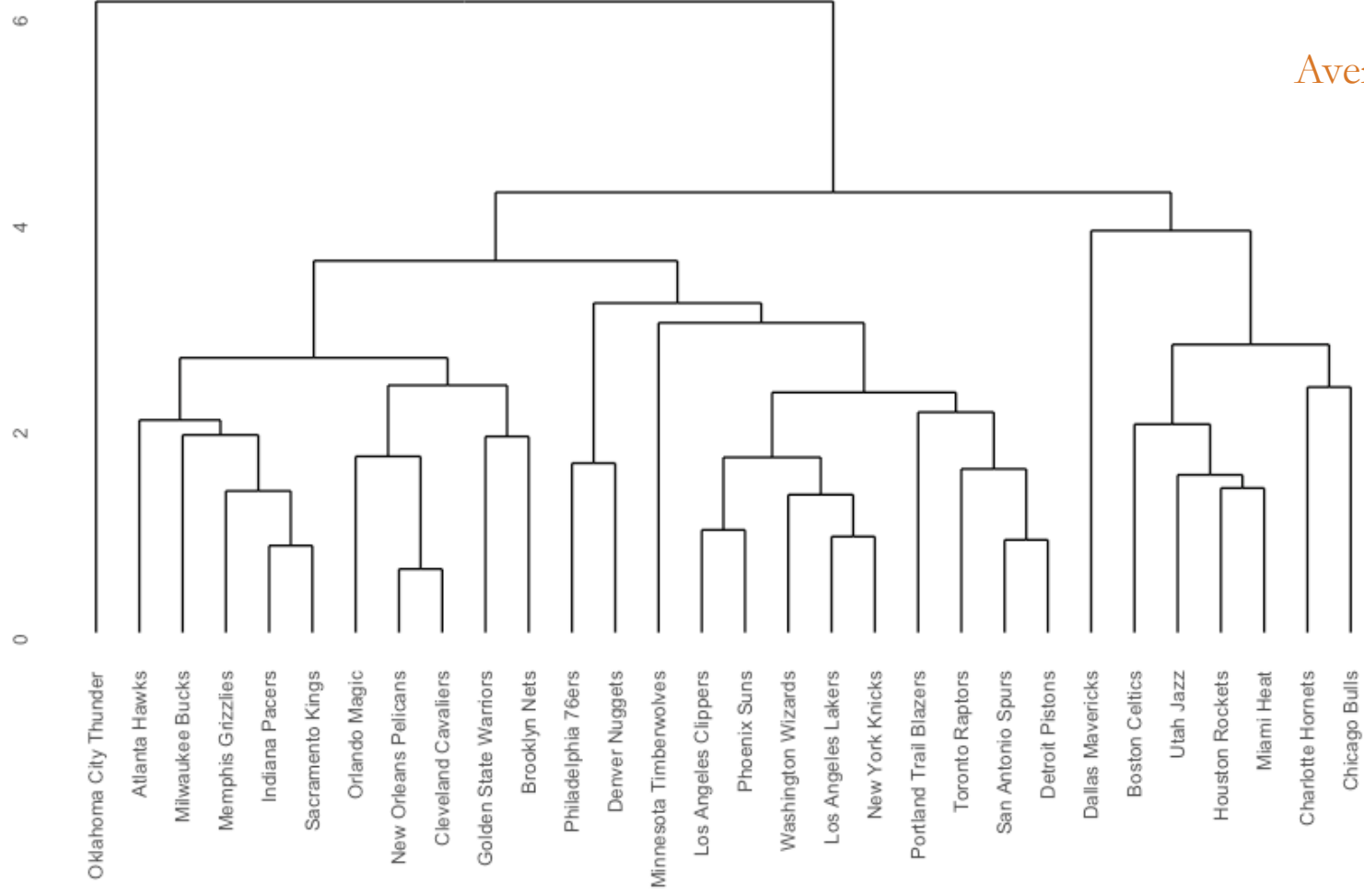


Complete linkage





Single linkage



Average linkage

# Hierarchical clustering

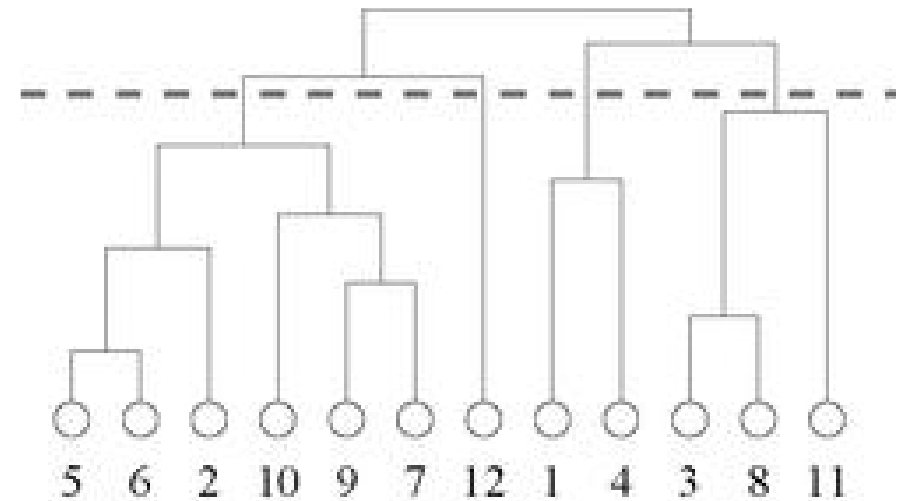
---

- Once we have decided upon a similarity measure  $\sigma$  and a linkage function we proceed as follows:
  - We assign each data point to a group of its own ( the initial similarities of the groups are simply the similarities of the data points )
  - We find the pair of groups with the highest similarity and join them together into a single group
  - We calculate the similarity between the new group and all the others
  - We repeat the last 2 steps until all data points have been joined to a single group

# Hierarchical clustering

- The final result of hierarchical clustering is represented through a dendrogram that represents the groups identified during the different steps of the agglomerative process
- From the dendrogram we can obtain several configurations by drawing a horizontal line that cuts through the dendrogram

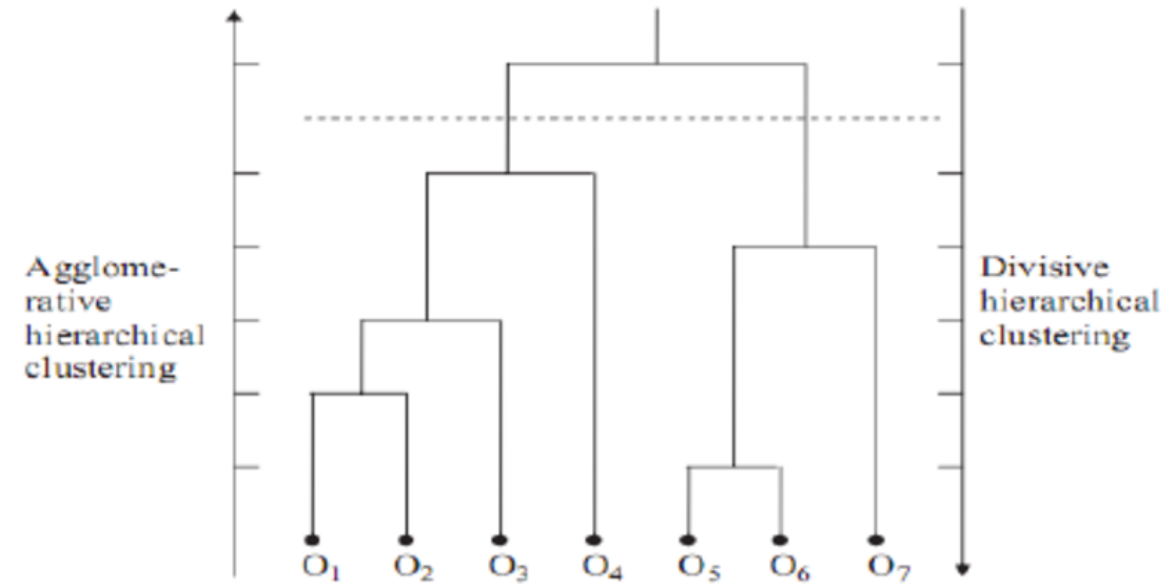
For example, the dashed line in the example here provides 4 groups/clusters



# Dendrogram

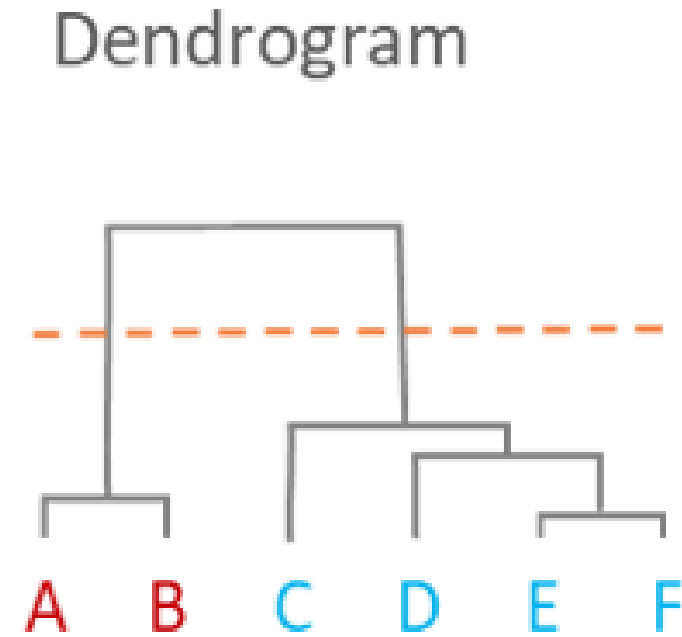
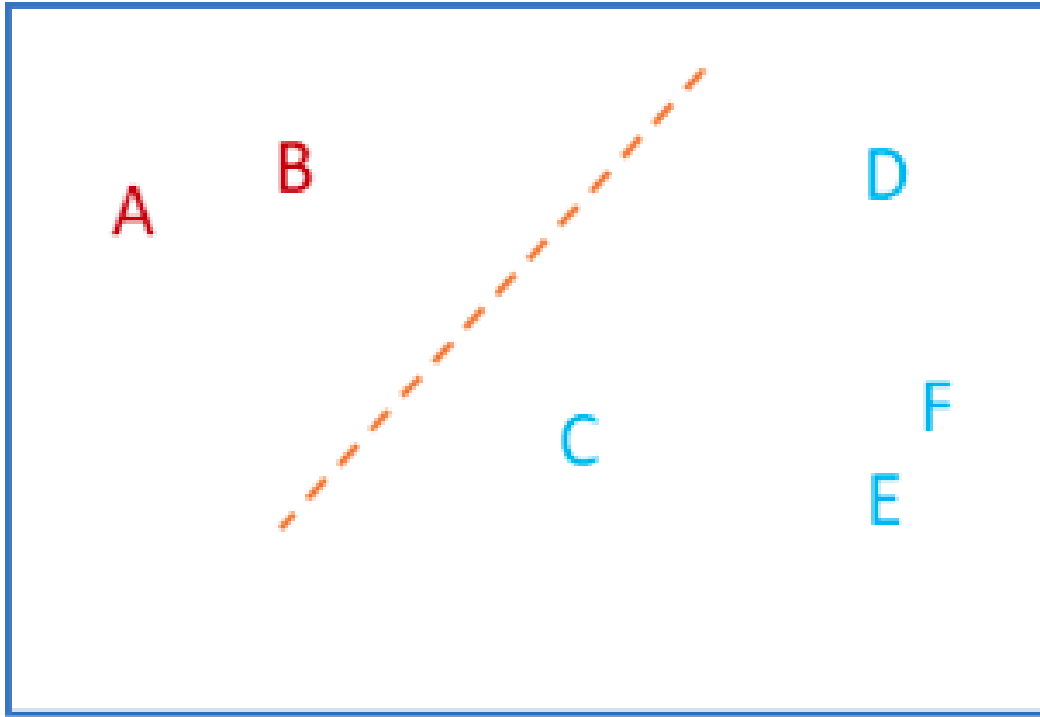
---

- A tree-like structure frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering.
- Hierarchical classifications produced by either
  - Agglomerative
  - Divisive



# Allocating observations to clusters

- Observations are allocated to clusters by drawing a horizontal line through the dendrogram.
  - Observations that are joined together below the line are in clusters.



# Measuring Performance: Clustering validation

3 kinds of measures: Internal, external and relative clustering validation

1. Internal: unsupervised, criteria derived from data itself
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are (e.g., silhouette coefficient)
2. External: supervised, employ criteria not inherent to the dataset
  - Compare a clustering against prior or expert-specified knowledge (i.e., the ground truth) using certain clustering quality
3. Relation: Directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm

# Measuring cluster quality: External

Clustering quality measure  $Q$  is good if it satisfies the following four essential criteria:

- Cluster homogeneity: the purer, the better
- Cluster completeness: should assign objects belonging to the same category in the ground truth to the same cluster.
- Rag bag: putting a heterogeneous object into a pure cluster should be penalized more than putting it into a rag bag (“other” category)
- Small cluster preservation: splitting a small category into pieces is more harmful than splitting a large category into pieces





# Extensions to Hierarchical clustering

---

- Major weakness of AGNES
  - Can never undo what was done previously
  - Does not scale well: time complexity of at least
- Integration of hierarchical and distance-based clustering
- BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters

# In Summary

- Hierarchical clustering uses the distances between cases to learn a hierarchy of clusters.
- How these distances are calculated is controlled by our choice of linkage method.
- Hierarchical clustering can be bottom-up (agglomerative) or top-down (divisive).
- A flat set of clusters can be returned from a hierarchical clustering model by “cutting” the dendrogram at a particular height.
- Cluster stability can be measured by clustering on bootstrap samples and using the Jaccard index to quantify the agreement of cluster membership between samples.

# dendrogram visualizations in R

- [1 plot.hclust\(\): R base function](#)
- [2 plot.dendrogram\(\) function](#)
- [3 Phylogenetic trees](#)
- [4 ggdendro package : ggplot2 and dendrogram](#)
  - [Visualize dendrogram using ggdendrogram\(\) function](#)
  - [Extract dendrogram plot data](#)
- [5 dendextend package: Extending R's dendrogram functionality](#)
  - [How to change a dendrogram](#)
  - [Create a simple dendrogram](#)
  - [Change labels](#)
  - [Change the points of a dendrogram nodes/leaves](#)
  - [Change the color of branches](#)
  - [Adding colored rectangles](#)
  - [Adding colored bars](#)
  - [ggplot2 integration](#)
  - [pvclust and dendextend](#)

# The curse of (high) dimensionality

- There are several facets, but the main idea is that when the **dimensionality increases**, the **volume of the space increases faster than the available data**.
  - Sparsity
- With regards to clustering the concepts of proximity and distance may **not** be qualitatively **meaningful**.
  - When Euclidean distance is defined in a high dimensional space, there is little difference in the distances between different pairs of samples!
  - **In high dimensions all data points look alike!**

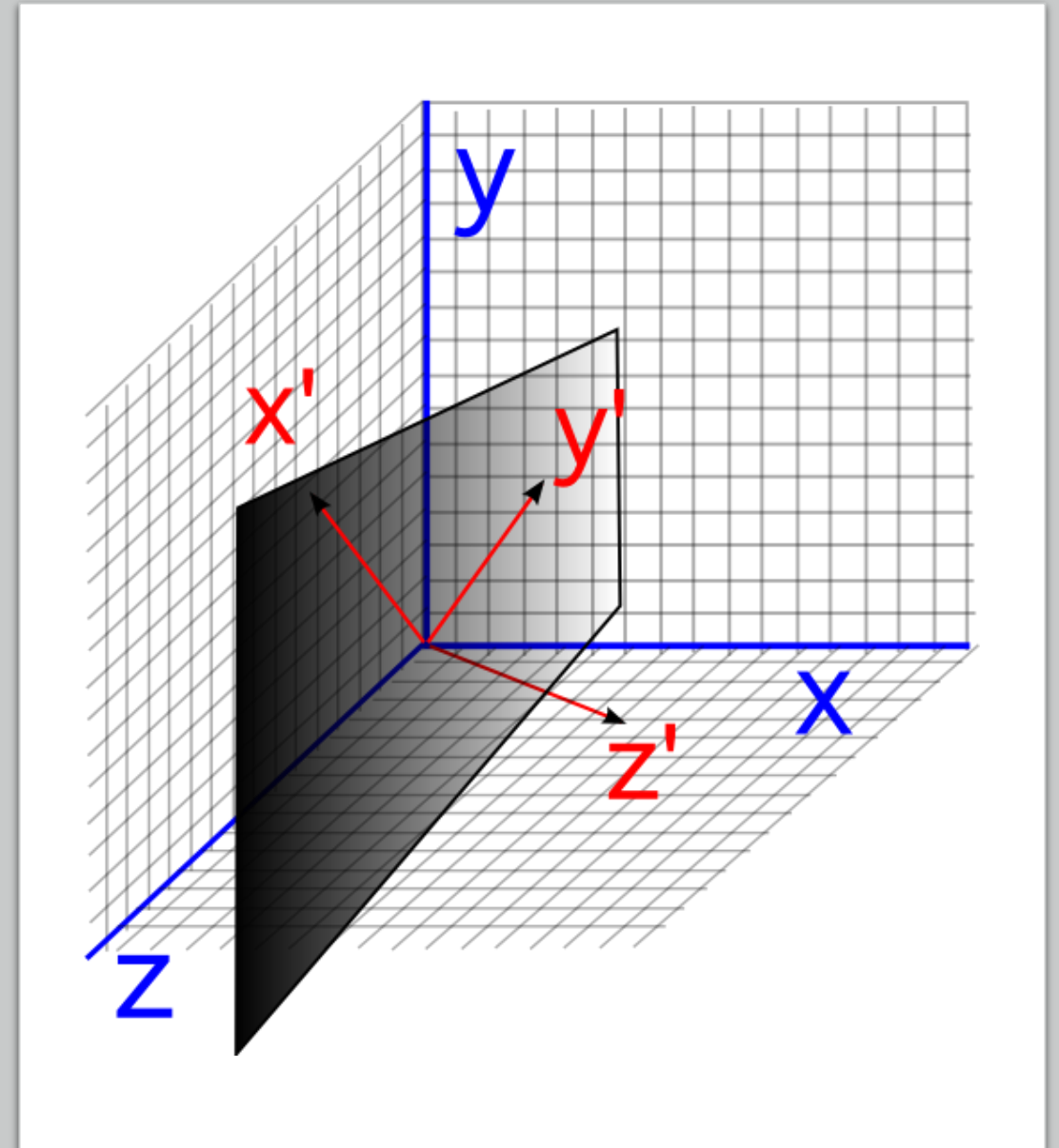
Distance metric choice can be crucial!

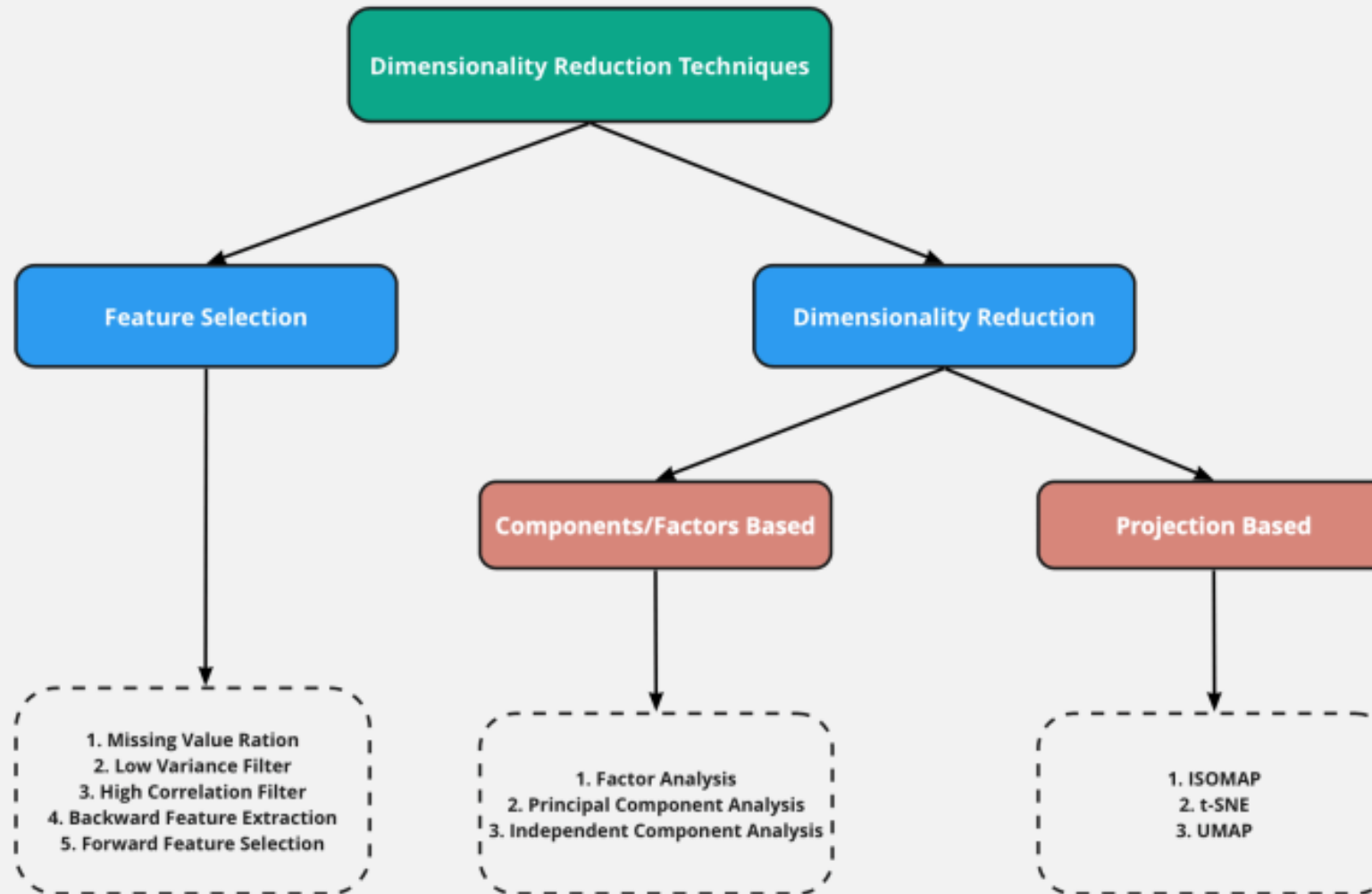
# The curse of (high) dimensionality

- Our intuition fails in high dimensions
  - Most of the mass of a multivariate Gaussian distribution is not near the mean, but in an increasingly distant *shell* around it
  - Most of the volume of the distribution/data is towards the boundaries
- In **practical** applications the curse of dimensionality is counteracted by the blessing of non-uniformity
  - Data instances are not spread uniformly through the input space but are concentrated on/near a low-dimensional manifold

# Dimensionality Reduction

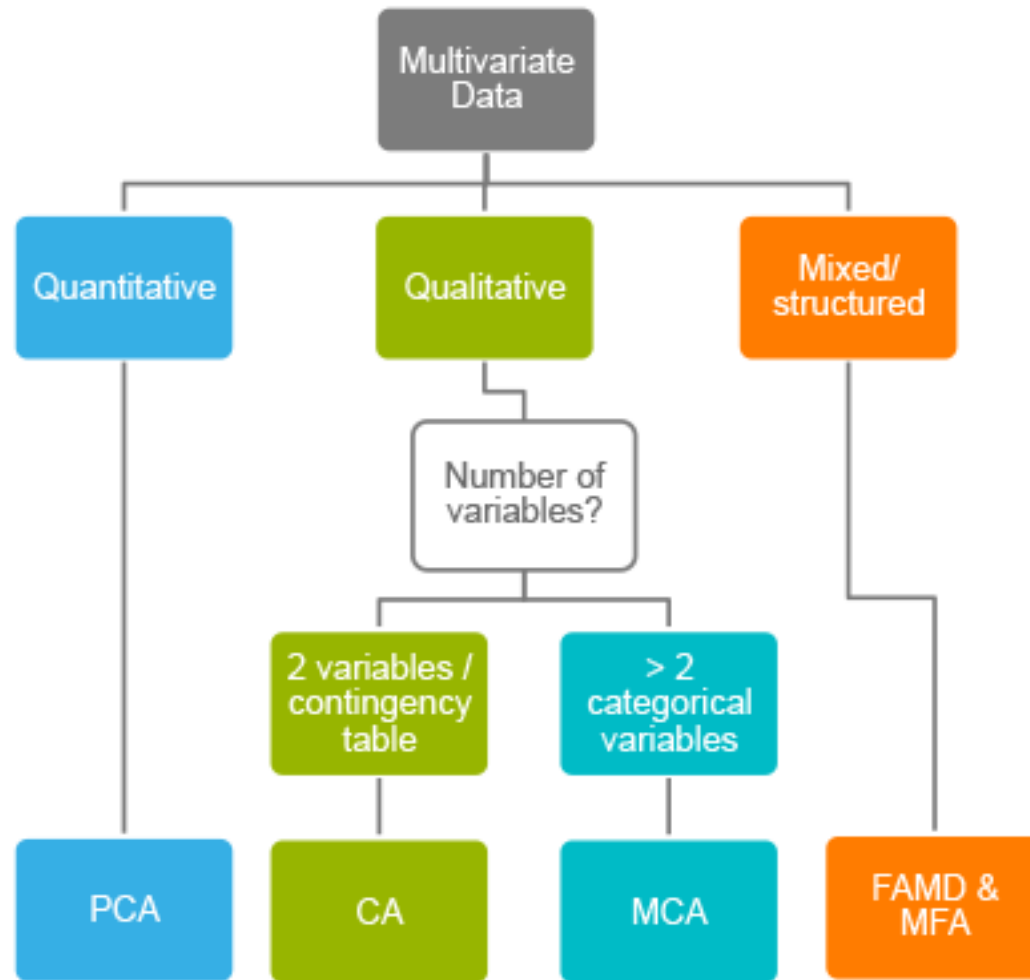
- The blessing of non-uniformity can be exploited through explicitly reducing the dimensionality of the data
  - The effective dimension of the data is smaller
- Linear and non-linear dimensionality reduction
- We have seen some dimensionality reduction methods
  - Which ones?





# Principal Component Methods

*Methods to Summarize & Visualize Multivariate Data*

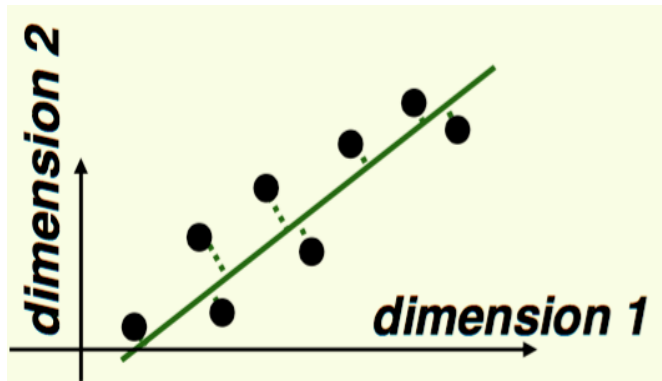


- PCA: Principal Component Analysis
- (M) CA: (Multiple) Correspondence Analysis
- FAMD: Factor Analysis of Mixed Data
- MFA: Multiple Factor Analysis

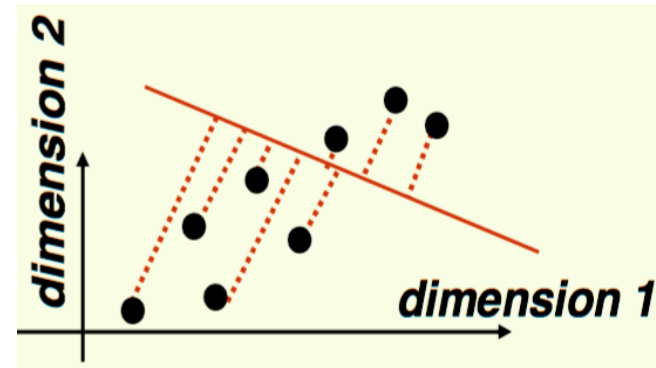


# Principal Component Analysis

- Main Idea: seek most accurate data representation in a lower dimensional space
- Example: Consider 2-dimensional data (**plane**) that you want to reduce into 1-dimension (**line**)
  - The task is now to find the line that minimizes the projection error for the data (from plane to line)



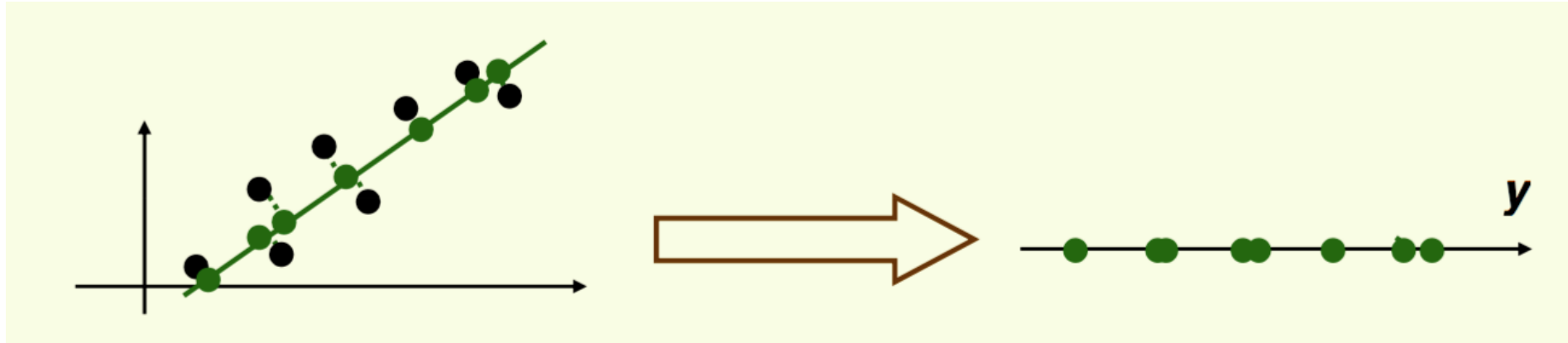
Small projection error!



Large projection error!

# Principal Component Analysis

- A **good** (the best too) line for **projection** lies in the **direction of largest variance**.



- We need to transform the coordinates.

# Principal Component Analysis

Standardize the data  $\mathbf{X}$

Compute the covariance matrix

Eigen-decompose the covariance matrix

$$\Sigma = \frac{1}{n-1} ((\mathbf{X} - \bar{\mathbf{x}})^T (\mathbf{X} - \bar{\mathbf{x}}))$$

$$\Sigma \mathbf{v} = \lambda \mathbf{v}$$

Sort eigenpairs from max to min eigenvalue

Explained variance

Projected data  $\mathbf{Y}$

$$\min_k \frac{\sum_{i=1}^k \lambda_i^2}{\sum_{j=1}^n \lambda_j^2} > \tau$$

$$\mathbf{P} = [\mathbf{v}_1 | \mathbf{v}_2 | \dots | \mathbf{v}_k]$$
$$\mathbf{Y} = \mathbf{XW}$$

# Principal Component Analysis

