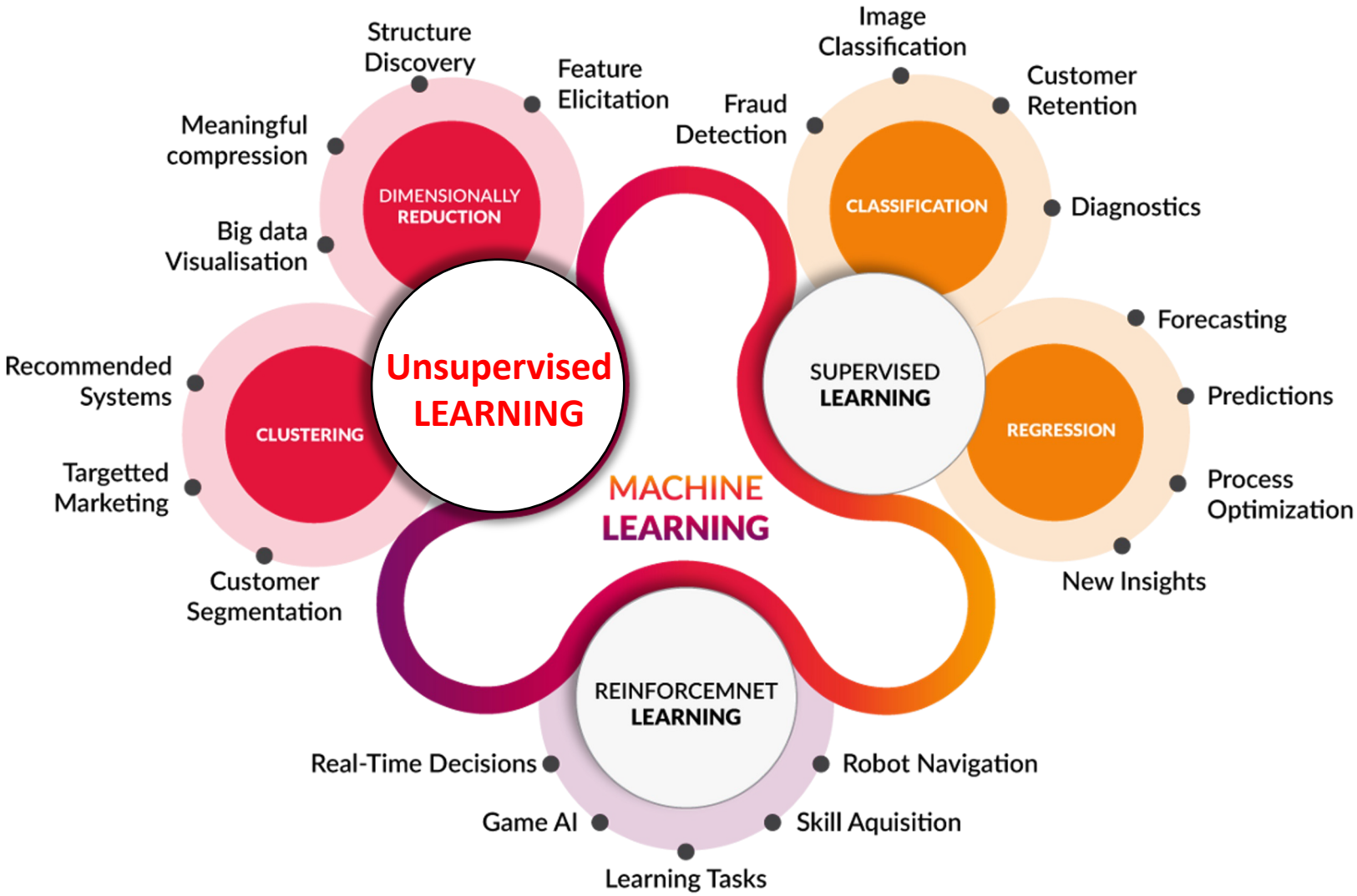
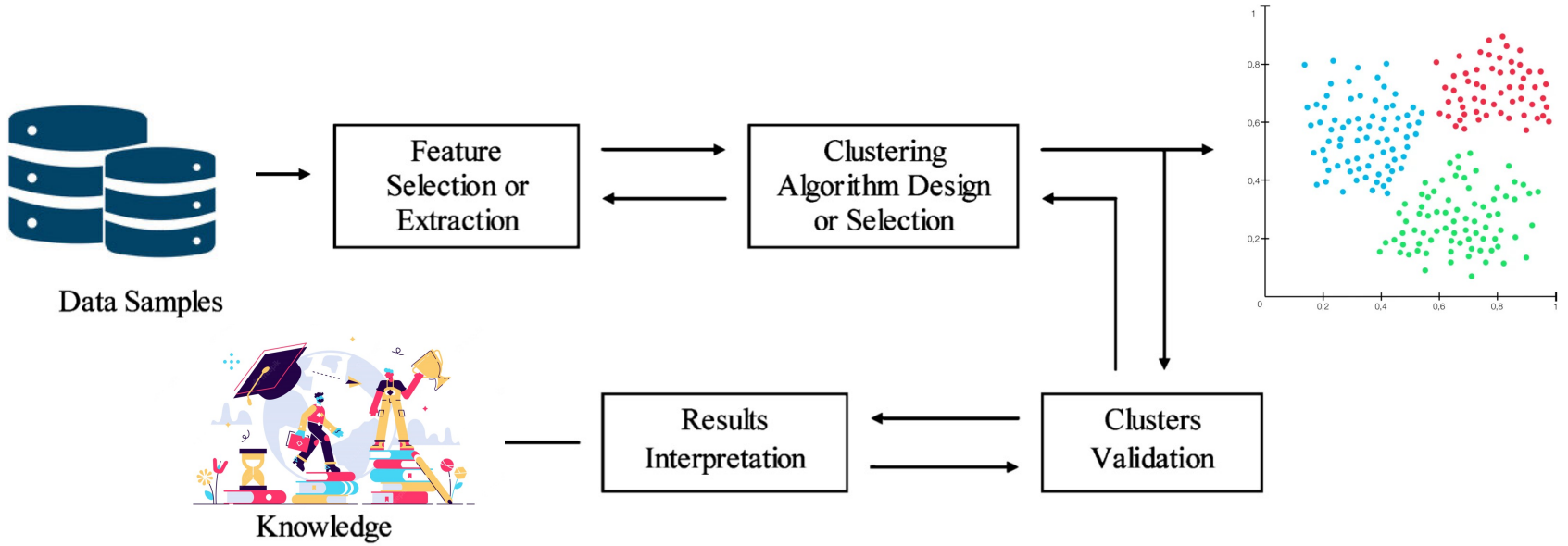

MACHINE LEARNING
Unsupervised LEARNING
SUPERVISED LEARNING
REINFORCEMNET LEARNING
DIMENSIONALLY REDUCTION
CLUSTERING
CLASSIFICATION
REGRESSION
Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
Recommended Systems
Targetted Marketing
Customer Segmentation
Image Classification
Customer Retention
Fraud Detection
Diagnostics
Forecasting
Predictions
Process Optimization
New Insights
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

Introduction

In unsupervised classification (called also clustering or exploratory data analysis), the goal is to separate a finite unlabelled dataset into a finite and discrete set of “natural,” hidden data structures [Ref. 1].



Introduction

1. ***Feature selection or extraction.*** Feature selection chooses distinguishing features from a set of candidates, while feature extraction utilizes some transformations to generate useful and novel features from the original ones. Both are very crucial to the effectiveness of clustering (and supervised learning) applications.

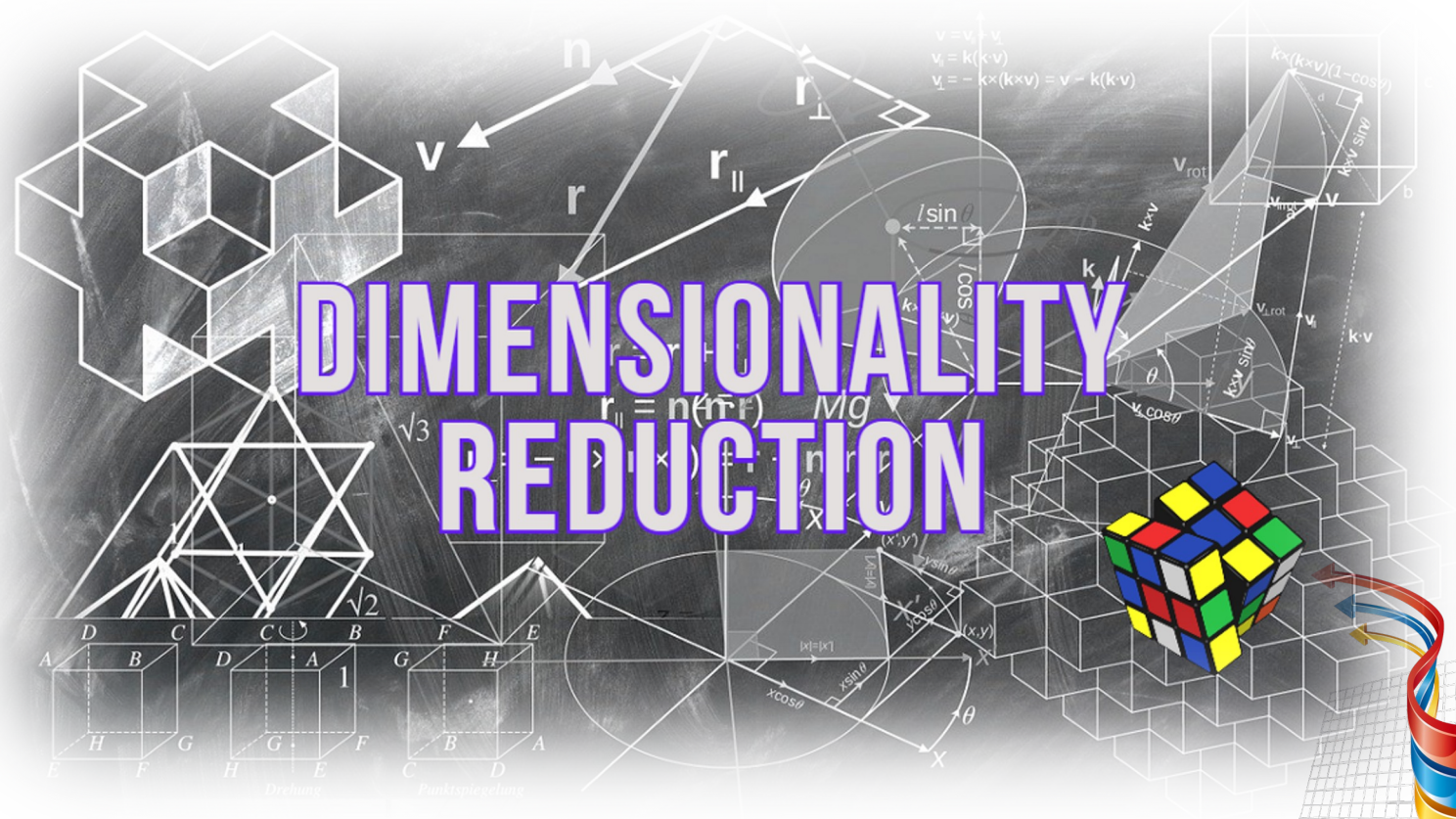
2. ***Clustering algorithm design or selection*****.** Although a wealth of clustering algorithms has been developed, there is no clustering algorithm that can be universally used to solve all problems. Therefore, it is important to carefully investigate the characteristics of the problem at hand, in order to select or design an appropriate clustering strategy.

3. ***Cluster validation*****.** Given a data set, each clustering algorithm can always generate a division, no matter whether the structure exists or not. Effective evaluation standards and criteria are important to provide the users with a degree of confidence for the clustering results derived from the used algorithms. (See [Ref. 1])

4. ***Results interpretation.*** The ultimate goal of clustering is to provide users with meaningful insights from the original data, so that they can effectively solve the problems encountered.

Dimensionality reduction

PCA and t-SNE



The course of dimensionality

The **curse of dimensionality** refers to various phenomena that arise when analysing and organizing data in high-dimensional spaces.

Intuitively, the problem is that when the dimensionality increases, the volume of the space increases so fast that the available **data become sparse**. In order to obtain a statistically sound and a reliable result, the amount of data needed to support the result often grows exponentially with the dimensionality.

- *Combinatorial example*:
  In a combinatorial problem with d binary variables, we have a space composed by 2^d solutions. If we need an extra variable to model an additional constraint, the effort needed to try all the combination doubles (2^{d+1}). This effect is known as **combinatorial explosion,** and it is a form of dimensionality issue.

Machine learning algorithms suffer from a similar problem, each time we add a feature/dimension, we need to add data to describe the interactions of the new features with all the other.

What is dimensionality reduction?

Dimensionality reduction [Ref. 2] is the transformation of data from a high-dimensional space into a low-dimensional space so that the low-dimensional representation retains some meaningful properties of the original data (Wikipedia).

The main idea is to project the d-dimensional points into a k-dimensional space so that:

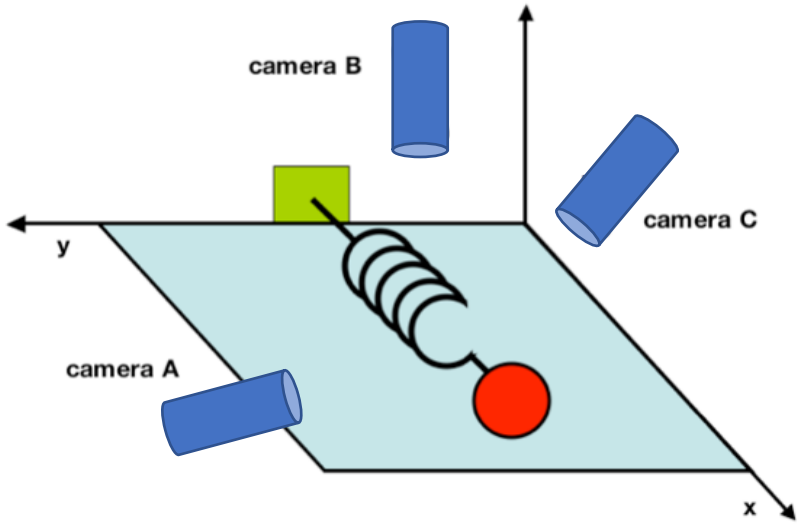
- $\boldsymbol{k \ll d}$
- **Distances are preserved as well as possible.**

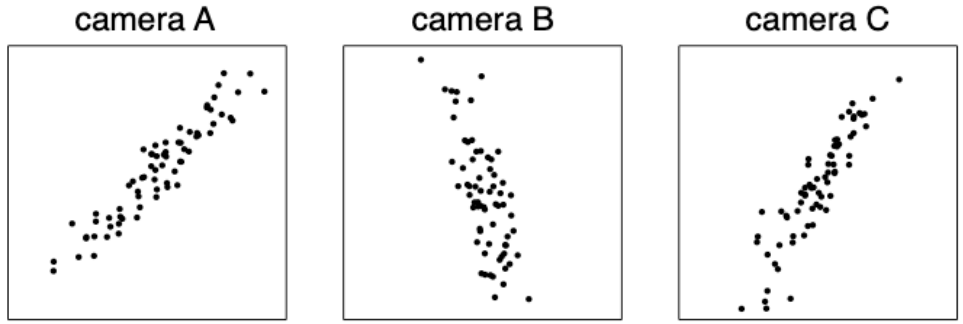
→ If we remove **highly correlated dimensions**, we do not lose any information (lossless compression).

→ If we remove **marginal dimensions** (e.g., dimensions with negligible variance), we should lose only meaningless information (lossy compression).

Principal Component Analysis (PCA)

Jonathon Shlens, “A Tutorial on Principal Component Analysis”; arXiv



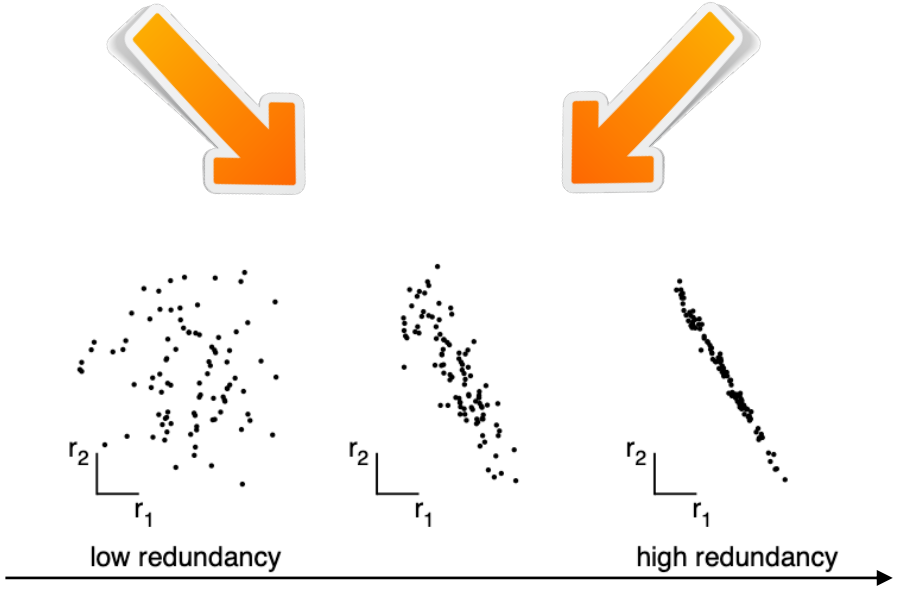


Principal Component Analysis (PCA)

Jonathon Shlens, "A Tutorial on Principal Component Analysis"; arXiv

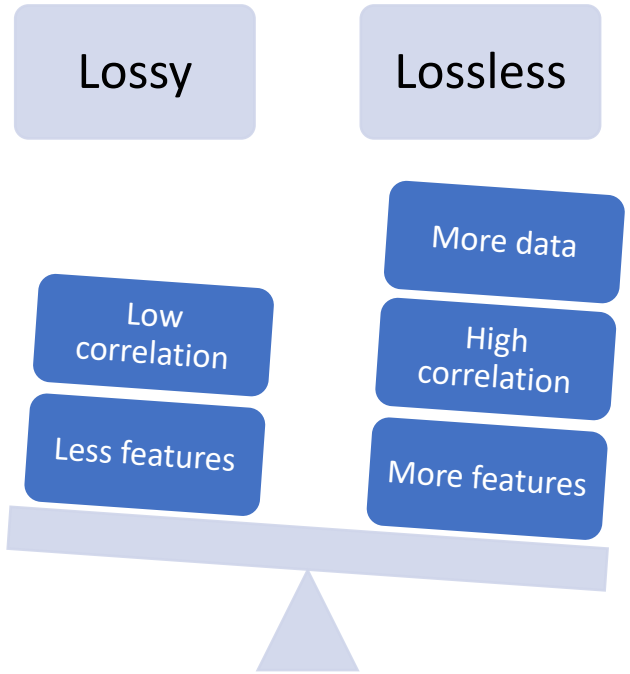
The objective of PCA is to rigidly rotate the axes of the d-dimensional space to new positions (principal axes) that have the following properties:

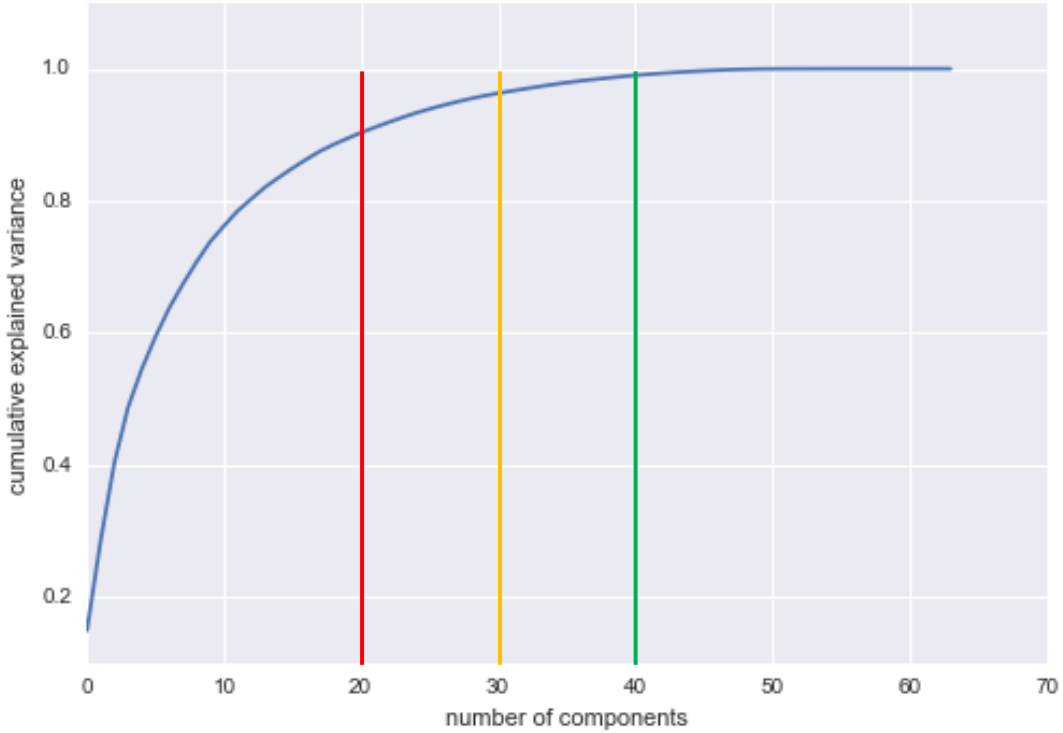
- The axis are ordered such that principal axis 1 has the highest **variance** and the axis d has the lowest variance.
- Covariance among each pair of the principal axes is zero (the principal axes are **uncorrelated**).



PCA: where is the reduction?

→ We must plot the eigenvalues and remove those principal components that have low eigenvalues.





Exe PCA

Notebook

t-distributed Stochastic Neighbor Embedding

T-SNE [Ref. 3] is a **nonlinear dimensionality** reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions.

t-SNE converts affinities of data points to probabilities. The affinities in the original space are represented by **Gaussian joint probabilities** and the affinities in the embedded space are represented by **Student's t-distributions**.

At a high level, t-SNE comprises two stages:

1. It constructs a probability distribution over pairs of high-dimensional objects in such a way that similar objects are assigned a higher probability while dissimilar points are assigned a lower probability.
2. It defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the Kullback–Leibler divergence (KL divergence) between the two distributions with respect to the locations of the points in the map.

t-distributed Stochastic Neighbor Embedding

In the first step, the probability distribution is created over pairs of high-dimensional points with:

$$p_{j|i} = \frac{e^{-\|x_i - x_j\|^2 / 2\sigma_i^2}}{\sum_{k \neq i} e^{-\|x_i - x_k\|^2 / 2\sigma_i^2}}$$

The **similarity of datapoint x_j to datapoint x_i** is the conditional probability that x_i would pick x_j as its neighbour if neighbours were picked in proportion to their probability density under a Gaussian centred at x_i.

In the second step, t-SNE aims to learn a d-dimensional map that reflects the similarity p_{ij} as well as possible. To this end, a **heavy-tailed** Student t-distribution is used:

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{l \neq k} (1 + \|y_k - y_l\|^2)^{-1}}$$

The locations of the points y_i in the map are determined by minimizing the (non-symmetric) Kullback–Leibler divergence of the distribution P from the distribution Q:

$$KL(P \parallel Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Exe t-SNE

Notebook

Clustering

K-Means and Agglomerative

MMSD 2

What is clustering?

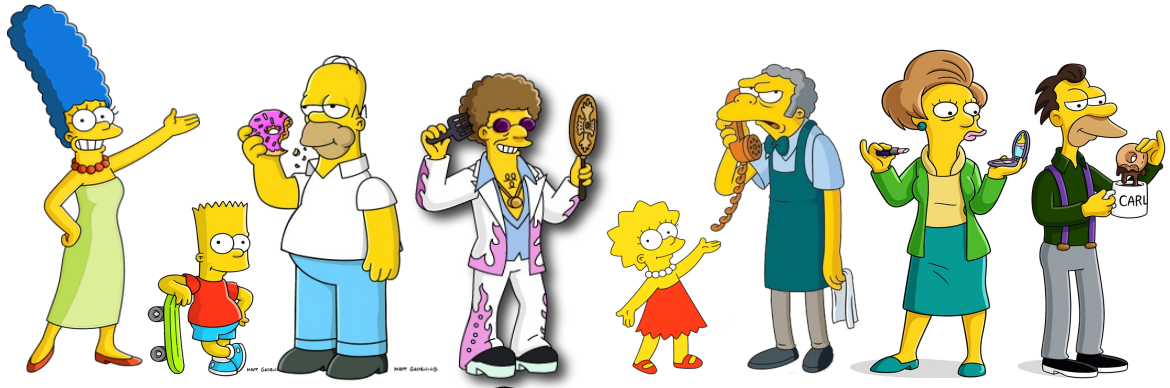
Also called unsupervised learning, clustering aim at organizing data into classes such that there is:

- High intra-class similarity.
- Low inter-class similarity.

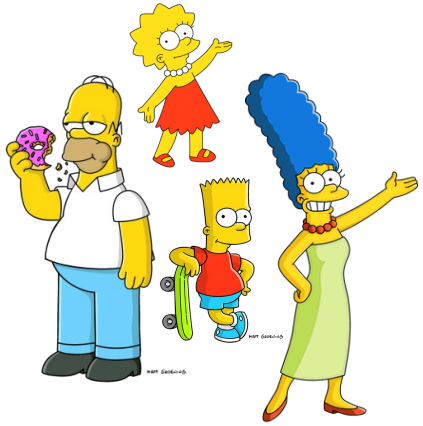
The main difference from supervised learning is that clustering finds the classes and the number of classes directly from the data, **without using any labels.**

→ It might be seen as a form of grouping similar objects.

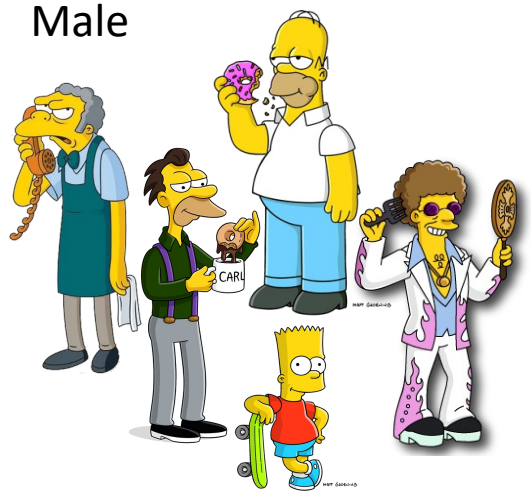
What is a group among these objects?



Simpson vs. non Simpson



Female vs. Male



What is similarity?

Similarity is hard to define, but... "*We know it when we see it*".

In Machine Learning, we define the similarity as the inverse of the distance.

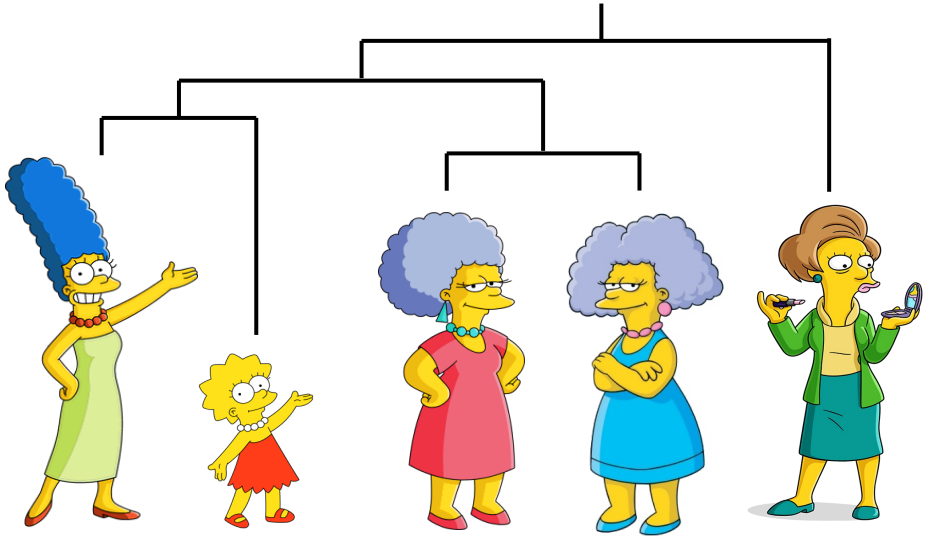
Distance properties:

- Symmetry:
  - $D(A,B) = D(B,A)$
- Self-similarity:
  - $D(A,A) = 0$
- Positivity:
  - $D(A,B) = 0\ IIf\ A = B$
- Triangular inequality:
  - $D(A,B) \leq D(A,C) + D(C,B)$

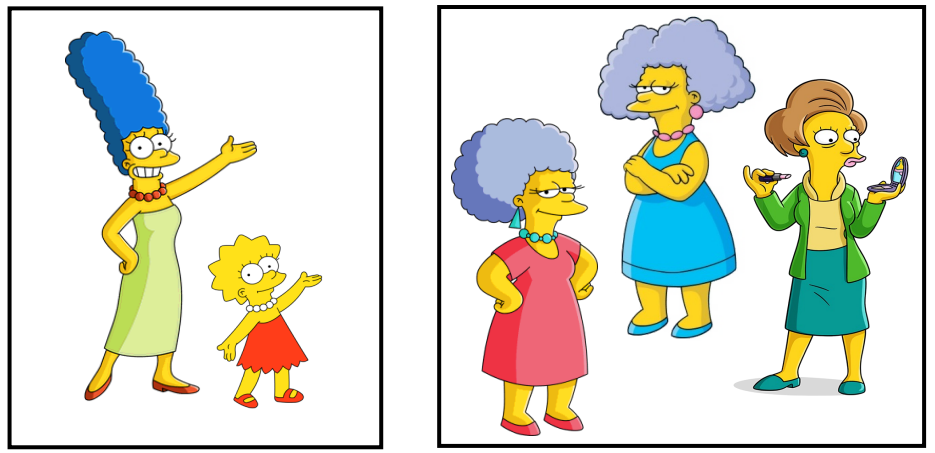
Two types of clustering

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion.
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion.

Hierarchical



Partitional



K-means

The K-Means algorithm clusters data by trying to separate samples in **K groups** of equal variance, minimizing a criterion known as the ***inertia*** or within-cluster sum-of-squares.

$$inertia = \sum_{i=1}^{n} \min_{j \in \{0, 1, .., k-1\}} (||x_i - \mu_j||^2)$$

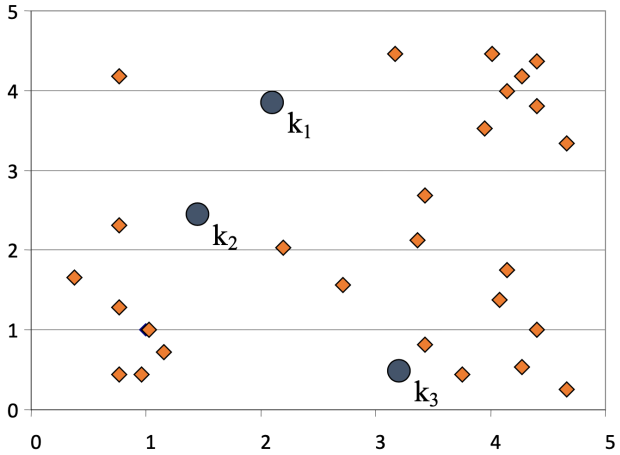
This algorithm requires **the number of clusters** to be specified.

The algorithm is simple:

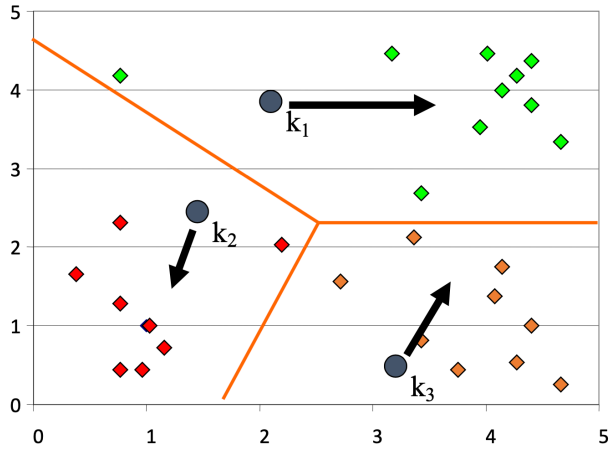
1. Decide the number of clusters K.
2. Initialize the K cluster centers.
3. Decide the objects membership by assigning each to the nearest center.
4. Re-estimate the center of each cluster by using the objects assigned to that cluster.
5. If none of the objects change membership, then stop. Otherwise go to step 3.

K-means

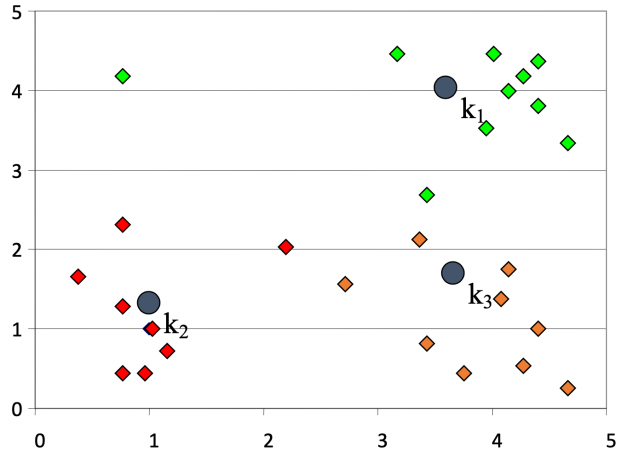
Step 1: Random initialization of the K cluster centers.



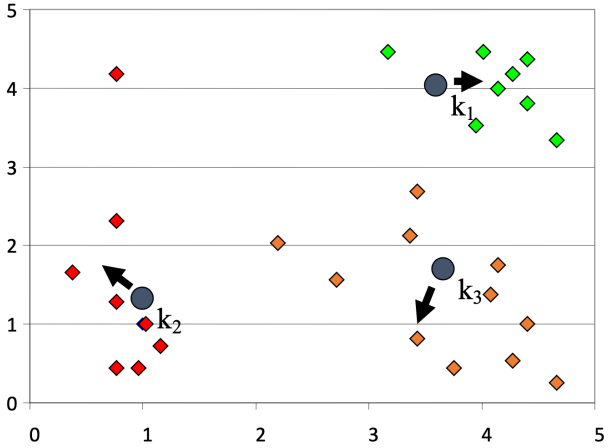
Step 2: Create clusters by assigning points to the nearest center.
We need a distance measure.



Step 3: Re-estimate the center of each cluster as the mean of the points in the cluster.



Step 4: Reconstruct the clusters with the new centers and check if some points have changed membership.



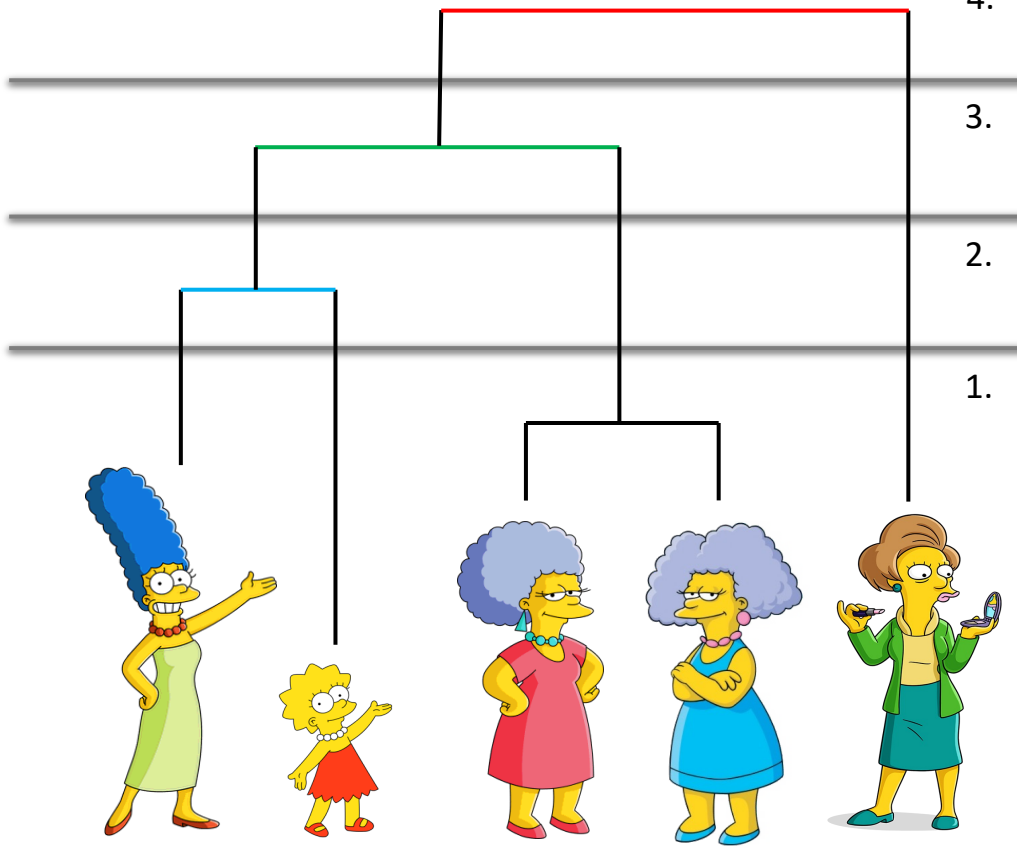
Exe K-means

Notebook

Agglomerative clustering

The agglomerative clustering, also known as the Bottom-Up approach, starts by placing each object in a cluster.
Then, it evaluates all the couples of clusters to find the best one to merge.
This process is repeated until we are left with a single cluster.

→ In this type of clustering, we need to define also how to measure the distance between one object and a cluster, and between two clusters.



Agglomerative clustering

Linkage metrics:

- **Single linkage**: the distance between two clusters is determined by the distance of the two closest objects (nearest neighbours) in the clusters.
- **Maximum linkage**: as single linkage but it uses the maximum.
- **Group average distance**: the distance between two clusters is the average distance between all pairs of objects in the two clusters.
- **Wards linkage**: minimize the variance of the merged clusters.

Exe Agglomerative

Notebook

References

1. Rui Xu and D. Wunsch. 2005. "Survey of clustering algorithms”. In *IEEE Transactions on Neural Networks*.
2. Van Der Maaten, Laurens, Eric Postma, and Jaap Van den Herik. 2009. "Dimensionality reduction: a comparative review.”
3. Van der Maaten, Laurens, and Geoffrey Hinton. 2008. "Visualizing data using t-SNE." *Journal of machine learning research*.