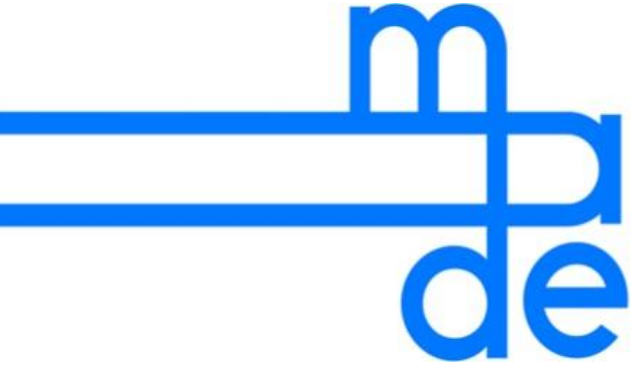


girafe ai

Machine Learning, Lecture 11

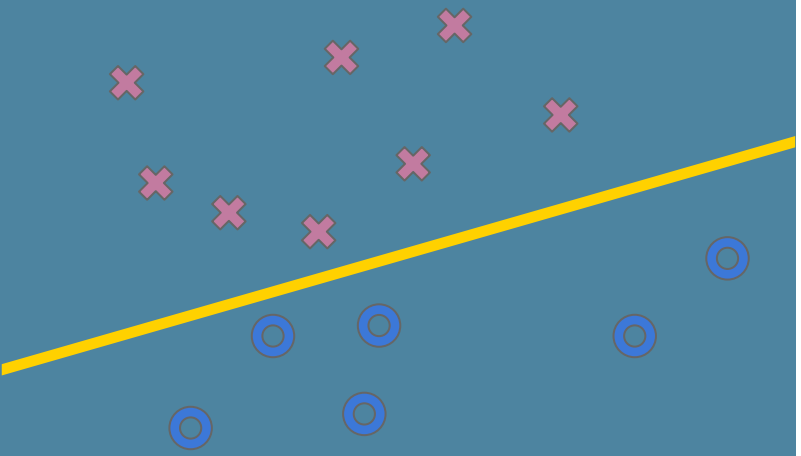
# Unsupervised Learning

Radoslav Neychev, Fall 2021

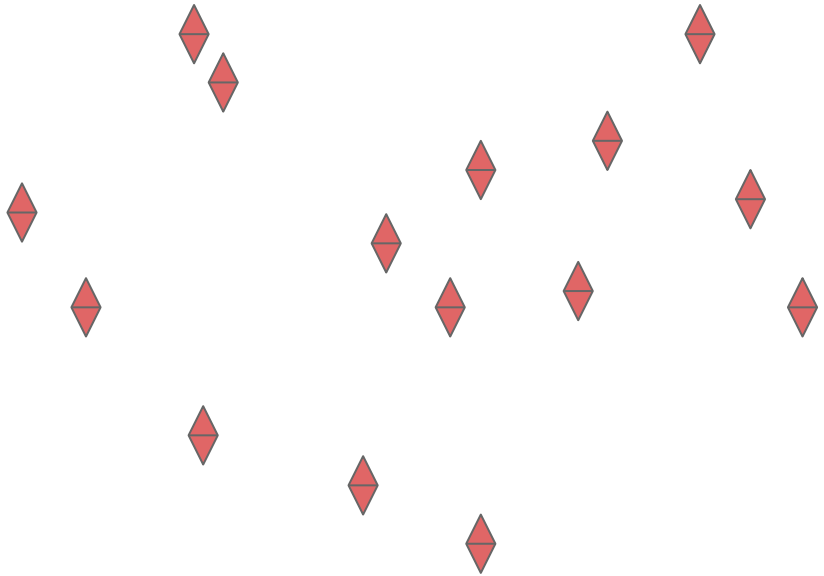
# Outline

1. Geometrical machine learning
    a. Dimensionality curse
    b. Manifold assumption
2. Dimensionality reduction
    a. Feature selection
    b. Multidimensional Scaling (MDS)
    c. Isomap
    d. Locally linear embedding (LLE)
    e. t-SNE
3. Clustering
    a. k-means
    b. DBSCAN
    c. Hierarchical clustering
    d. metrics
4. Density estimation
    a. Kernel density estimation

# Supervised learning



# Unsupervised learning

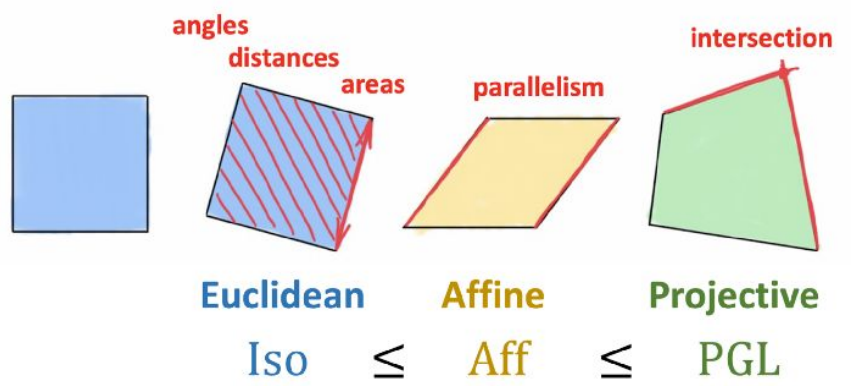


# Geometrical machine learning

girafe ai

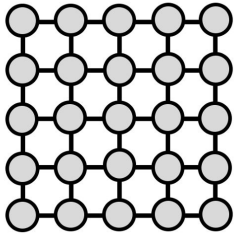
01

# Geometrical machine learning

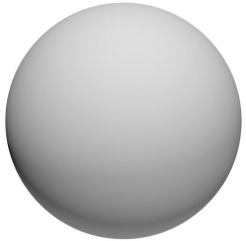



The breakthrough insight of Klein was to approach the definition of geometry as the **study of invariants**, or in other words, structures that are preserved under a certain type of transformations (symmetries)

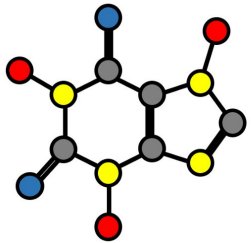
[Article introducing a book on Geometric Deep Learning]



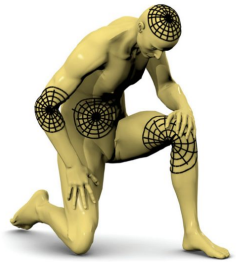
Grids



Groups



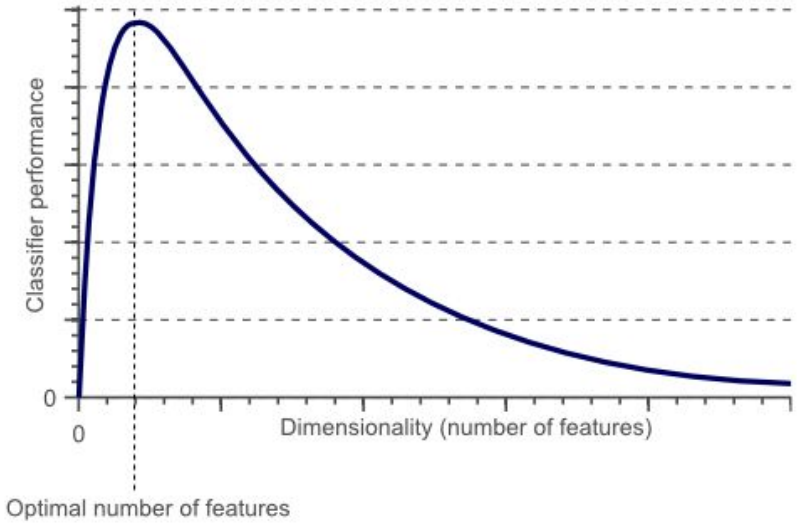
Graphs



Geodesics & Gauges

# Dimensionality curse

Certain behaviours or effects that appear when analysing data in high dimensions, that do not occur in low-dimensional spaces



# Sphere volume decrease

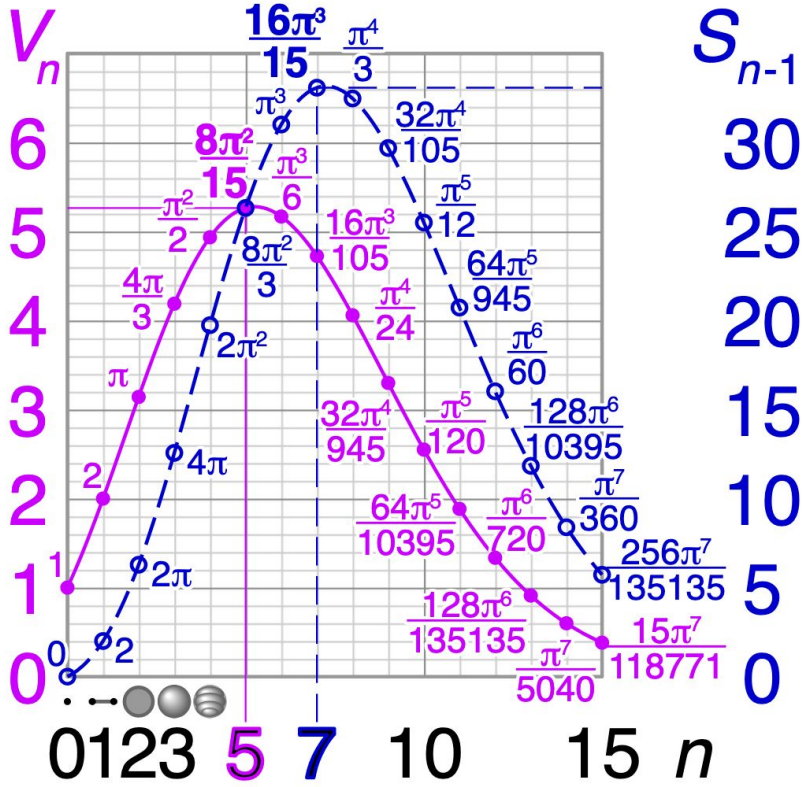
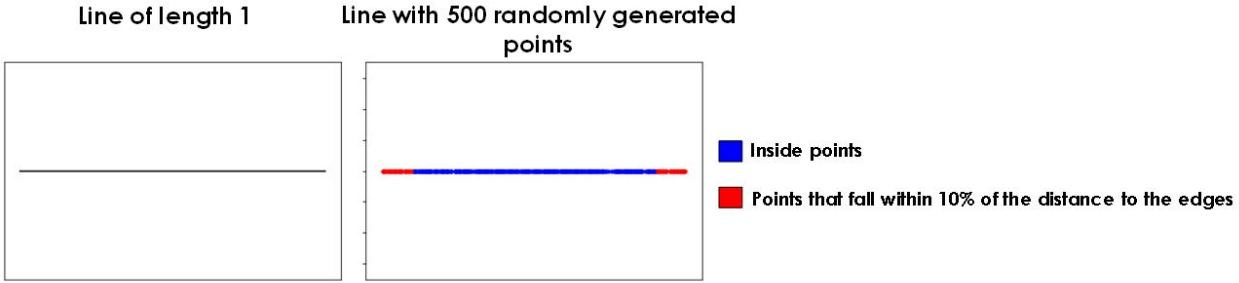
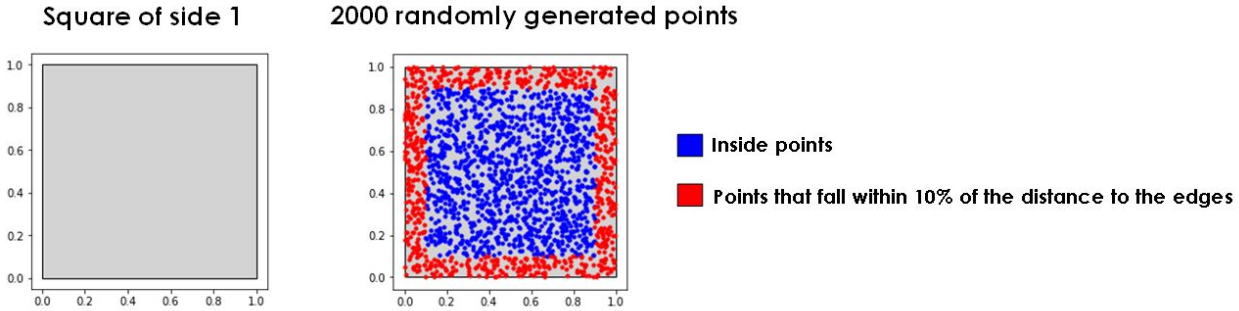



image source

# Distance in high dimensional space




Ratio of inside points to total points = 80%
Average Distance between 2 points = 0.34

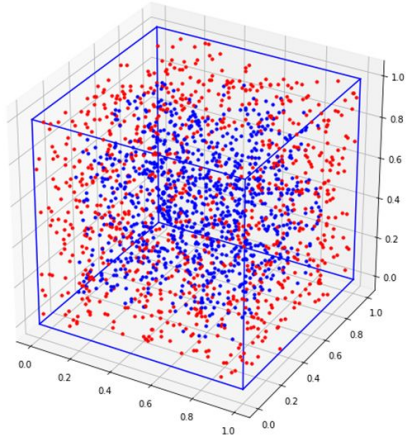



Ratio of inside points to total points = 63%
Average Distance between 2 points = 0.52

image source

# Distance in high dimensional space

Cube of side 1




Inside points

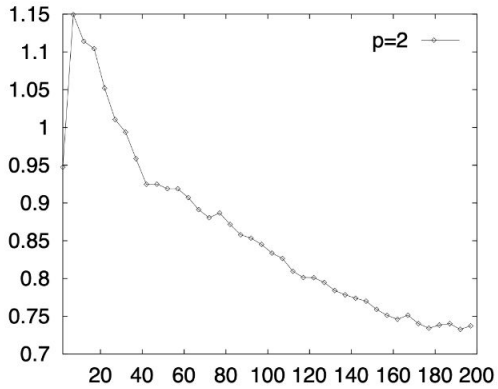
Points that fall within 10% of the distance to the edges

Ratio of inside points to total points = 51%
Average Distance between 2 points = 0.65

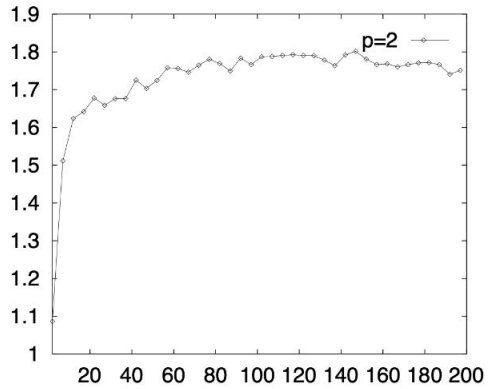
| N° of dimensions | % Outside Points | Average distance (A,B) |
|---|---|---|
| 1 | 20% | 0.34 |
| 2 | 37% | 0.52 |
| 3 | 49% | 0.65 |

# Distance relative contrast

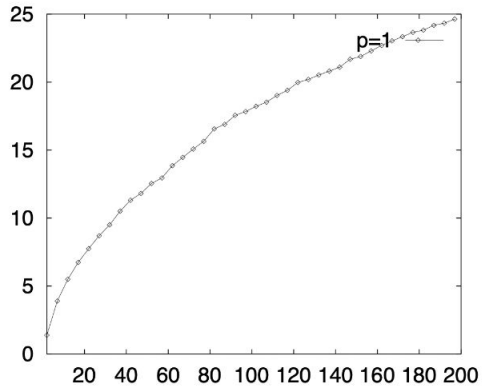
Take random points uniformly distributed in D dimensional cube and calculate distance to the farthest point and to the closest point. Plot their difference depending on D for different Minkowski metrics




L3




L2




L1

[On the Surprising Behavior of Distance Metric in High-Dimensional Space, Aggarwal et al., 2002]

# Conclusions

- Distance loses its meaning - closest and farthest points are equally far
- Proximity concept becomes ill defined
- Lower powers of Minkowski metrics are more sustainable to dimensionality curse

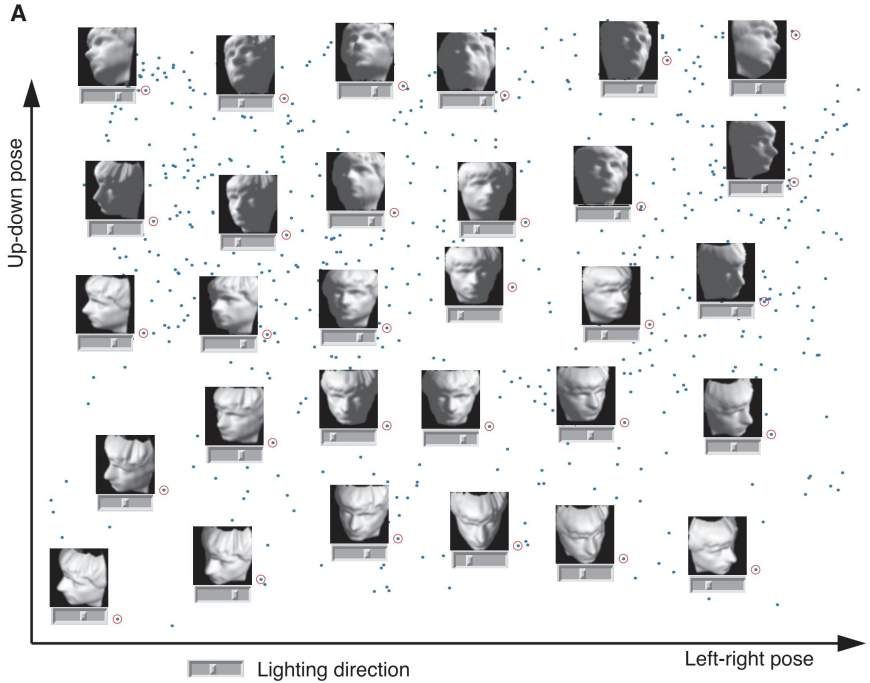
| N° of dimensions | % Outside Points | Average distance (A,B) |
|---|---|---|
| 1 | 20% | 0.34 |
| 2 | 37% | 0.52 |
| 3 | 49% | 0.65 |

# Manifold assumption

The data lie approximately on a surface (called manifold) of usually much lower dimension than the input space

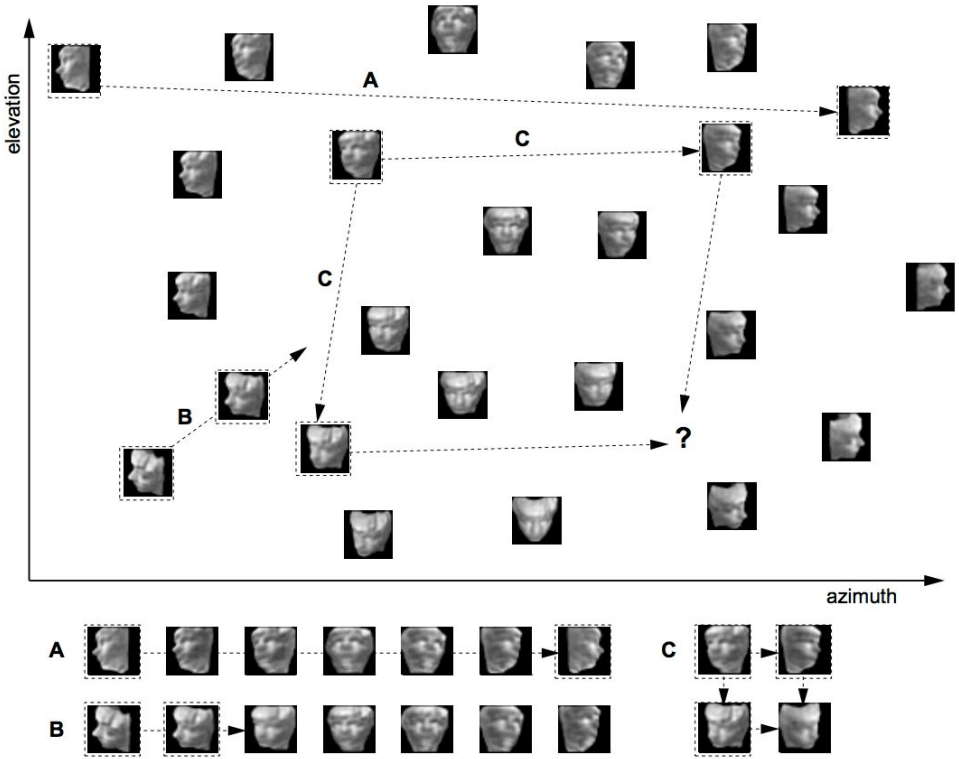
So problem dimensionality could be (non-)linearly reduced or other tasks solved

Sometimes dimensionality of manifold is referred as intrinsic dimension (see this article)




Tenenbaum, de Silva, Langford
A Global Geometric Framework for Nonlinear Dimensionality Reduction

# Latent space




Latent (embedding) space describes data in coordinates more relevant to humans' reason and often allows useful linear operations:

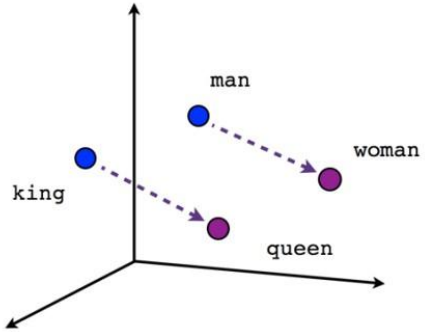
- Interpolation (A)
- Extrapolation (B)
- Analogy (C)

This process is also called embedding space 'walking'

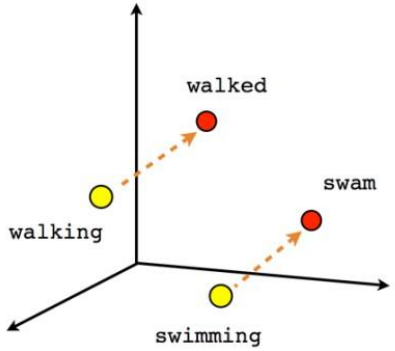
# Latent space example

Word2vec is a method to embed words from text corpus into linear space

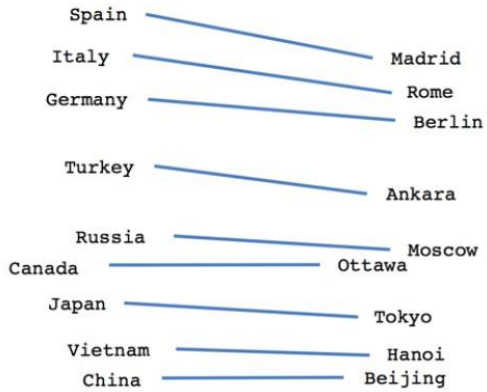
Read more: [manifold assumption], [assessing assumption]




Male-Female




Verb tense




Country-Capital

# Dimensionality reduction

girafe ai

02

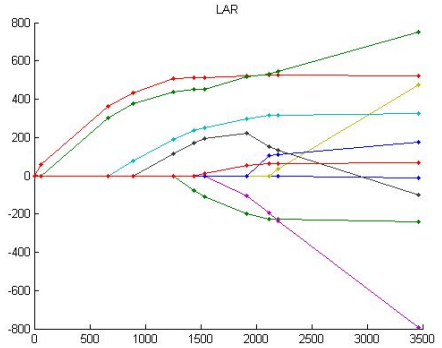
# Feature selection

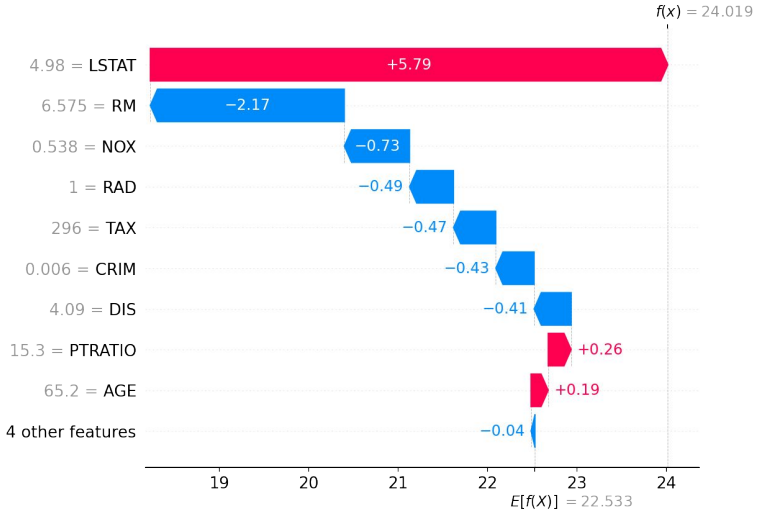
Select subset of existing features to use in further modelling

Usually is based on some supervised method

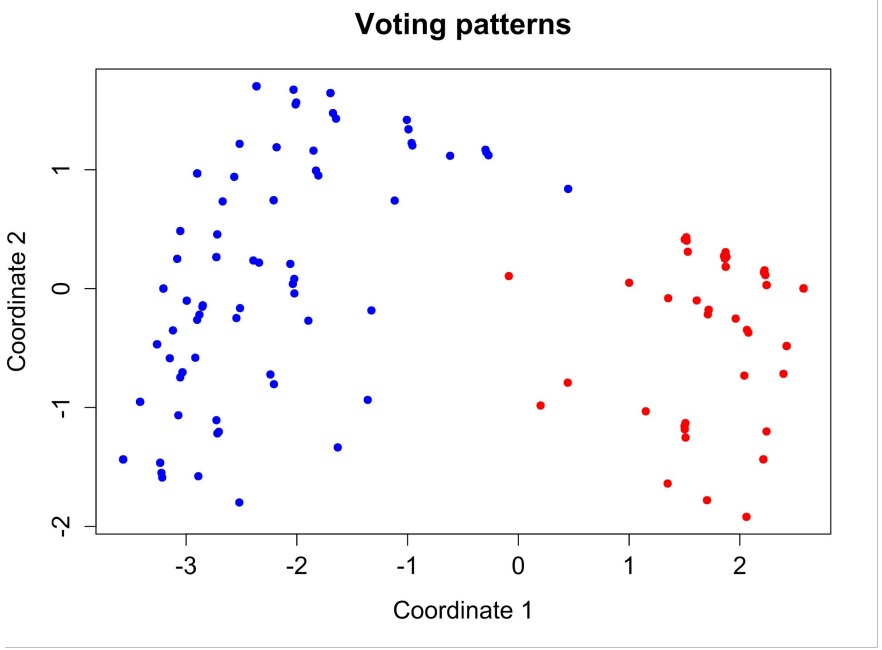
Examples:

- stepwise regression
- LARS
- SHAP values
- etc...



# Multidimensional Scaling (MDS)




Voting patterns in the United States House of Representatives

Goal:
Linearly embed to given lower space

Solution:
PCA

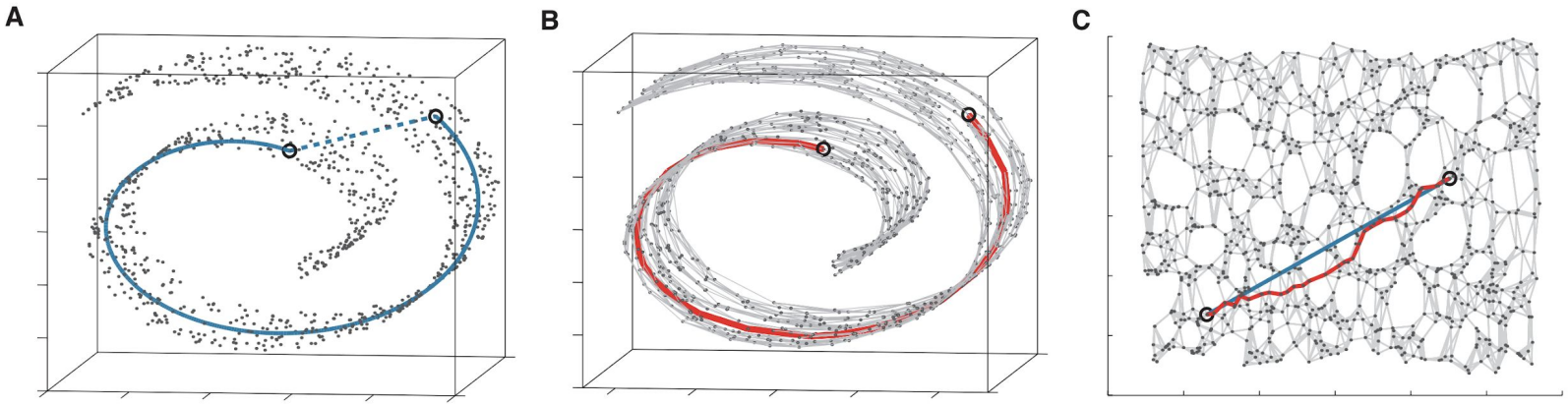
$$L = ||D_x - D_y||_2 \rightarrow \min_{y=Ax}$$

$$y = \Lambda^{1/2} V^T$$

Params: p - target dimensionality

Also could be non-linear

# Isomap




Now make distancies geodesic!
And measure distances on the produced graph

[A Global Geometric Framework for Nonlinear Dimensionality Reduction, Tenenbaum et al., Science, 2002.]

Params:
n - number of neighbours to connect
p - dimensionality of manifold

# Isomap algorithm

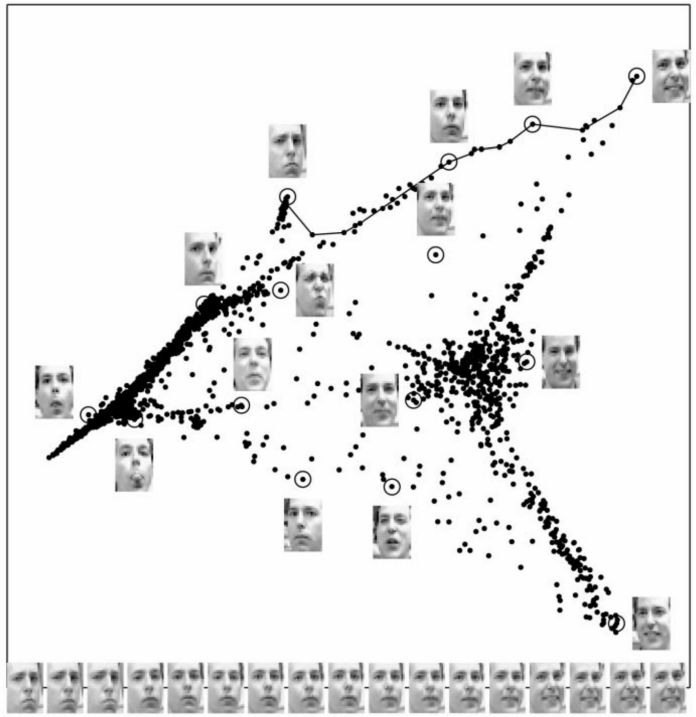
| Step | | |
|---|---|---|
| 1 | Construct neighborhood graph | Define the graph $G$ over all data points by connecting points $i$ and $j$ if [as measured by $d_X(i,j)$] they are closer than $\epsilon$ ($\epsilon$-Isomap), or if $i$ is one of the $K$ nearest neighbors of $j$ ($K$-Isomap). Set edge lengths equal to $d_X(i,j)$. |
| 2 | Compute shortest paths | Initialize $d_G(i,j) = d_X(i,j)$ if $i,j$ are linked by an edge; $d_G(i,j) = \infty$ otherwise. Then for each value of $k = 1, 2, \ldots, N$ in turn, replace all entries $d_G(i,j)$ by $\min\{d_G(i,j), d_G(i,k) + d_G(k,j)\}$. The matrix of final values $D_G = \{d_G(i,j)\}$ will contain the shortest path distances between all pairs of points in $G$ (*16*, *19*). |
| 3 | Construct $d$-dimensional embedding | Let $\lambda_p$ be the $p$-th eigenvalue (in decreasing order) of the matrix $\tau(D_G)$ (*17*), and $v_p^i$ be the $i$-th component of the $p$-th eigenvector. Then set the $p$-th component of the $d$-dimensional coordinate vector $\mathbf{y}_i$ equal to $\sqrt{\lambda_p} v_p^i$. |

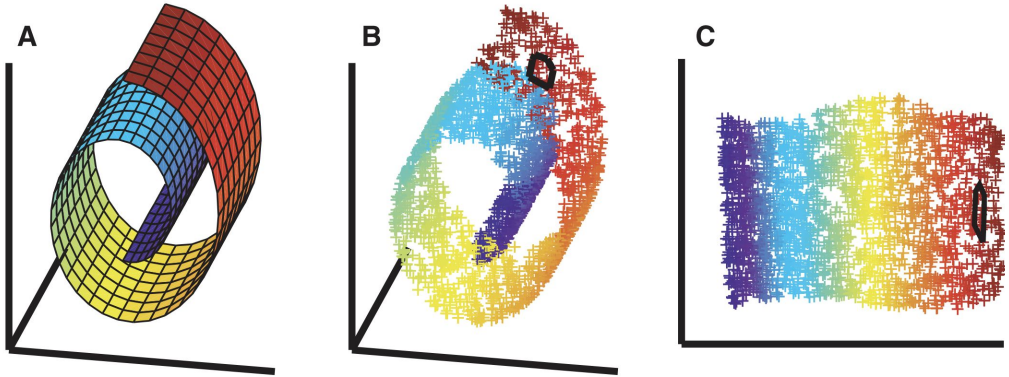
17. The operator $\tau$ is defined by $\tau(D) = -HSH/2$, where $S$ is the matrix of squared distances $\{S_{ij} = D_{ij}^2\}$, and $H$ is the "centering matrix" $\{H_{ij} = \delta_{ij} - 1/N\}$ (*13*).

# Locally linear embedding (LLE)

Smooth manifold locally approximated with hyperplane. Linear pieces are stitched together.

[Nonlinear Dimensionality Reduction by Locally Linear Embedding, Roweis et al., Science, 2000]





# LLE algorithm

1. estimate point by its K neighbours

$$\varepsilon(W) = \sum_{i=1}^{n} ||x_i - \sum_{j=1}^{K} W_{ij} x_j||^2$$

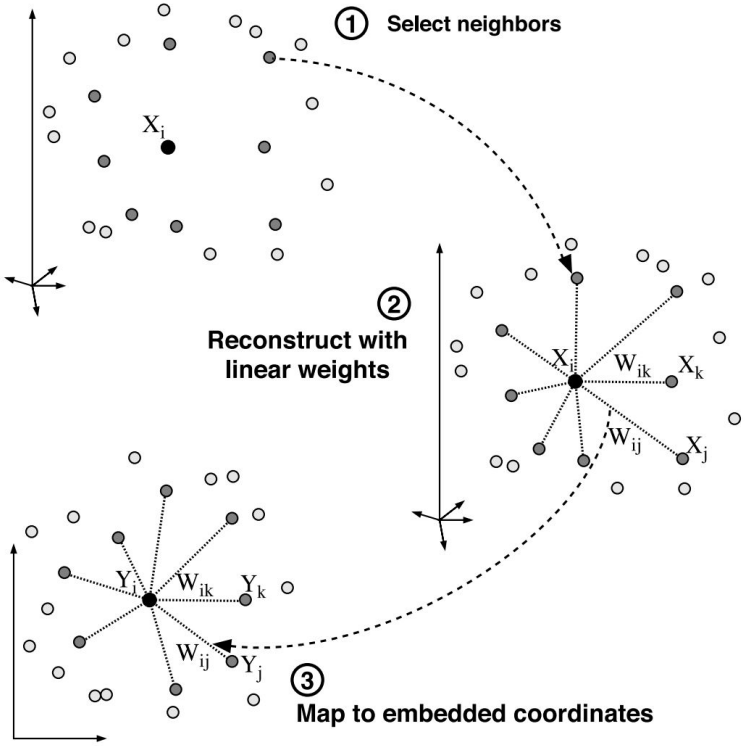
2. Estimate new points based on known relations

$$\Phi(Y) = \sum_{i=1}^{n} ||y_i - \sum_{j=1}^{n} W_{ij} y_j||^2$$

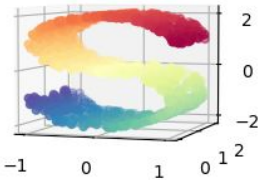
Params:

n - number of neighbours to connec

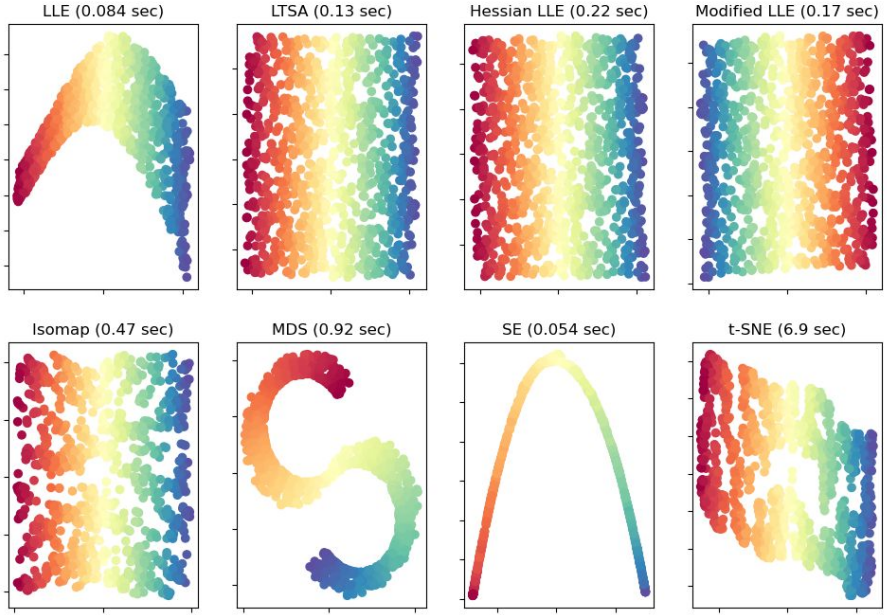
p - dimensionality of manifold



# Many more




Manifold Learning with 1000 points, 10 neighbors

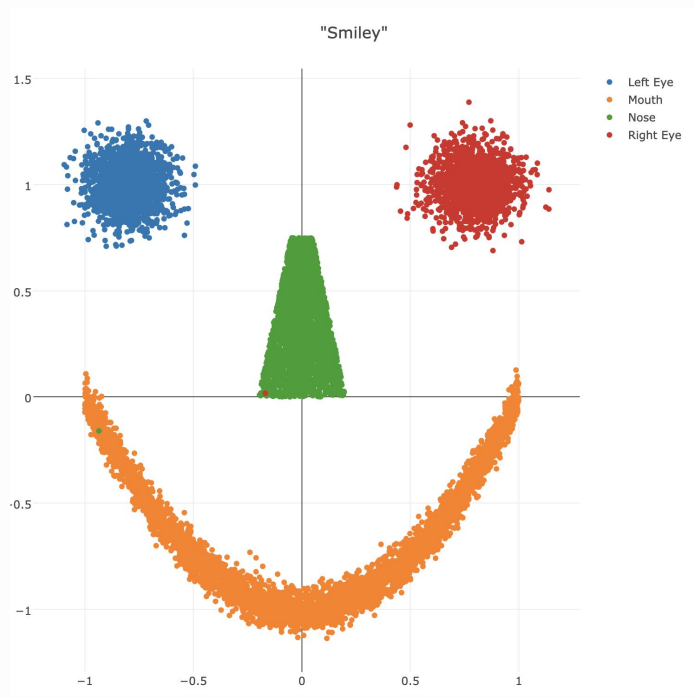



- Hessian Eigenmapping
- Spectral Embedding
- Local Tangent Space Alignment
- Riemannian Geometry
- UMAP
- .....

Read more:

[sklearn manifold methods]
[sklearn signals decomposition]

# Next level: Neural Networks





[UMAP vs Ivis embeddings]

# t-SNE

## t-distributed Stochastic Neighbor Embedding

SNE

Stochastic Neighbor Embedding, Hinton et al., NIPS, 2002

# Stochastic Neighbor Embedding

Convert pairwise distances to probabilities, preserve probabilities through the spaces

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

asymmetric probability
of object i chooses j as its neighbour

$$q_{j|i} = \frac{\exp(-||y_i - y_j||^2)}{\sum_{k \neq i} \exp(-||y_i - y_k||^2)}$$

the same in target space

Let's construct embedding s.t. this distributions are close.

*What are close distributions?*

# Kullback–Leibler divergence

$$D_{KL}(P \parallel Q) = \sum_{i,j} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$


Suspiciously similar to Shannon entropy

Learn more

# SNE problem

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

$$q_{j|i} = \frac{\exp(-||y_i - y_j||^2)}{\sum_{k \neq i} \exp(-||y_i - y_k||^2)}$$

$$D_{KL}(P \,||\, Q) \to \min_Y$$

# t-distributed SNE

Patches over SNE:

1. choose common variance
2. make distributions symmetric

$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq l} \exp\left(-\|x_k - x_l\|^2 / 2\sigma^2\right)}$$

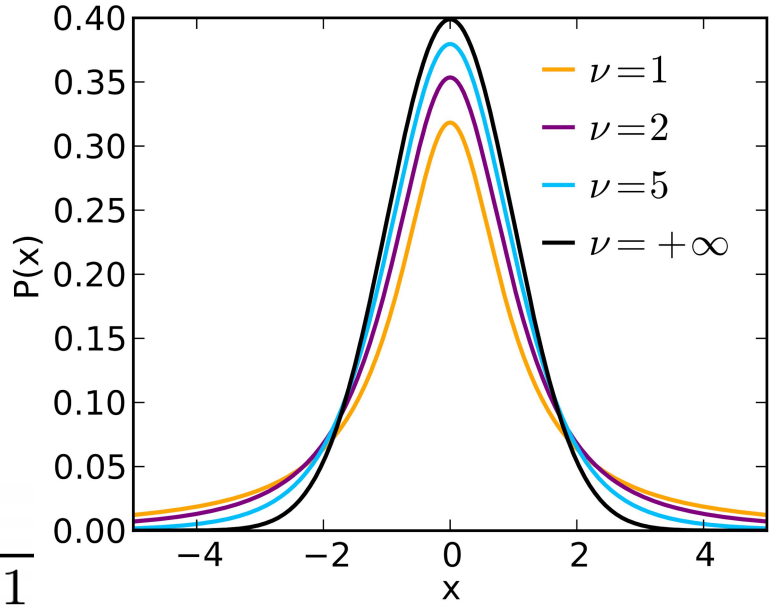
$$p^s_{ij} = \frac{p_{ij} + p_{ji}}{2N}$$

Visualizing Data using t-SNE, Maaten, Hinton, 2008, JMLR

# t-distributed SNE

Patches over SNE:

1. choose common variance
2. make distributions symmetric
3. make it decrease faster than Gaussian (use Student's t-distribution)




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

# t-SNE problem

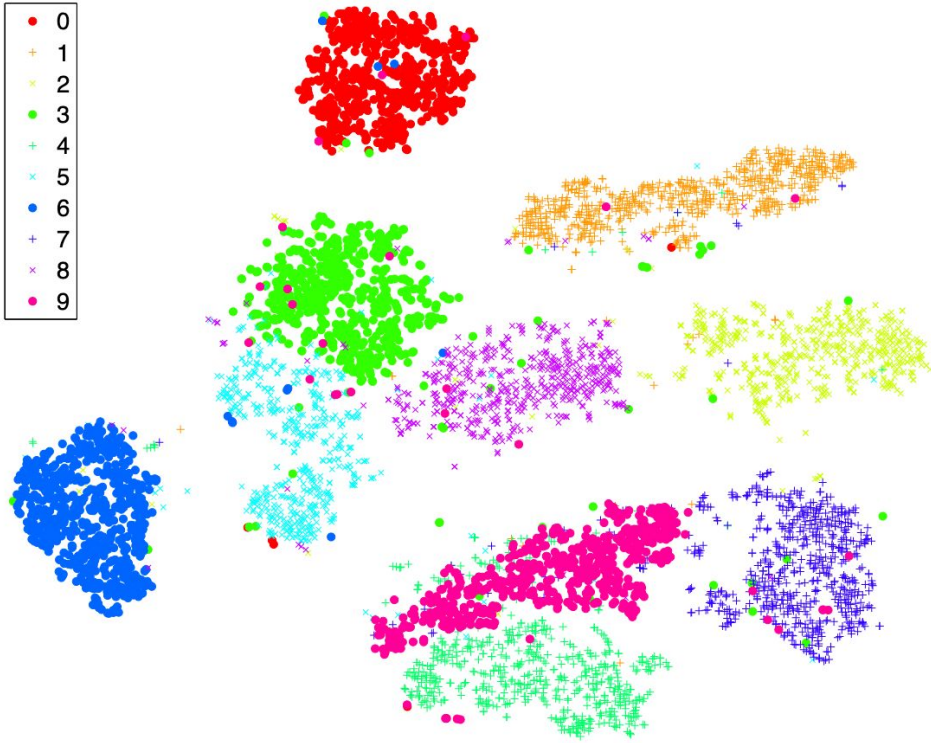
$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq l} \exp\left(-\|x_k - x_l\|^2 / 2\sigma^2\right)} \qquad p_{ij}^s = \frac{p_{ij} + p_{ji}}{2N}$$

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

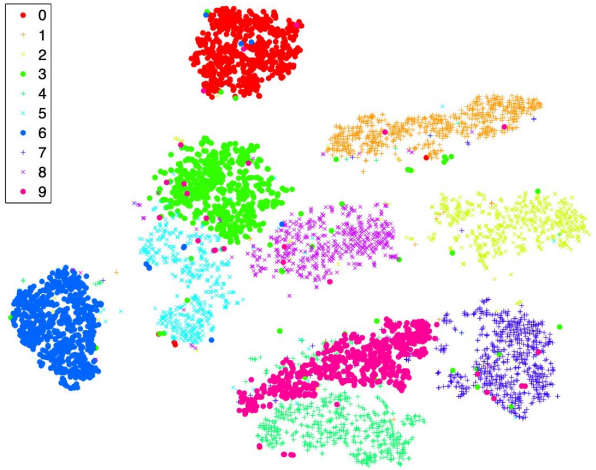
$$D_{KL}(P \;||\; Q) \to \min_Y$$

# Result: nice and light visualizations

t-SNE on MNIST dataset

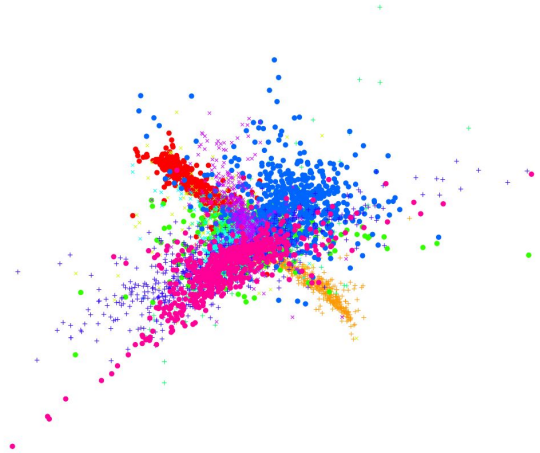


# Comparison




t-SNE

Isomap



LLE

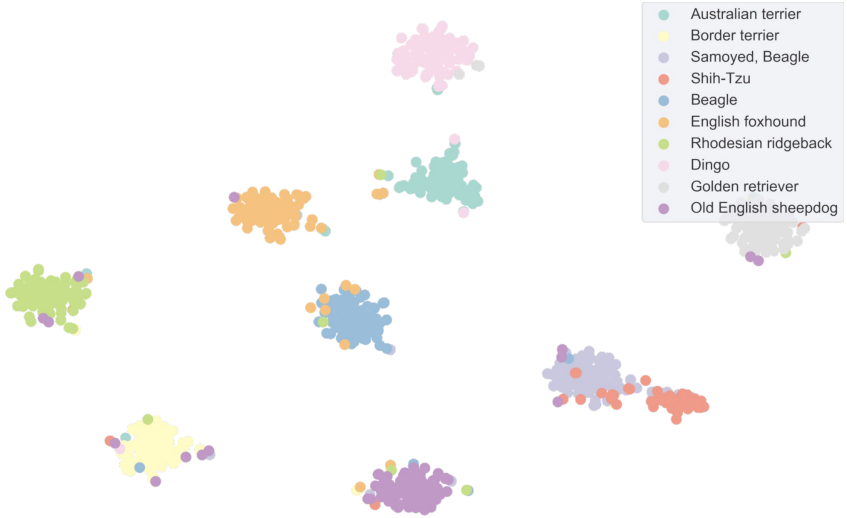
# Real life example



MDS (PCA) on faces embeddings

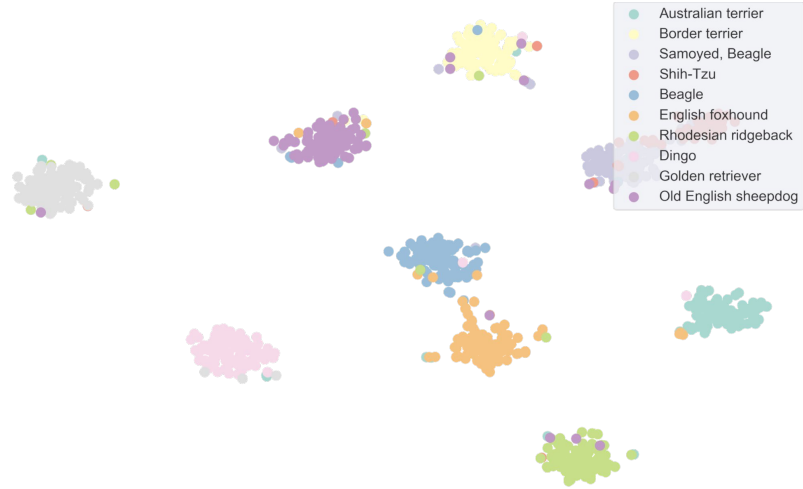
image source

# Real life example




t-SNE on ArcFace

t-SNE on CosFace

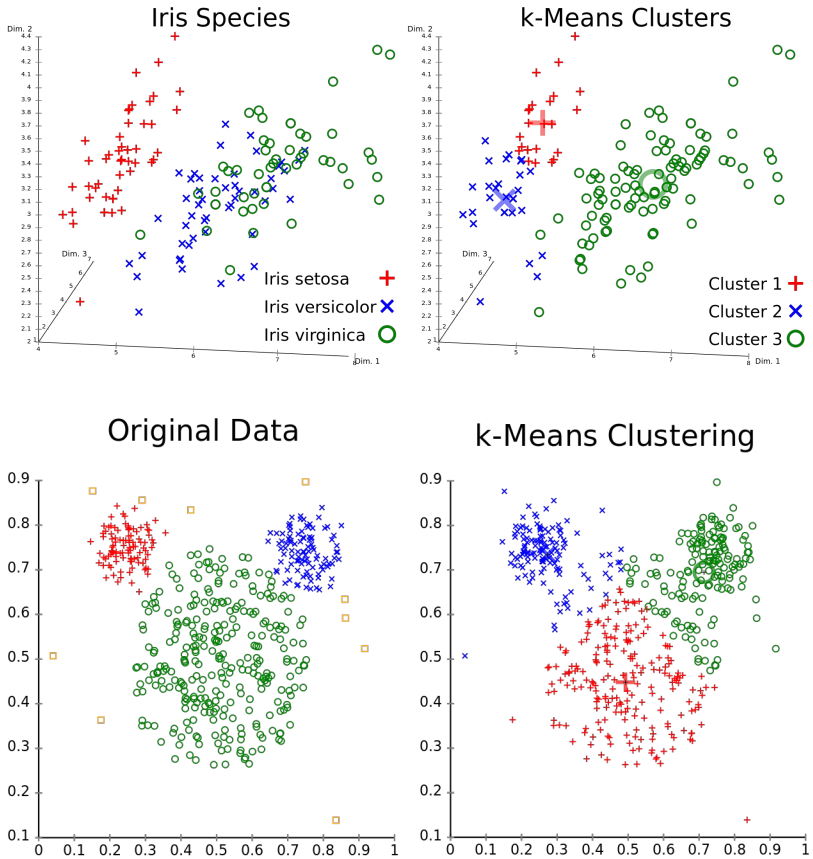


# Clustering

girafe
ai

03

# k-means




Based on K nearest neighbours algorithm

1. Init clusters centers randomly
2. Define current cluster of an object as a nearest center
3. Calculate new cluster center as a mean of all objects in cluster
4. Repeat from p. 2 until convergence

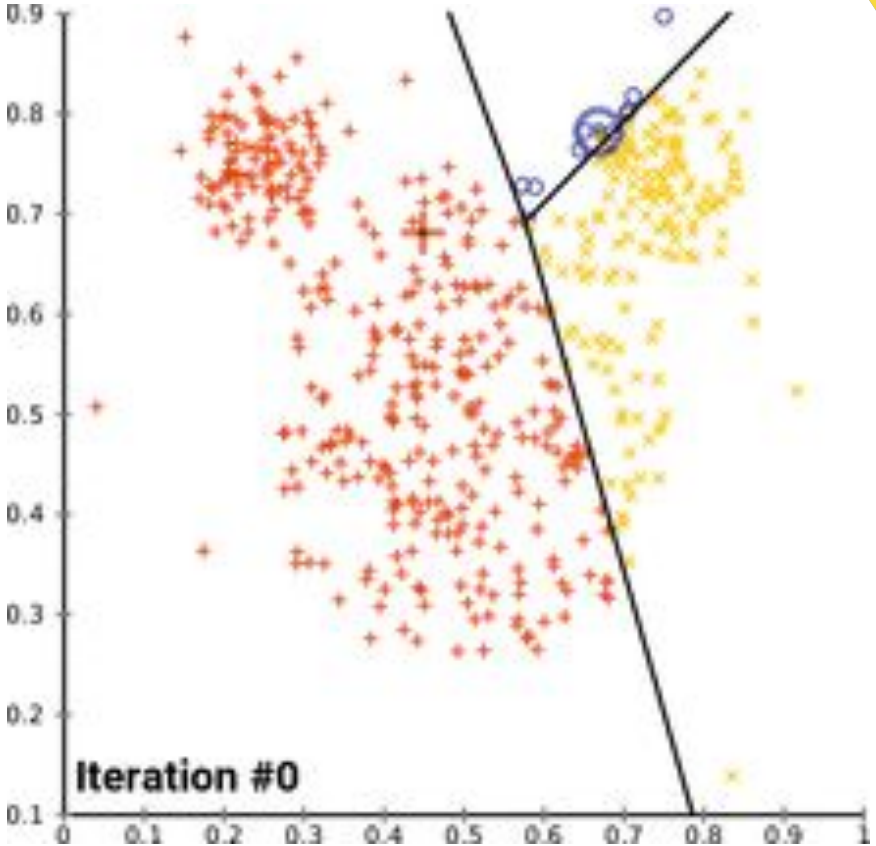
# k-means

Params:

k - number of clusters

Advanced version: k-means++

[The Advantages of Careful Seeding, Arthur, Vassilvitskii, 2007, ACM-SIAM SODA]



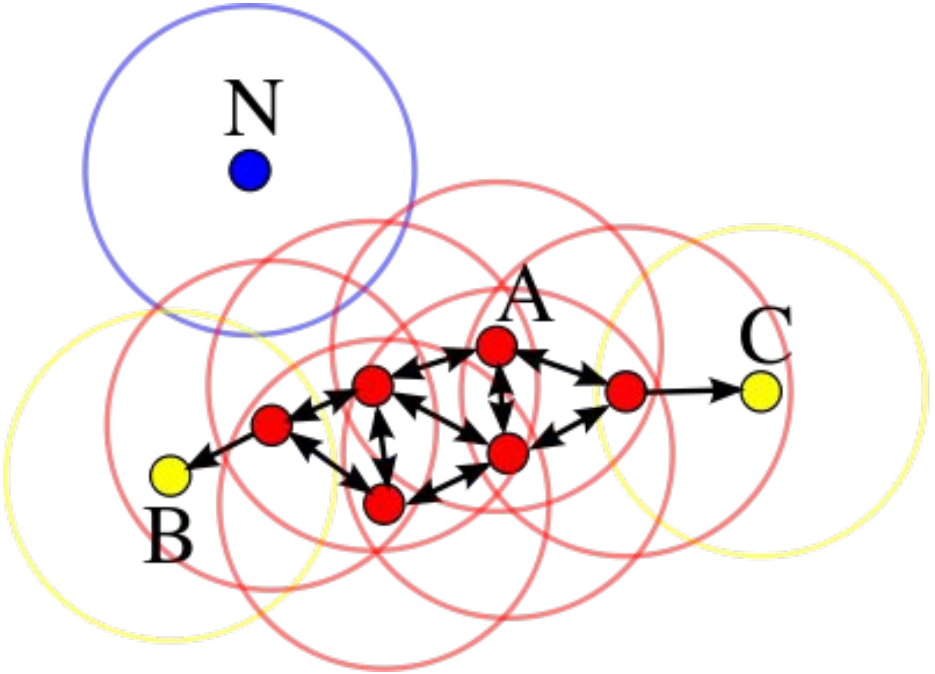
# DBSCAN

Density-Based Spatial Clustering of Applications with Noise

Split all data points into 3 groups:

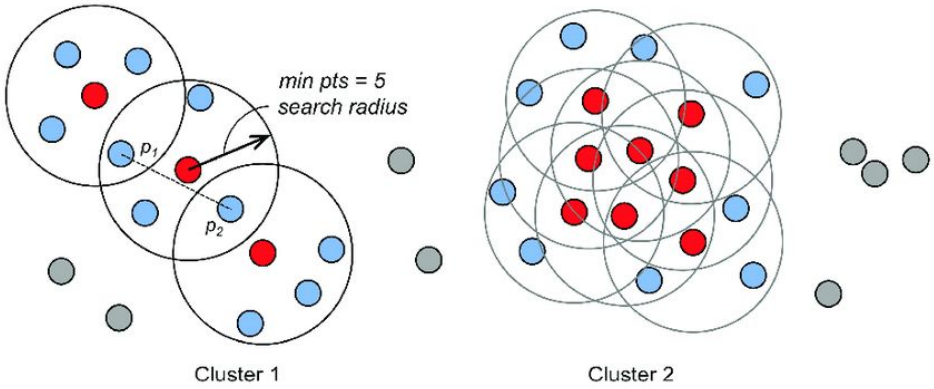
1. Core (red)
2. Border (yellow)
3. Noise (blue)

Core point has at least k other points in **ε**-neighbourhood




[A density-based algorithm for discovering clusters in large spatial databases with noise, Ester et al., 1996, KDD-96]

# DBSCAN




Any two core or border points in **ε**-neighbourhood noted as connected points

Two points both connected to a common point also defined connected (transitivity)

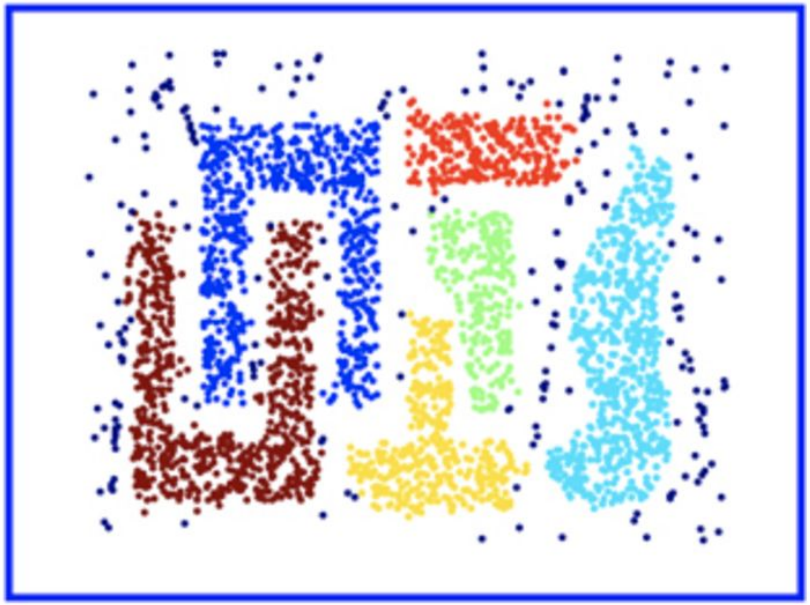
Cluster is defined as maximum connected set of points

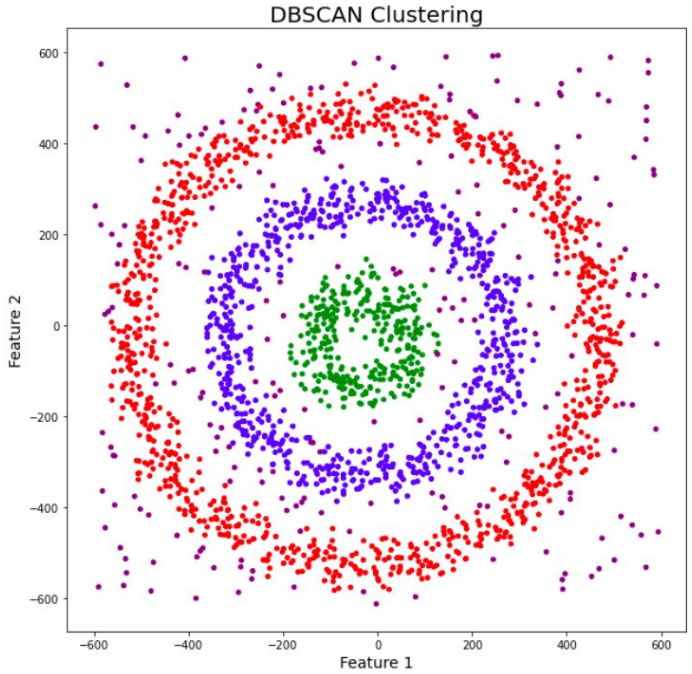
Params:

**ε** - radius of neighbourhood

k - minimal number of neighbours of core point

# DBSCAN examples

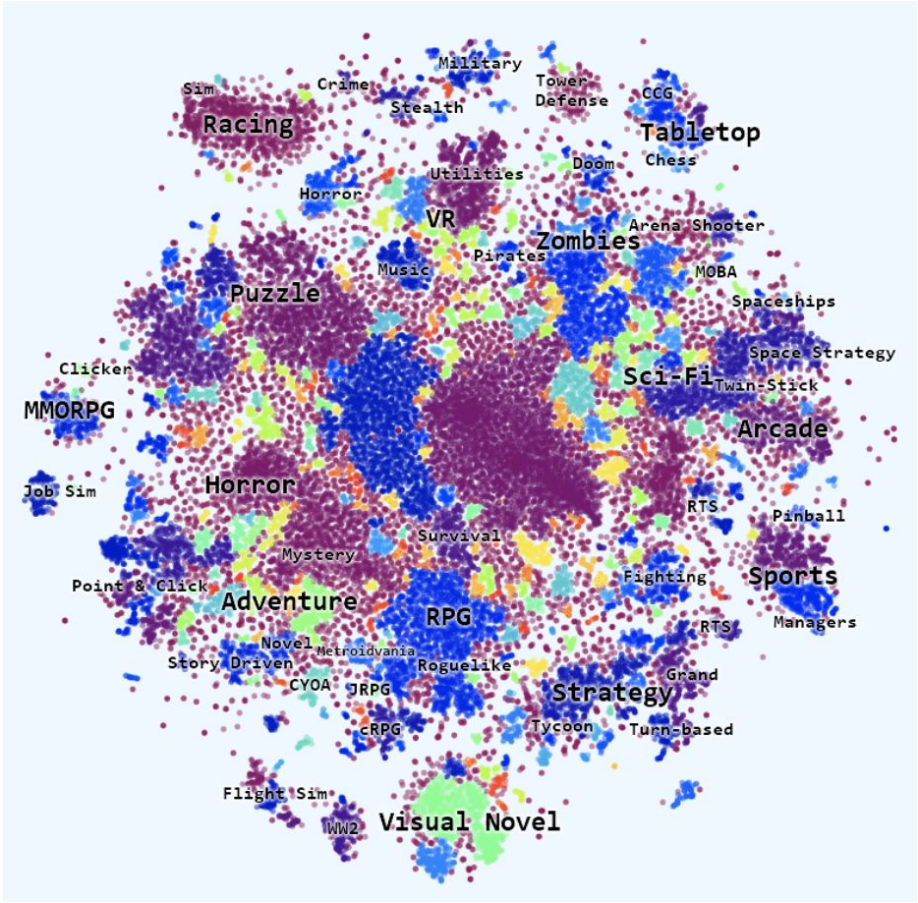




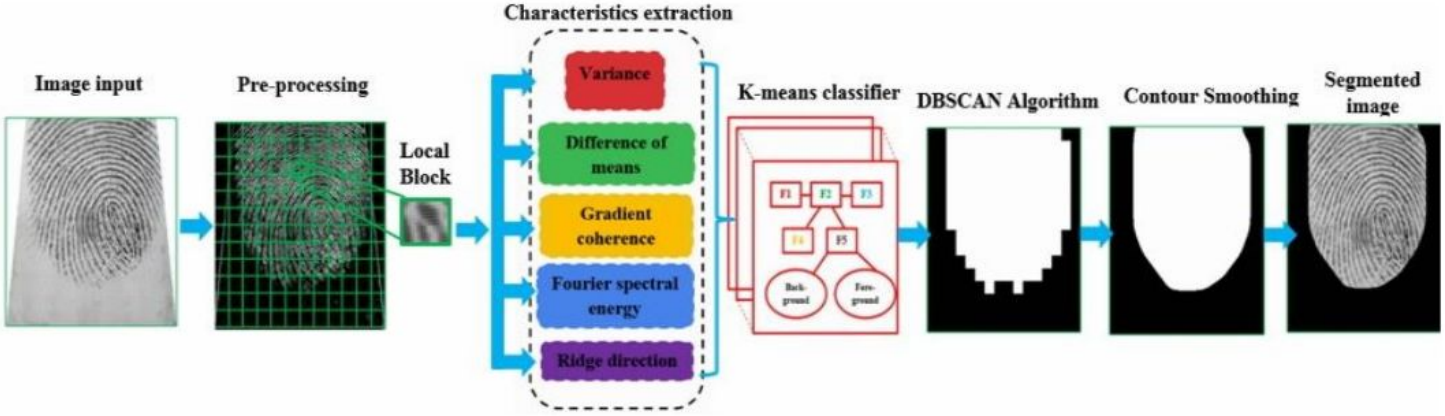
# DBSCAN examples

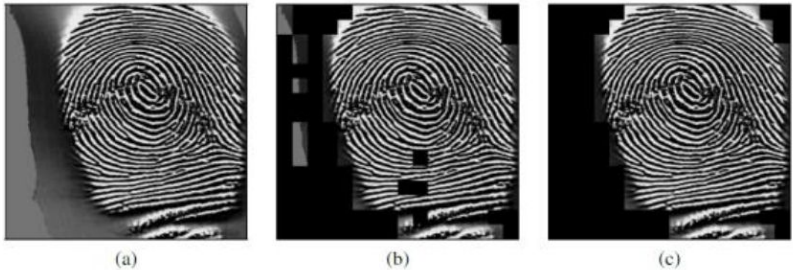
DBSCAN on t-SNE output
to analyze embeddings (Doc2vec)
of video games

[image source]



# DBSCAN examples

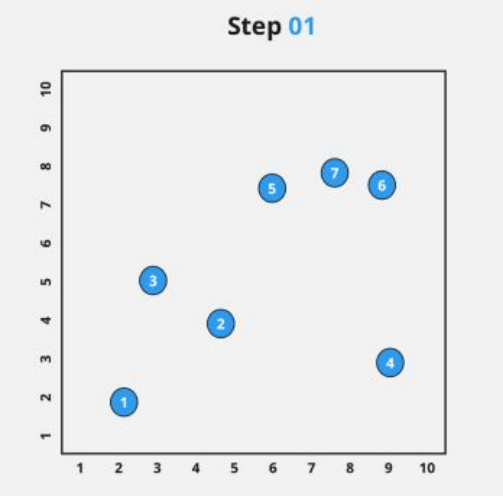


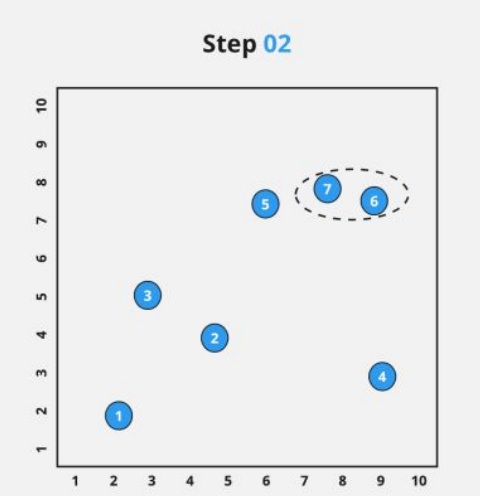




a - original, b - k-means, c - DBSCAN

Improving of Fingerprint Segmentation Images Based on K-MEANS and DBSCAN Clustering, Cherrat et al., 2019, IJECE

# Hierarchical clustering



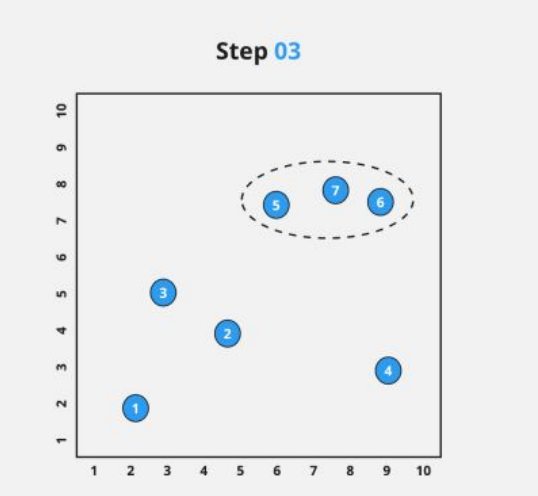
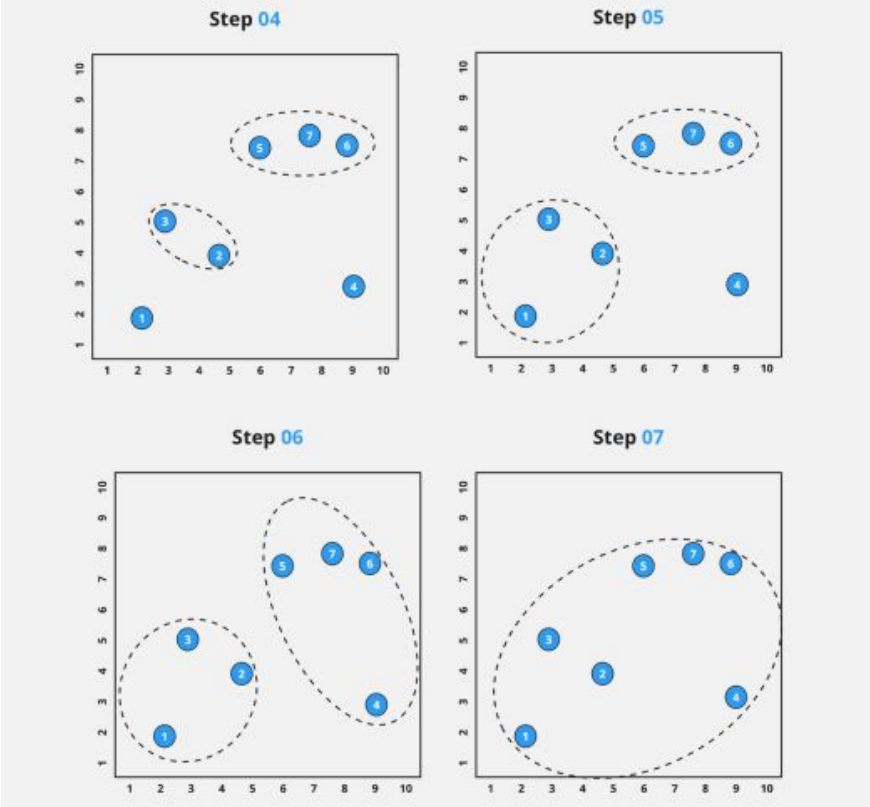
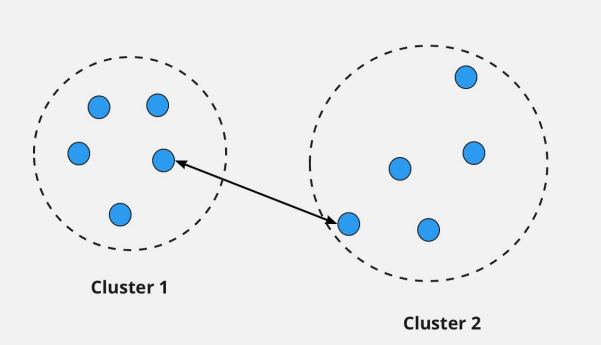




image source

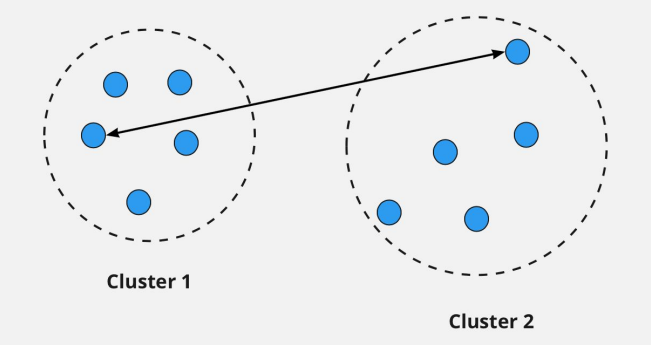
# Hierarchical clustering



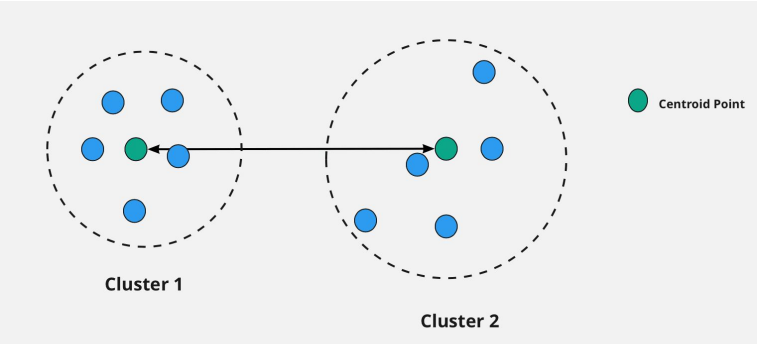
# Distance between clusters




Closest point

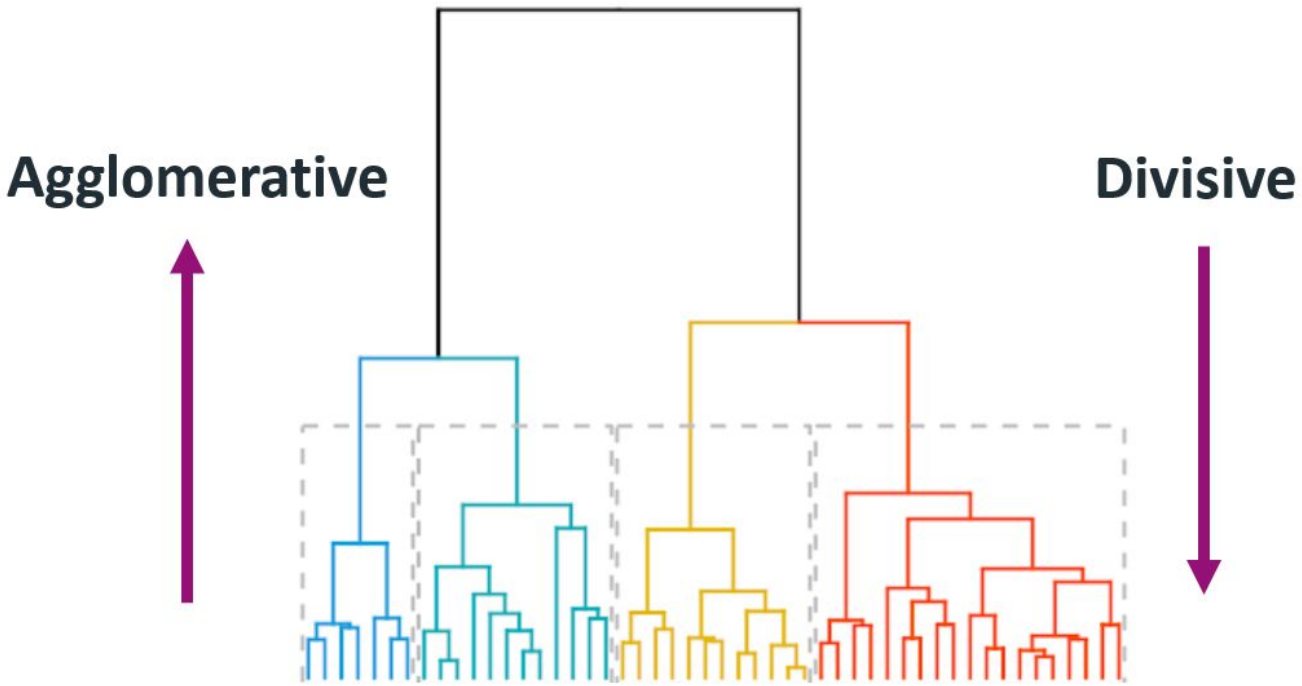



Farthest point

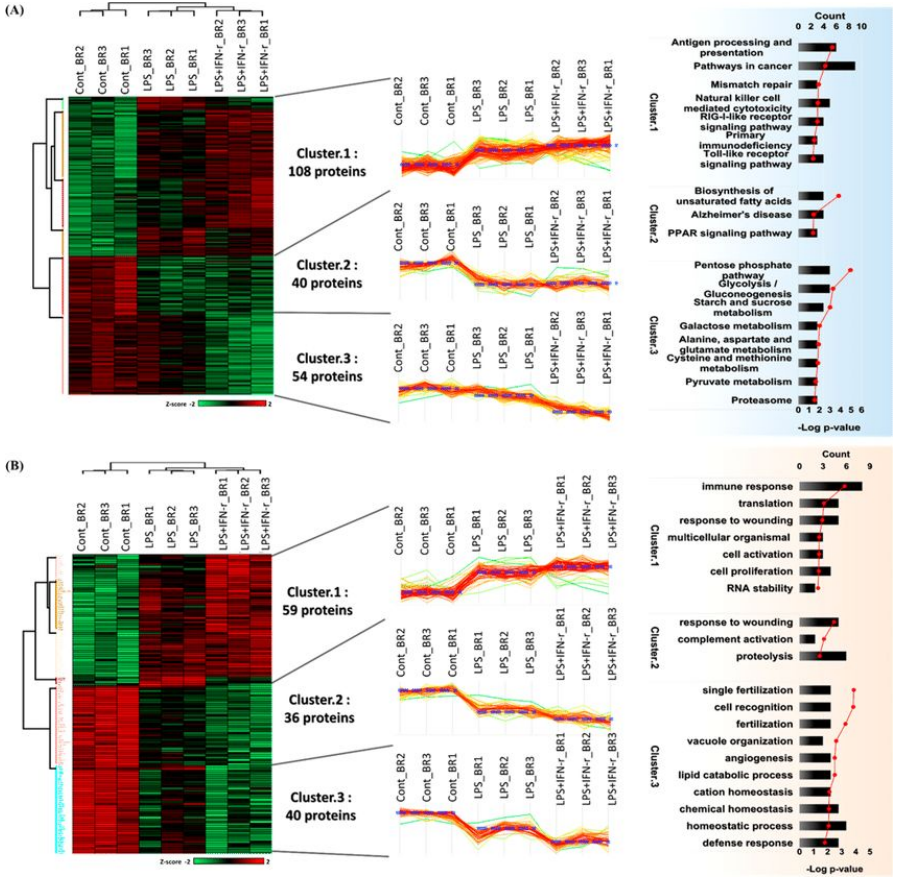



Centroid

# Dendrogram



# Dendrograms example

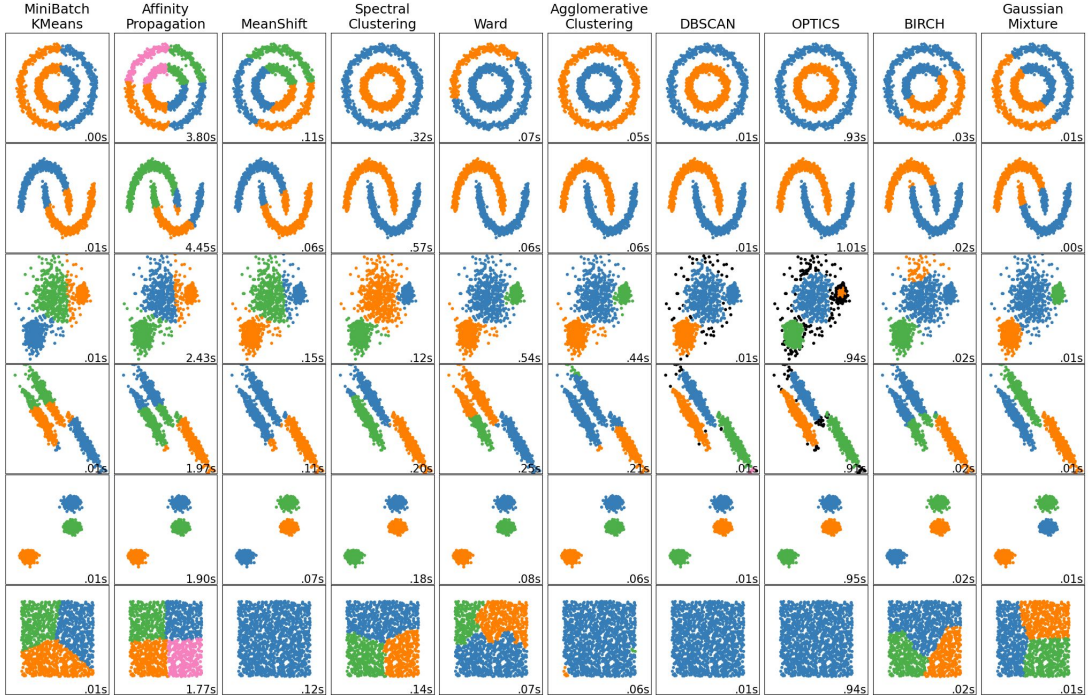



Hierarchical clustering and dendrograms often used in bioinformatics to visualize heatmaps of molecules interactions

Image source

Practical case: seriation, historical overview, Python implementation for images

# Many more

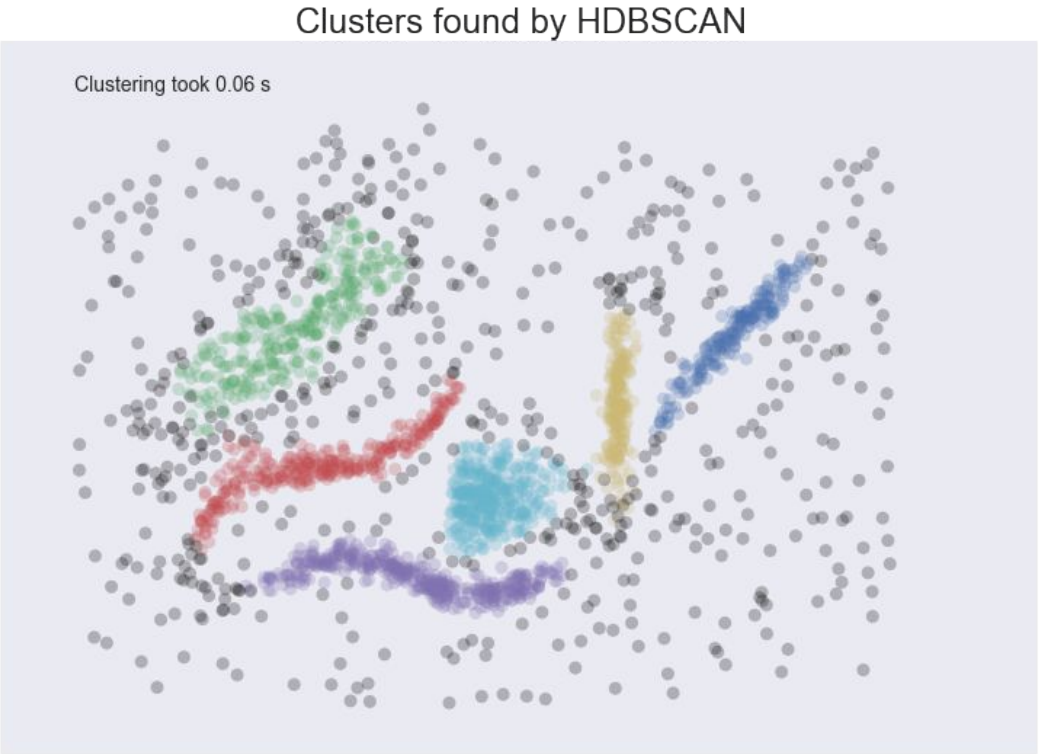



Read more:

[sklearn huge overview]
[HDBSCAN (hierarchy + DBSCAN)]
[timeseries clusterization] (ru)

# Clustering metrics

- Label based
  - Rand index
  - Mutual Information
  - Homogeneity
  - Completeness
  - V-measure
  - ...
- Label free
  - Silhouette Coefficient
  - Calinski-Harabasz Index
  - Davies-Bouldin Index
  - ...

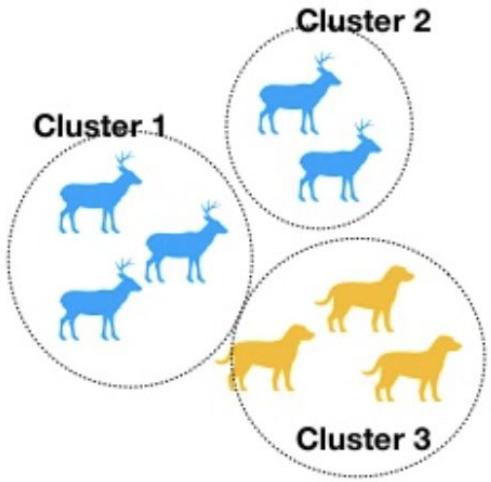



[Nice overview [ru]]
[Detailed explanations (sklearn docs)]

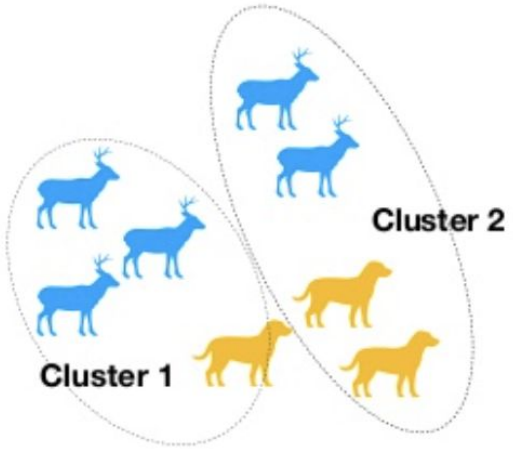
image source

# Homogeneity

Each cluster contains only members of a single class




Good

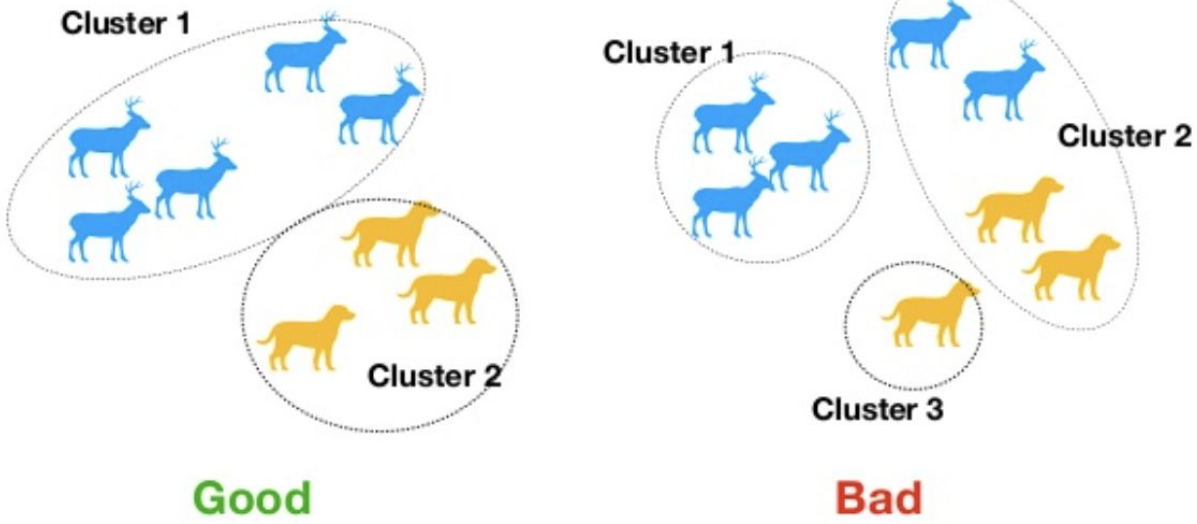



Bad

image source
and great slides on topic

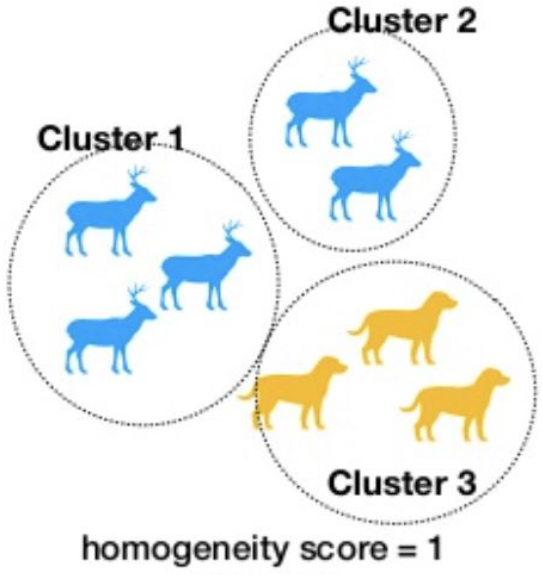
# Completeness

all members of a given class are assigned to the same cluster

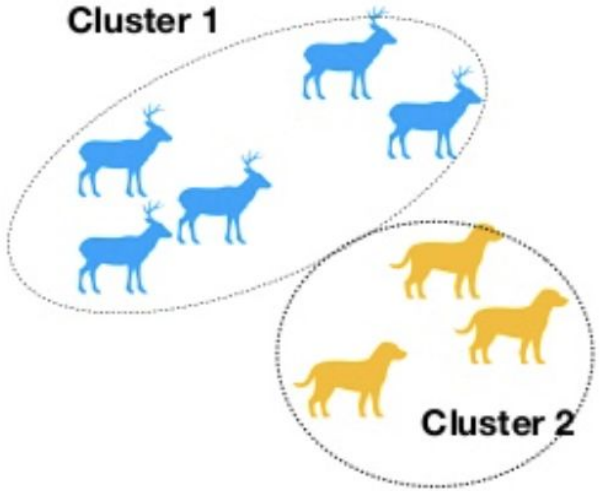


# V-measure

geometric mean of Homogeneity and Completeness




homogeneity score = 1




completeness score = 1

# Silhouette coefficient

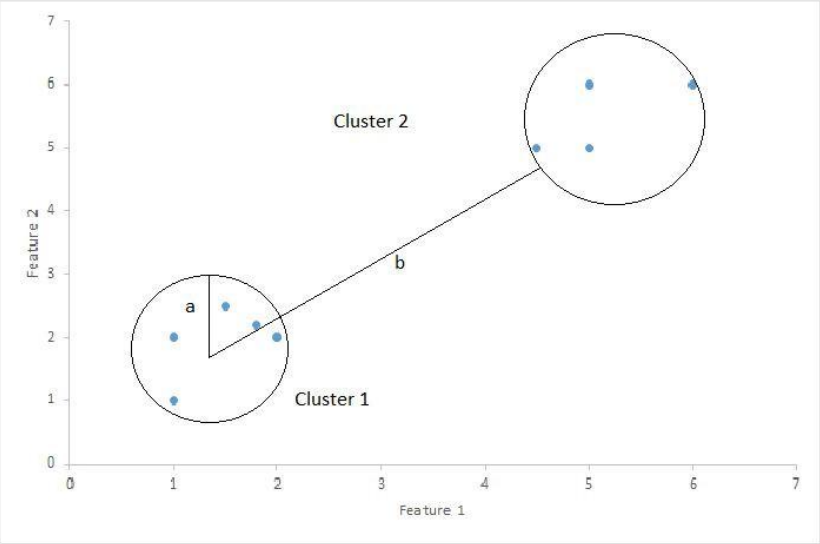


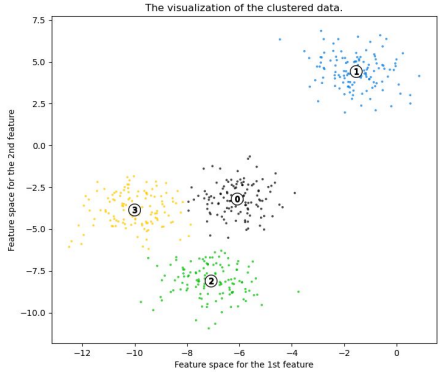

image source

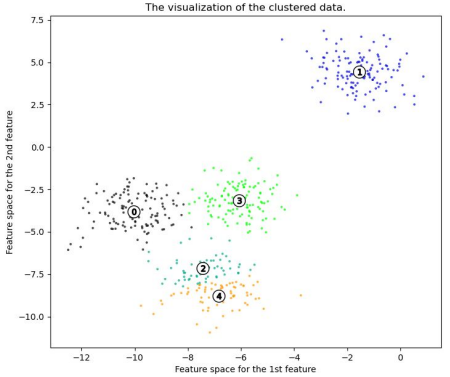
**a**: mean distance to points
in the *same cluster*

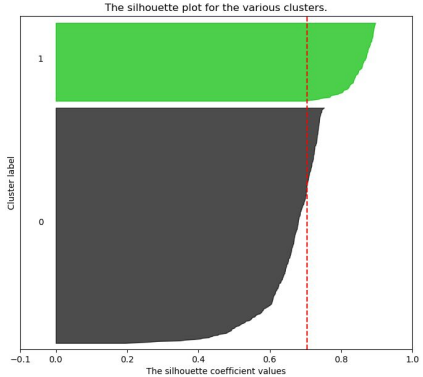
**b**: mean distance to points
in the *next nearest cluster*

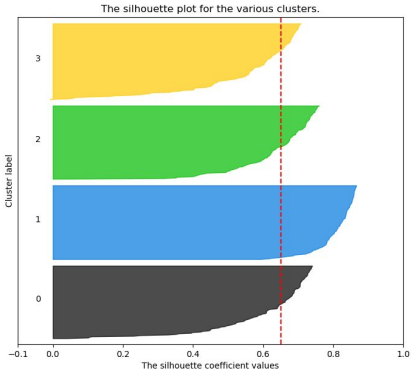
$$s_i = \frac{b_i - a_i}{max(a_i, b_i)}$$

# Silhouette analysis

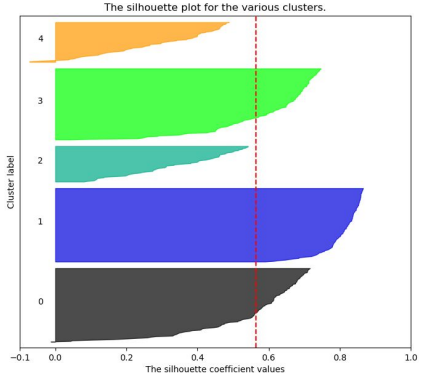



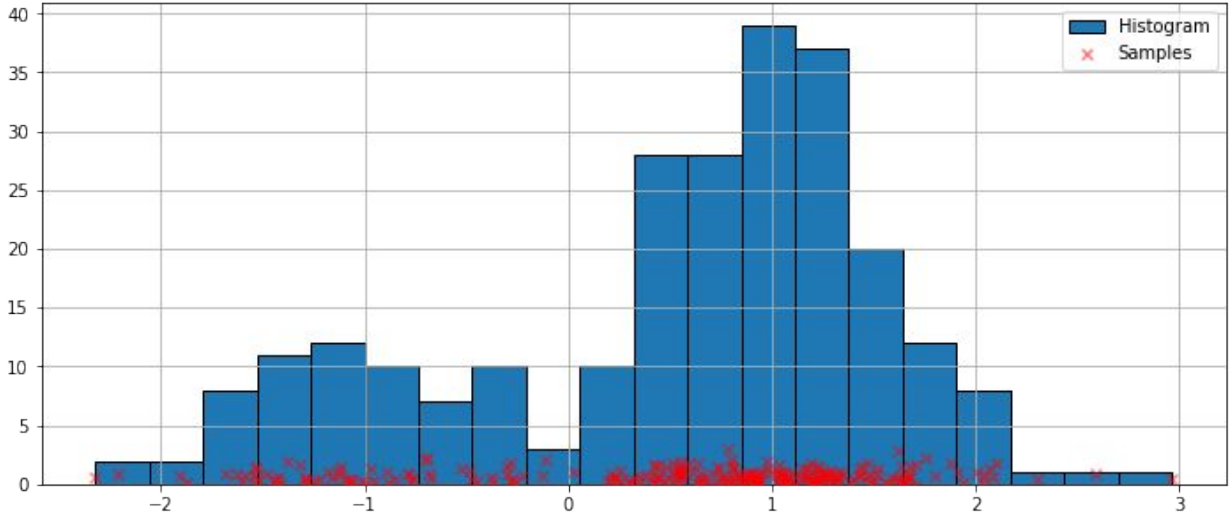

image source

# Density estimation

girafe ai

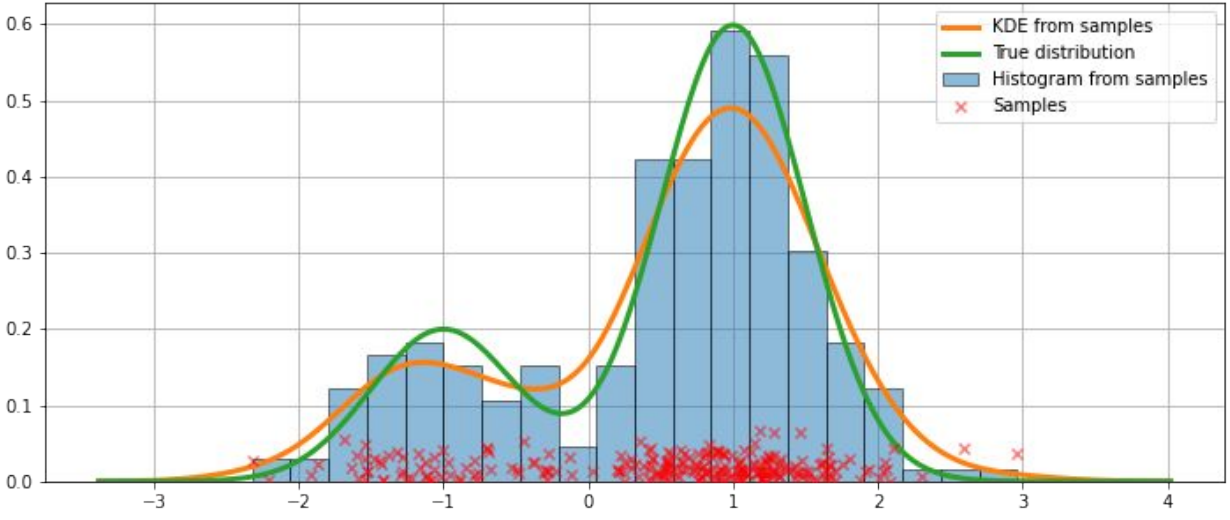
04

# Kernel density estimation

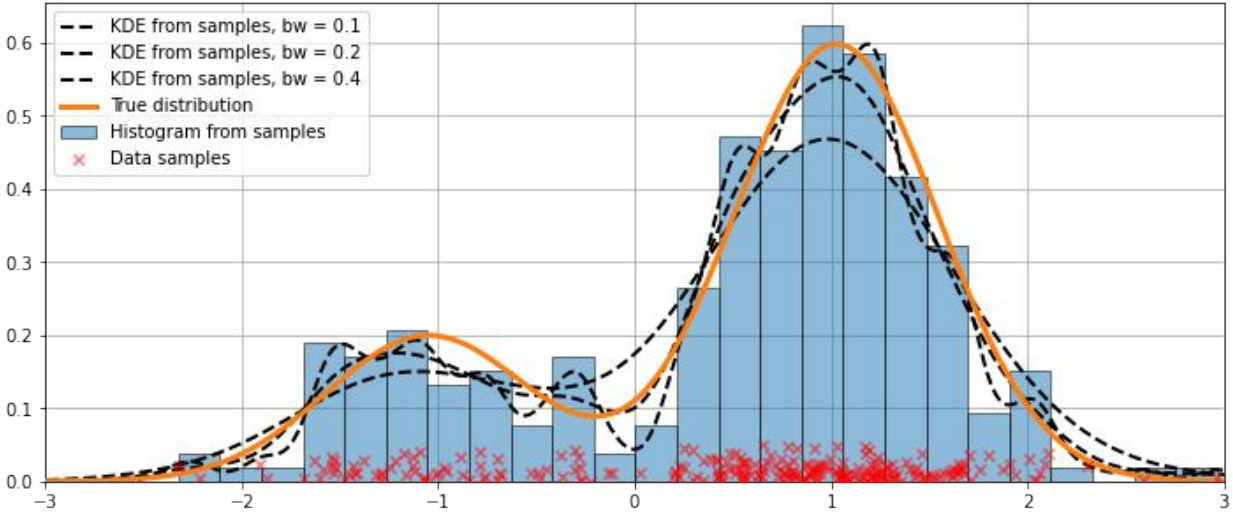



statsmodels documentation example

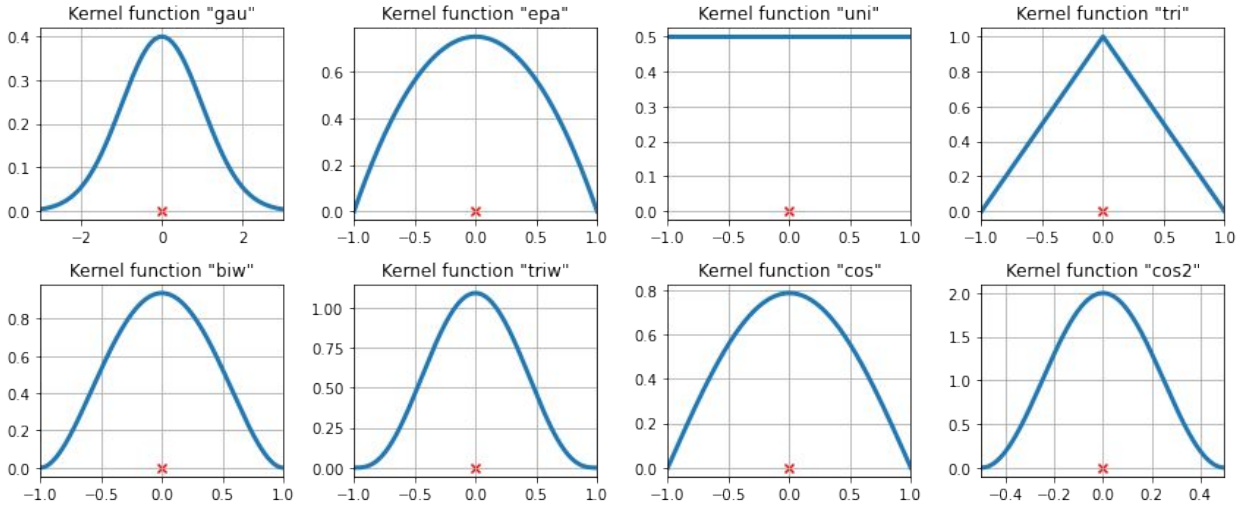
# Kernel density estimation



# Window size


KDE from samples, bw = 0.1
KDE from samples, bw = 0.2
KDE from samples, bw = 0.4
True distribution
Histogram from samples
Data samples
0.0
0.1
0.2
0.3
0.4
0.5
0.6
−3
−2
−1
0
1
2
3

# Kernel types



# Revise

- Geometrical machine learning
  - Dimensionality curse
  - Manifold assumption
- Dimensionality reduction
  - Feature selection
  - Multidimensional Scaling (MDS)
  - Isomap
  - Locally linear embedding (LLE)
  - t-SNE
- Clustering
  - k-means
  - DBSCAN
  - Hierarchical clustering
  - metrics
- Density estimation
  - Kernel density estimation

# Thanks for attention!

Questions?

---

girafe
ai

# Notable links

1. Good lecture on MDS, Isomap, LLE
2. Lecture on t-SNE (this one is good too)
3. Slides about clusterization
4. Metrics in clusterization
5. Slides about ICA
6. More clustering methods (in Russian)