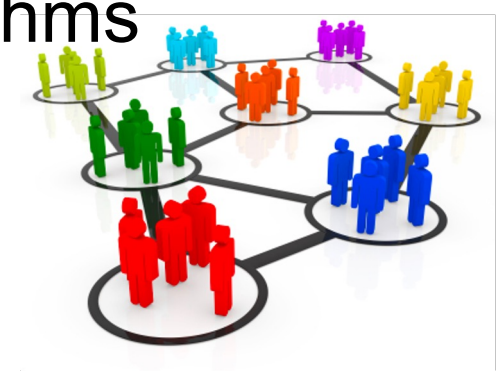
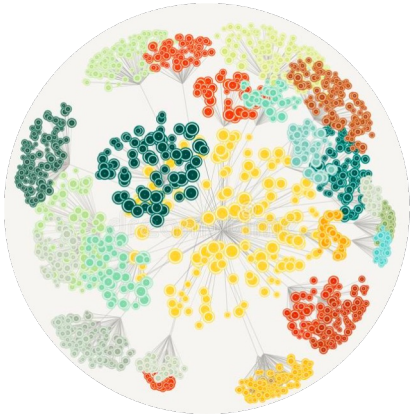
Unsupervise Learning

November 2023

Outline

- ◎ Part I: Clustering - General Concepts
  - ○ Real-life Applications
  - ○ Types of Clusterings
- ◎ Part II: Typical Clustering Algorithms

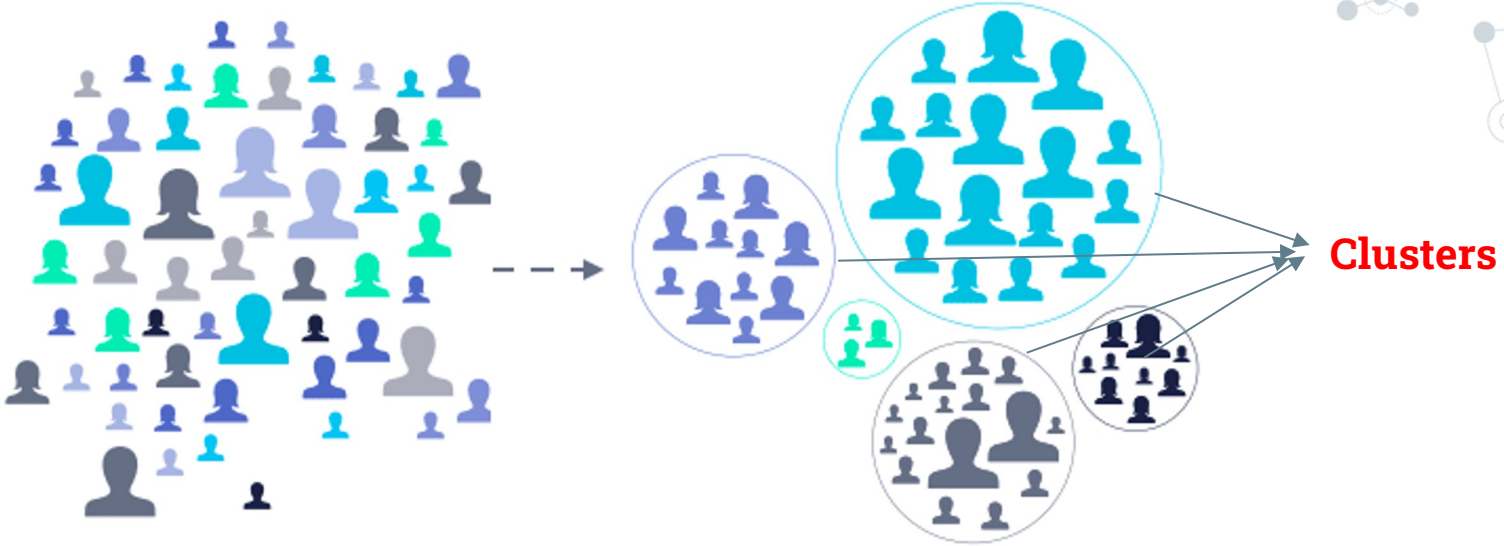




1.
Clustering
- General Concepts

Main idea, real-life applications, types

Motivating Example: Customer Segmentation

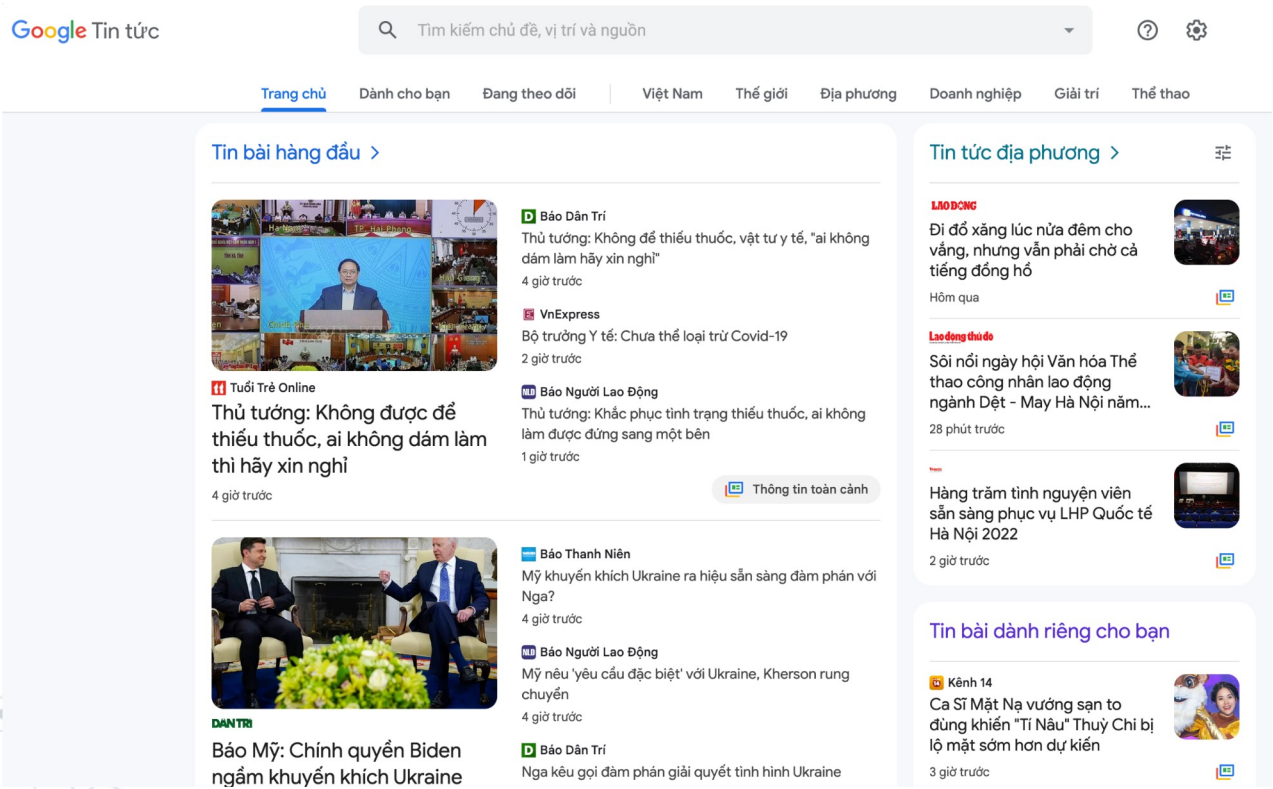


- Demographic
- Behavioral
- Geographic
- Psychographic

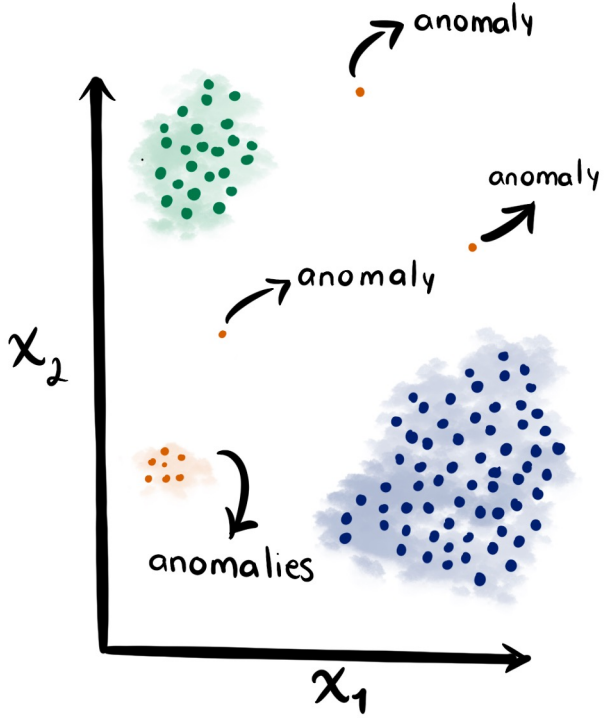
What is Cluster Analysis or Clustering?

Given a set of objects, place them in **groups** such that the **objects in a group are similar** (or related) to **one another and different from (or unrelated to) the objects in other groups**

Real-life Applications: Google News



Real-life Applications: Anomaly Detection

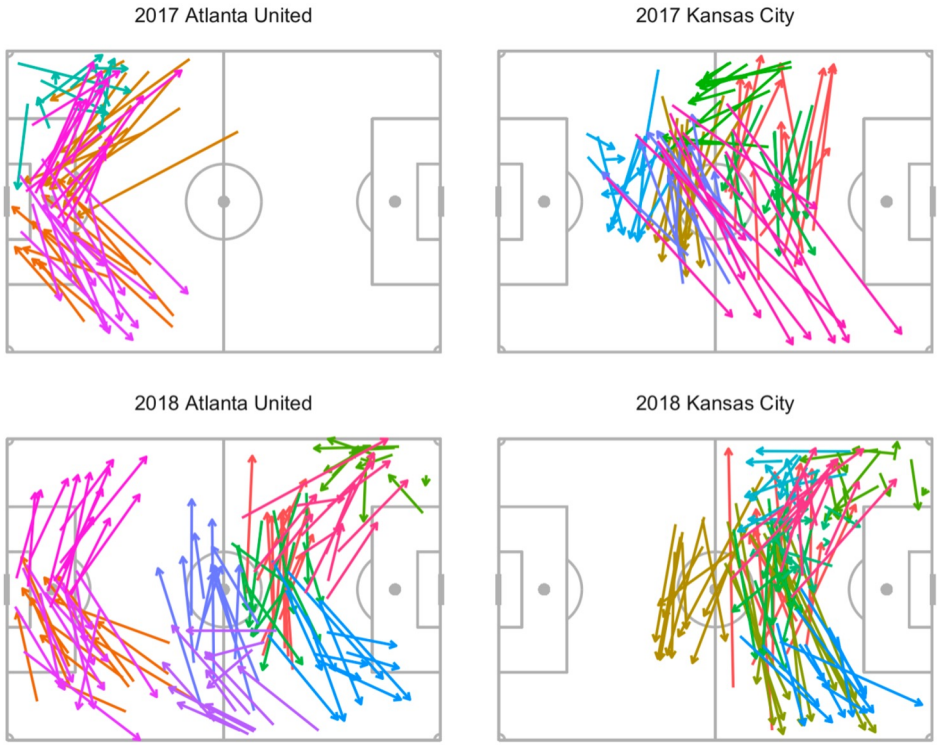


- Fake News Detection
- Fraud Detection
- Spam Email Detection

Source: https://towardsdatascience.com/unsupervised-anomaly-detection-on-spotify-data-k-means-vs-local-outlier-factor-f96ae783d7a7

Real-life Applications: Sport Science

Find players with similar styles

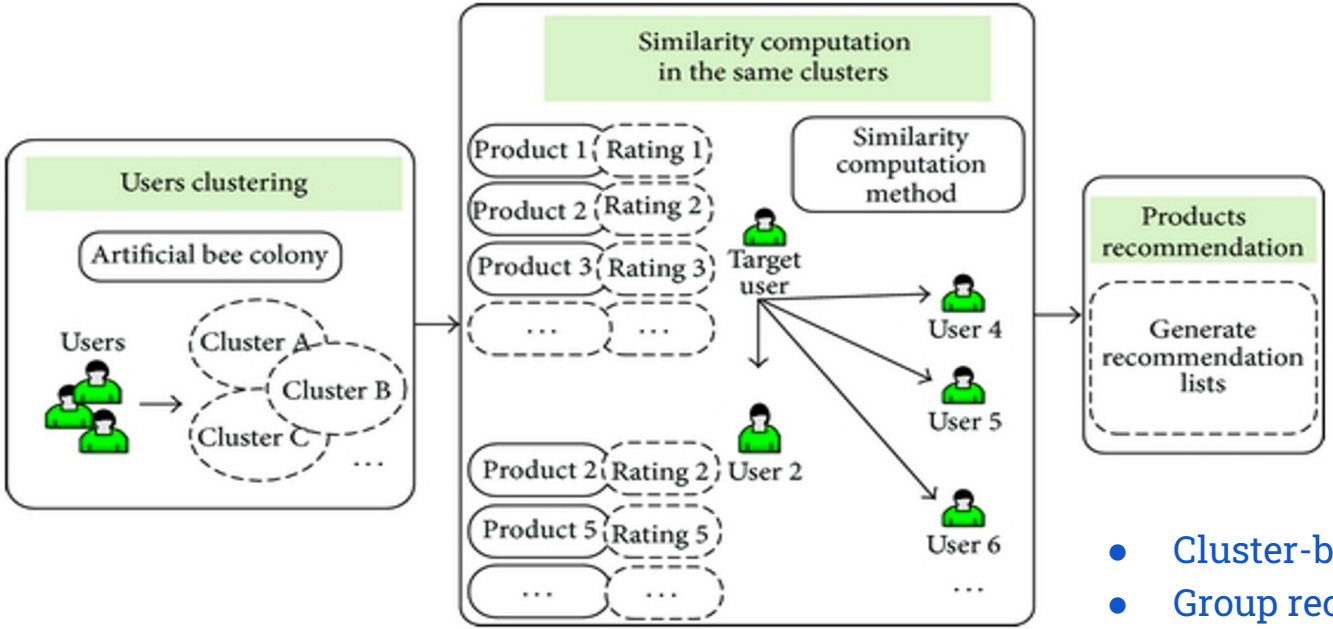


Source: https://www.americansocceranalysis.com/home/2019/3/11/using-k-means-to-learn-what-soccer-passing-tells-us-about-playing-styles

Real-life Applications: Image Segmentation

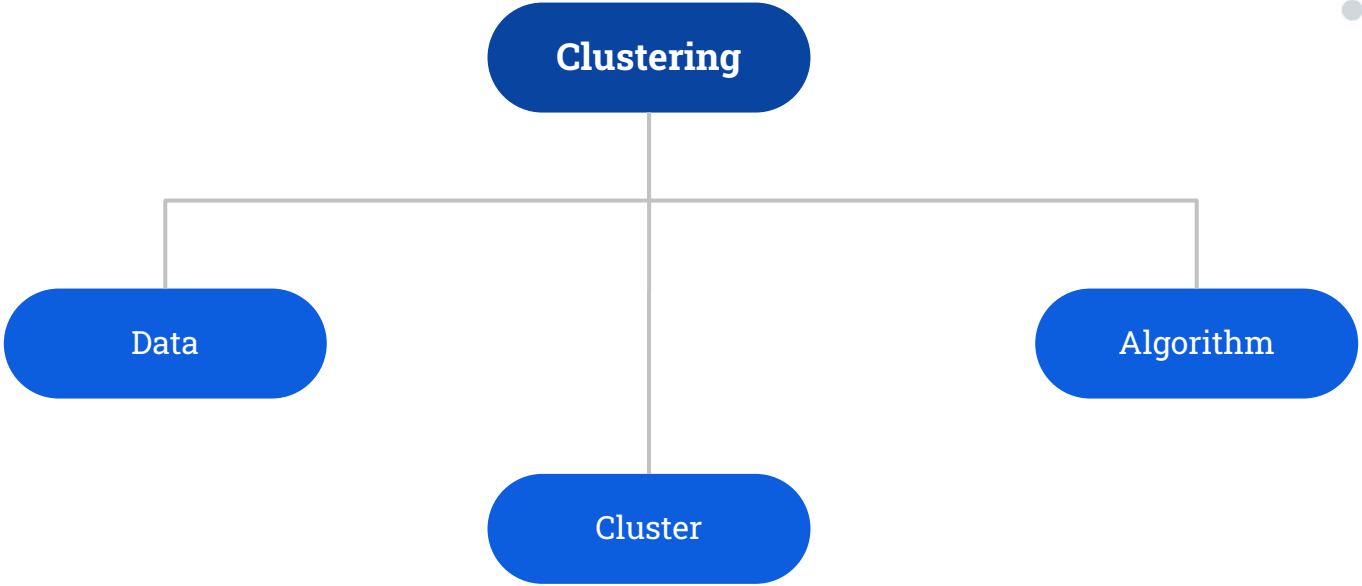
Source: http://pixelsciences.blogspot.com/2017/07/image-segmentation-k-means-clustering.html

Real-life Applications: Recommendation



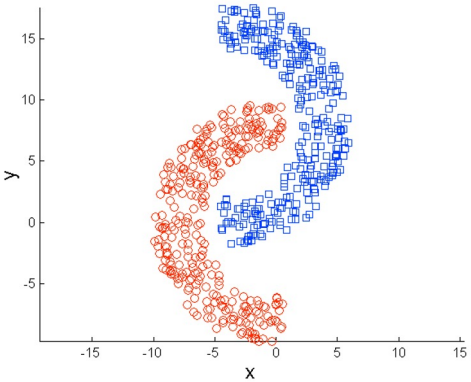
- Cluster-based ranking
- Group recommendation
- ...

What do affect on Cluster Analysis?



Characteristics of the Input Data Are Important

- **High dimensionality**
  - Dimensionality reduction
- **Types of attributes**
  - Binary, discrete, continuous, asymmetric
  - Mixed attribute types, e.g., continuous & nominal)
- **Differences in attribute scales**
  - Normalization techniques
- **Size of data set**
- **Noise and Outliers**
- **Properties of the data space**



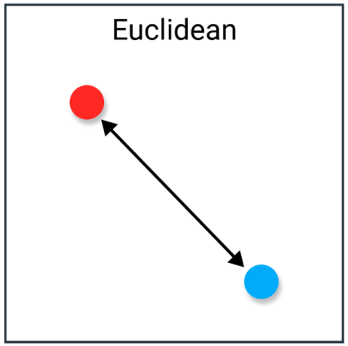
Characteristics of Cluster

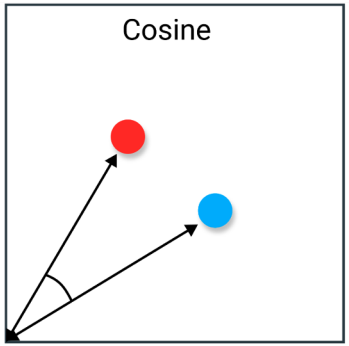
- **Data distribution**
  - Parametric models
- **Shape**
  - Globular or arbitrary shape
- **Differing sizes**
- **Differing densities**
- **Level of separation among clusters**
- **Relationship among clusters**
- **Subspace clusters**

⇨ **Distance Metrics**

How to Measure the Similarity/Distance?

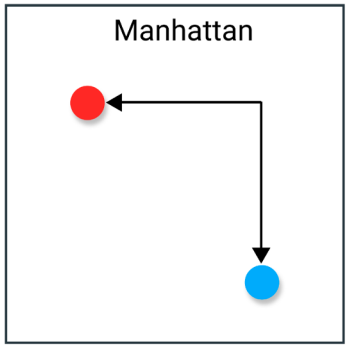
$$D(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2)}$$

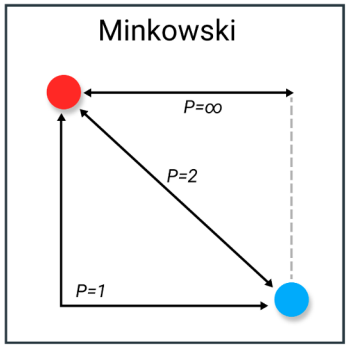




$$D(x,y) = cos(\theta) = \frac{x \cdot y}{\|x\| \, \|y\|}$$

$$D(x,y) = \sum_{i=1}^{k} |x_i - y_i|$$





$$D(x,y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Source: https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

Notion of a Cluster can be Ambiguous

How many clusters?

6 Clusters

2 Clusters

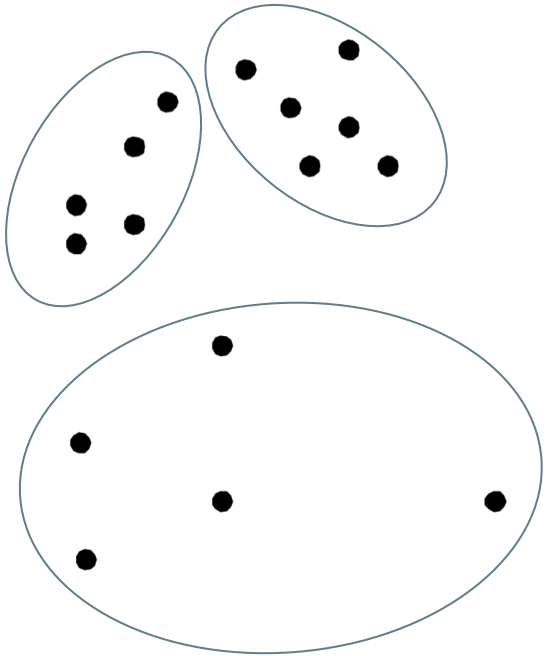
4 Clusters

Types of Clusterings

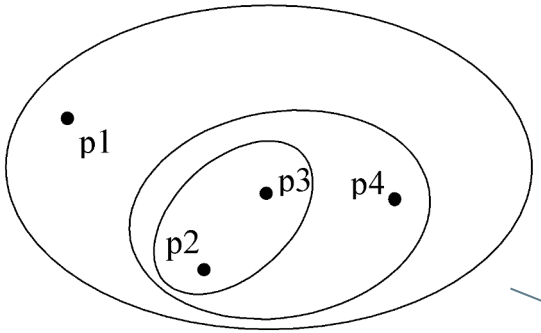
Source: https://www.datanovia.com/en/blog/types-of-clustering-methods-overview-and-quick-start-r-code/

Partitional Clustering

Data objects are separated into **non-overlapping subsets, i.e., clusters**

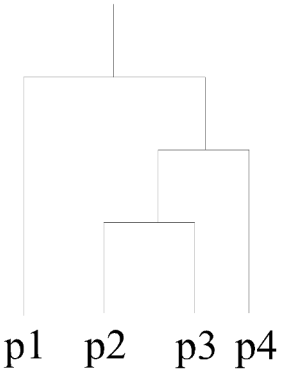


Hierarchical Clustering



Hierarchical Clustering

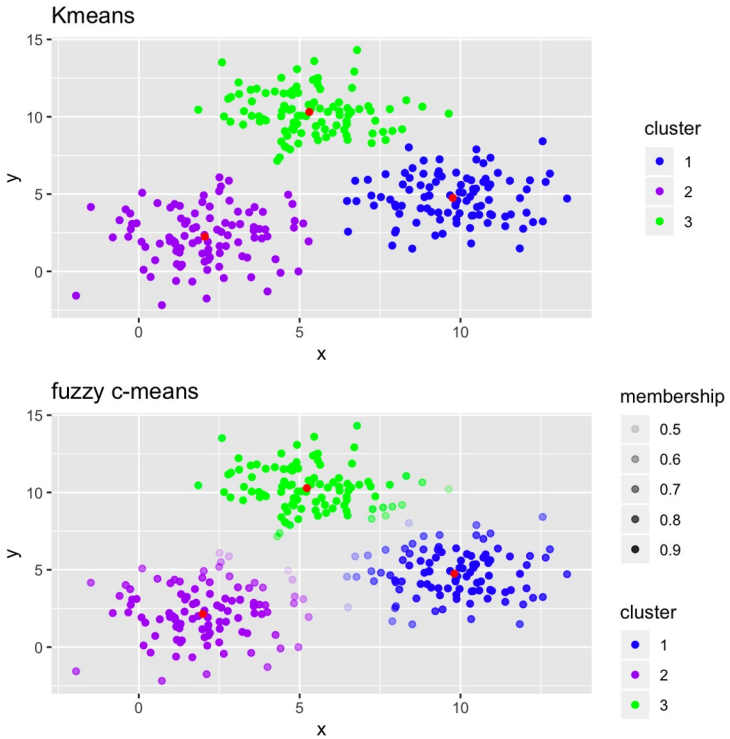
Data objects are separated into **nested clusters as a hierarchical tree**



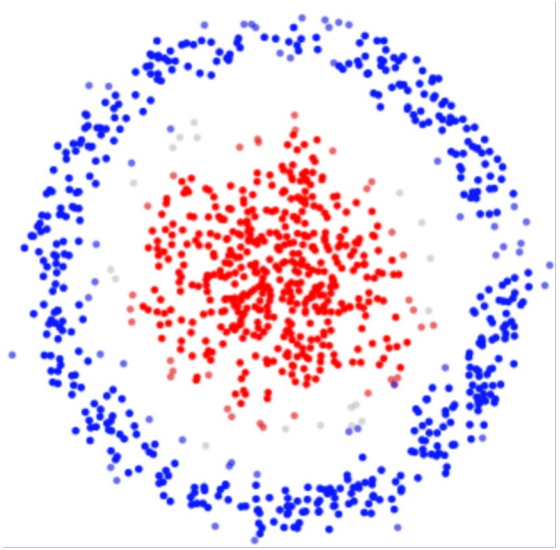
Clustering dendrogram

Fuzzy Clustering

Fuzzy clustering, i.e., soft clustering, is a form of clustering in which **each data point can belong to more than one cluster with weights**



Density-based Clustering

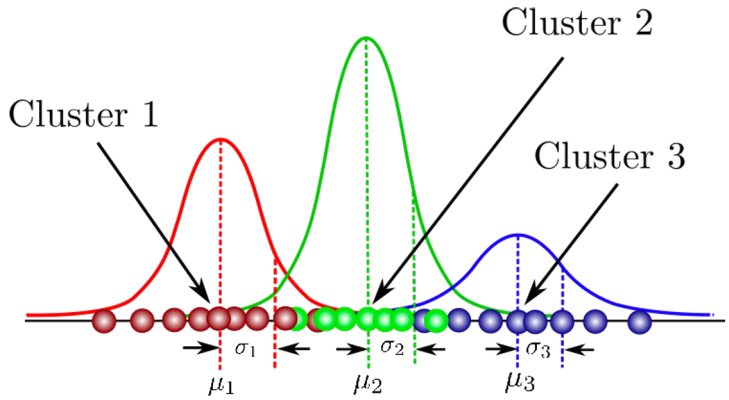


Non-linear separation

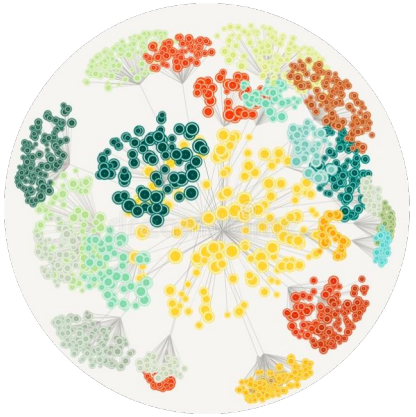
A cluster is **a dense region of points**, which is separated by **low-density regions**, from **other regions of high density**.

Model-based Clustering

Model-based clustering assumes that **the data were generated by a model** and **tries to recover the original model from the data**.



Gaussian Mixture Model



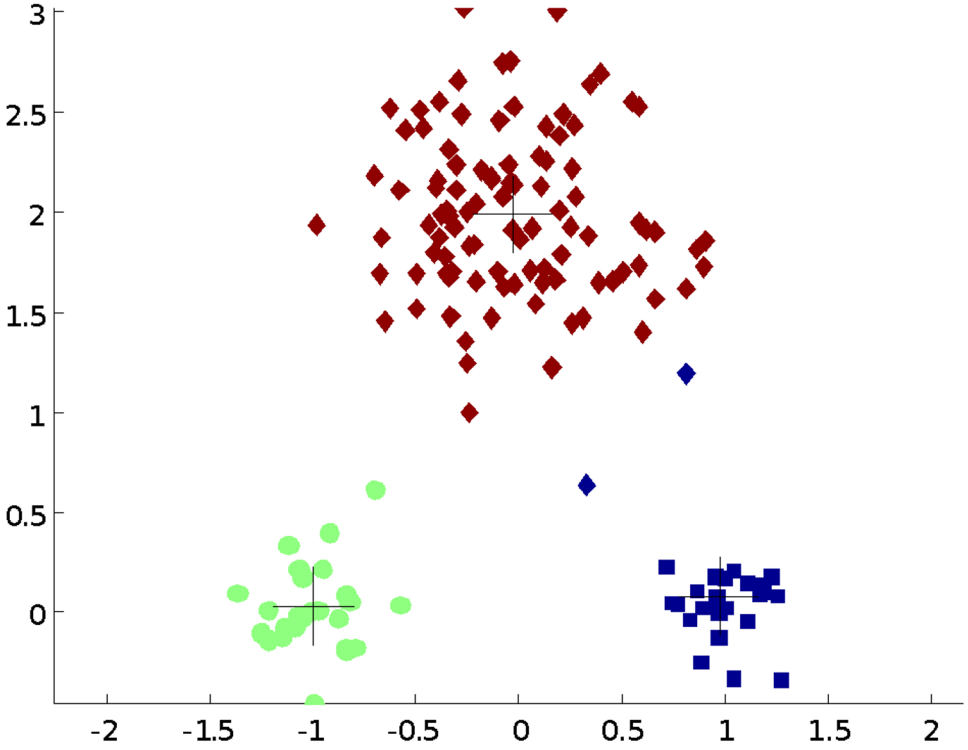
2.
Typical Clustering Algorithms

Intuition, Main Idea, Limitation

Typical Clustering Algorithms

- Partitional Clustering
  - K-Means & Variants
- Hierarchical Clustering
  - HAC
- Density-based Clustering
  - DBSCAN

K-Means Clustering: An Example



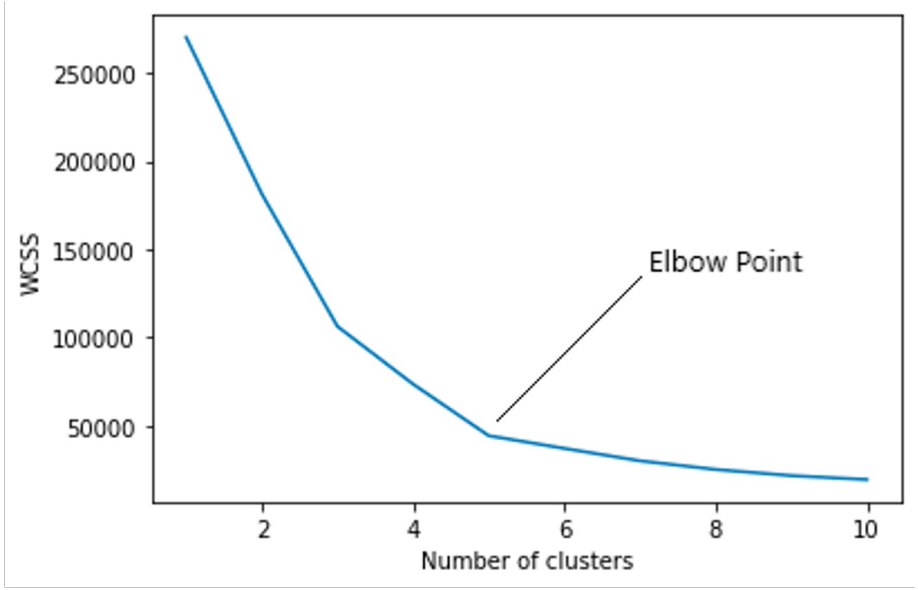
K-Means Clustering

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

- **Main idea**: Each point is assigned to the cluster **with the closest centroid**
- Number of clusters, **K, must be specified**
- Sum of Squared Error (SSE)
- Complexity: O(n * K * I * d)
  - n = number of points, K = number of clusters,
  - I = number of iterations, d = number of attributes

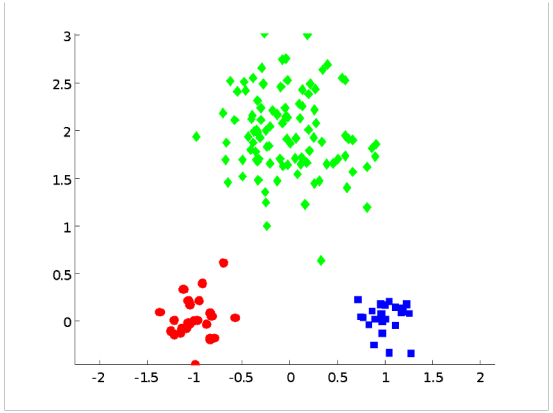
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

Elbow Method for Optimal Value of K

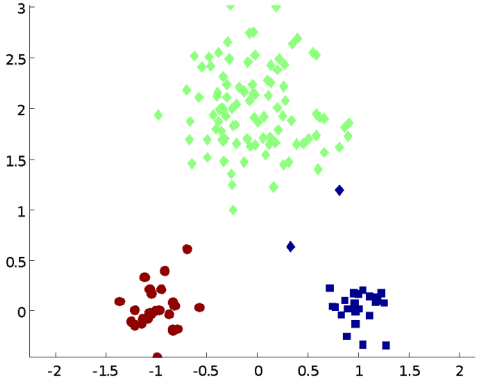


WCSS is the sum of squared distance between each point and the centroid in a cluster
The graph will rapidly change at a point named **Elbow Point**

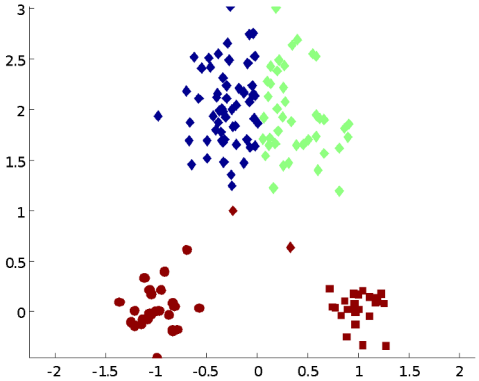
Two different K-means Clusterings



Original Points

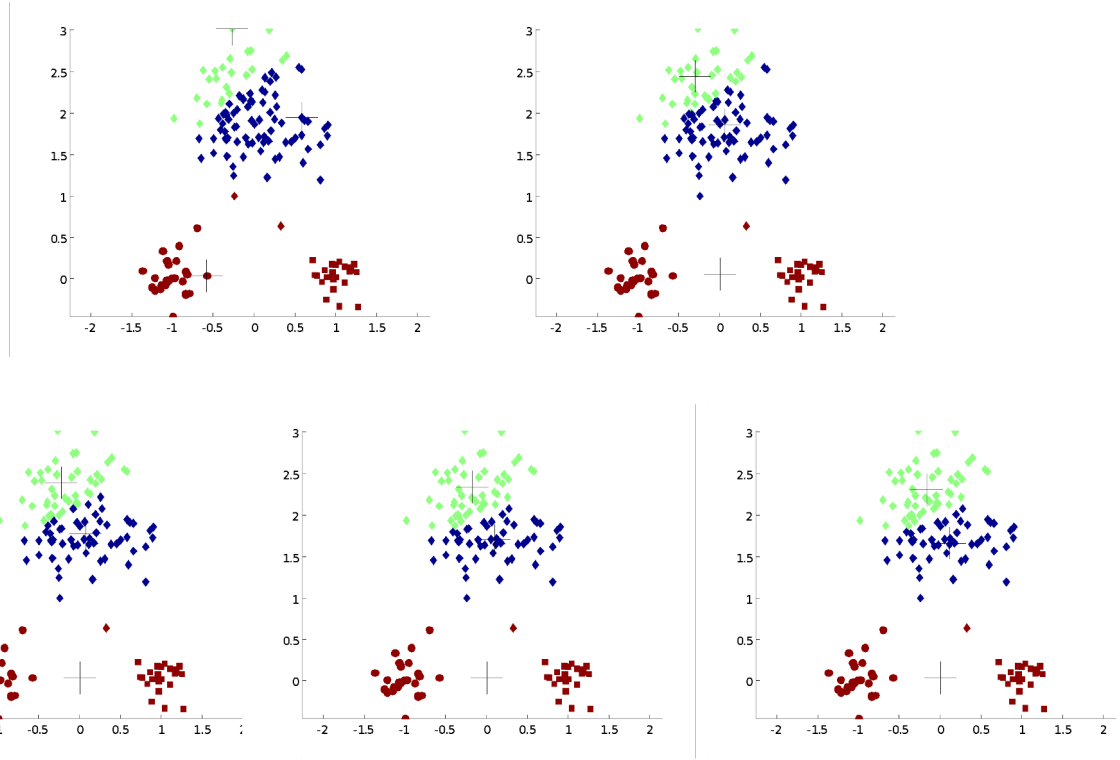


Optimal Clustering



Sub-optimal Clustering

Importance of Choosing Initial Centroids



Solutions to Initial Centroids Problem

- **Multiple runs**
  - Helps, but probability is not on your side
- **Use some strategies to select the k initial centroids** and then select among these initial centroids
  - Select most widely separated, e.g., K-means++
  - Use hierarchical clustering to determine initial centroids
- **Bisecting K-Means**
  - Not as susceptible to initialization issues

K-Means++

1. **Choose one center uniformly** at random among the data points.
2. For each data point **x** not chosen yet, compute **D(x)**, the distance between x and the nearest center that has already been chosen.
3. Choose **one new data point at random** as a new center, using a weighted probability distribution where a point x is chosen with probability proportional to $D(x)^2$.
4. Repeat Steps 2 and 3 until k centers have been chosen.
5. Now that the initial centers have been chosen, proceed using **standard *K*-Means clustering**

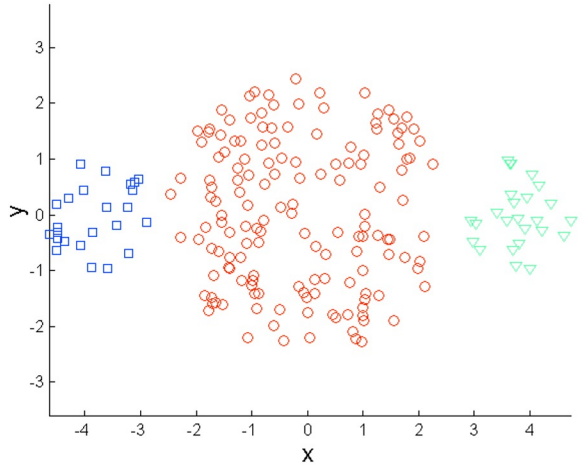
$$\frac{\min_j d^2(C_j, x_i)}{\sum_i \min_j d^2(C_j, x_i)}$$

Bisecting K-Means

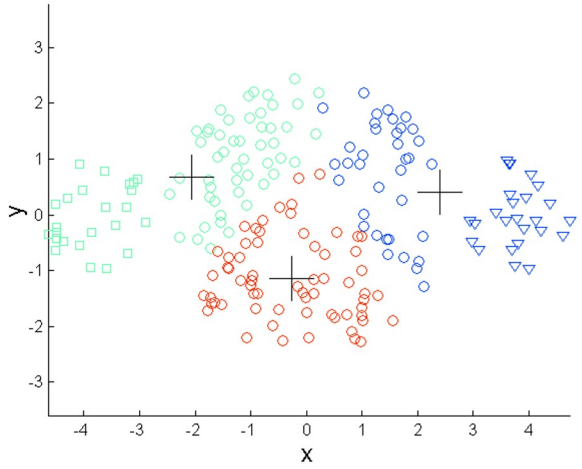
1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3: Select a cluster from the list of clusters
4: **for** $i = 1$ to *number_of_iterations* **do**
5: Bisect the selected cluster using basic K-means
6: **end for**
7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains K clusters

It is a variant of K-means that can produce a partitional or a hierarchical clustering

Limitations of K-means: Differing Sizes

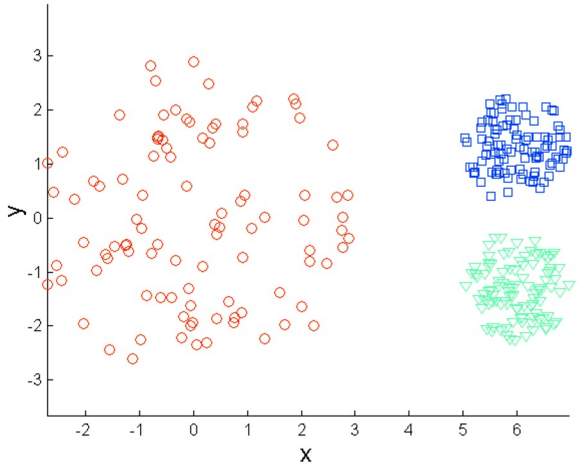


Original Points

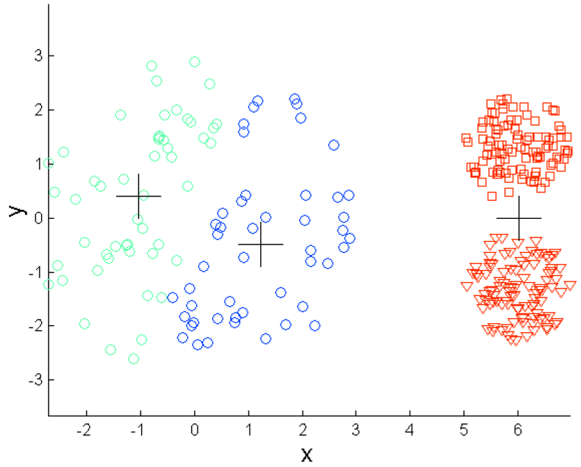


K-means (3 Clusters)

Limitations of K-means: Differing Density

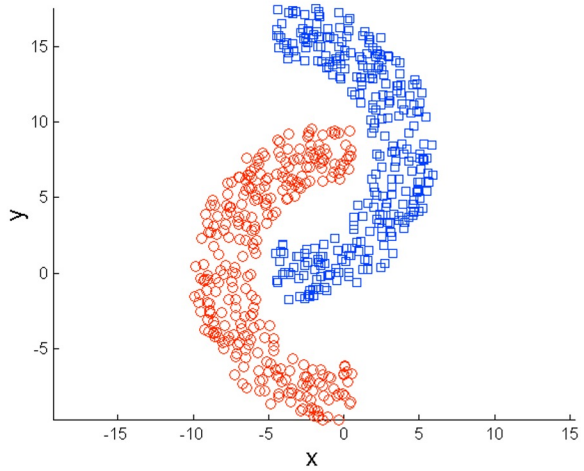


Original Points

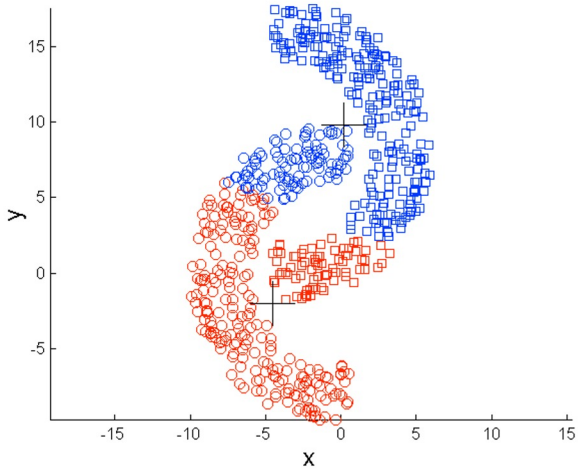


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

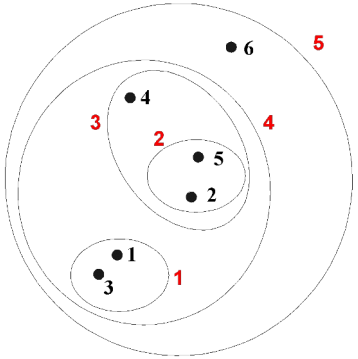


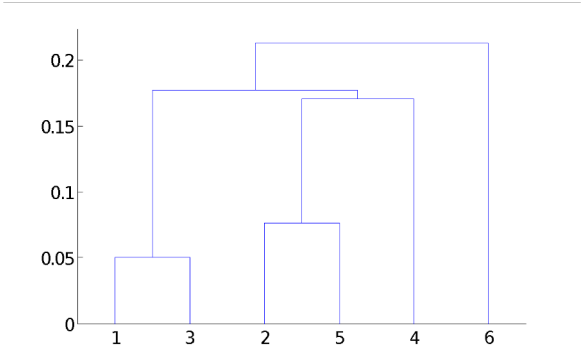
Original Points



K-means (2 Clusters)

Hierarchical Agglomerative Clustering

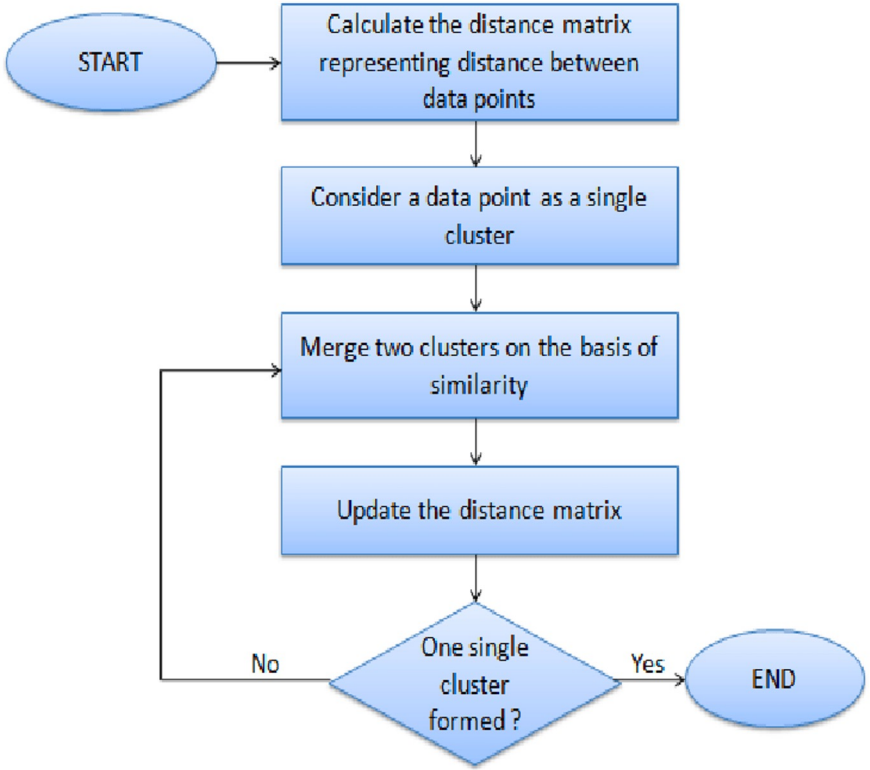




dendrogram

- **Main Idea**:
  - Start with the points as **individual clusters**
  - At each step, merge the **closest pair** of clusters until **only one cluster** (or K clusters) left
- Key operation is the computation of the proximity of two clusters
  - Worst-case Complexity: $O(N^3)$

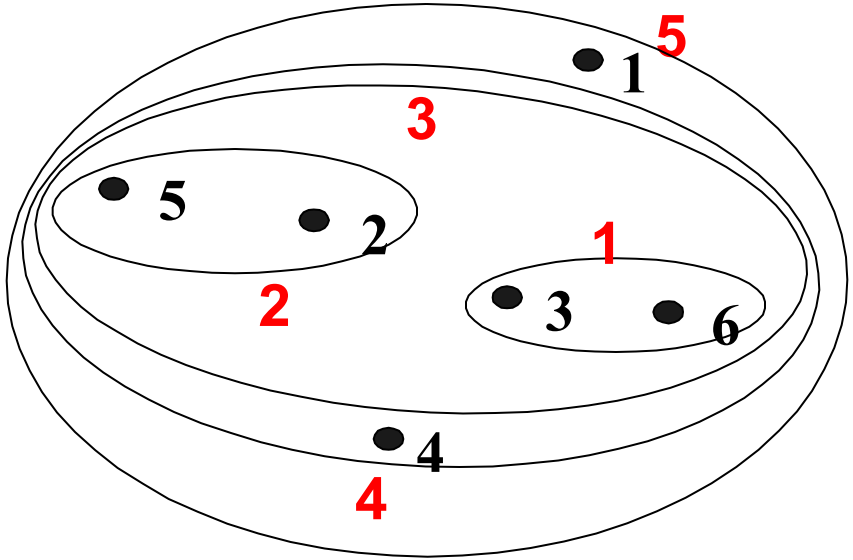
HAC: Algorithm



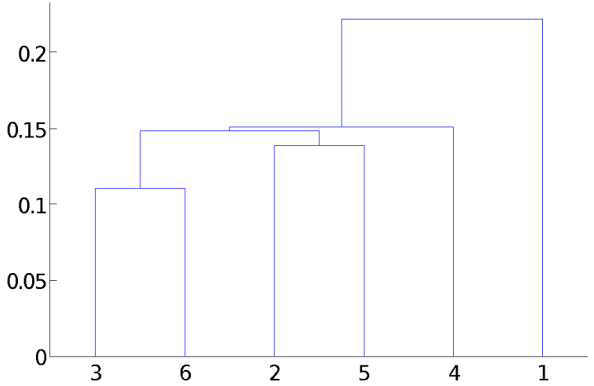
Closest Pair of Clusters

- Many variants to defining closest pair of clusters
- **Single-link**
  - Similarity of the closet elements
- **Complete-link**
  - Similarity of the “furthest” points
- **Average-link**
  - Average cosine between pairs of elements
- **Ward’s Method**
  - The increase in squared error when two clusters are merged

HAC - Single-link (MIN)

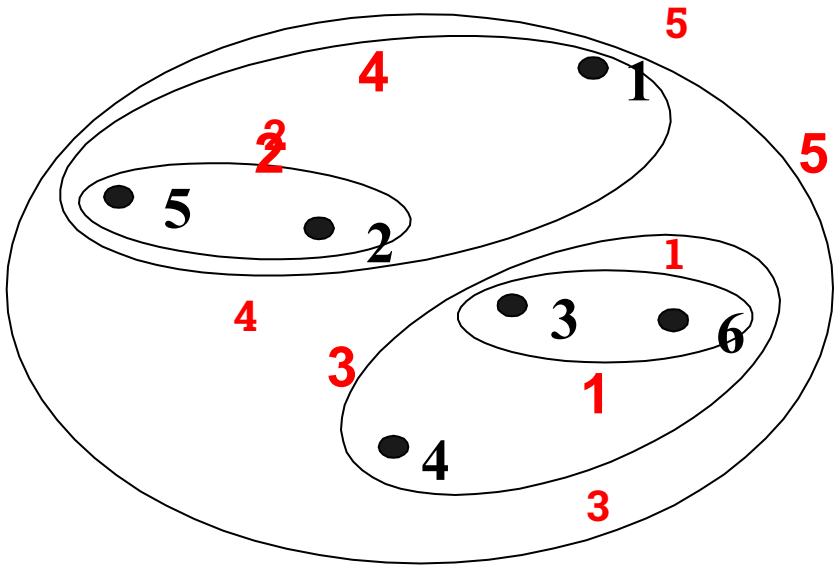


Nested Clusters

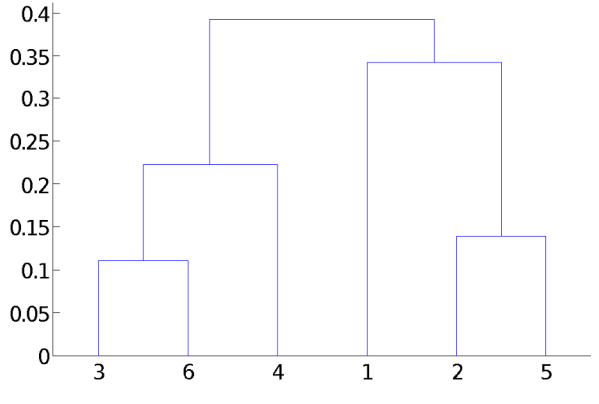


Dendrogram

HAC - Complete-link (MAX)

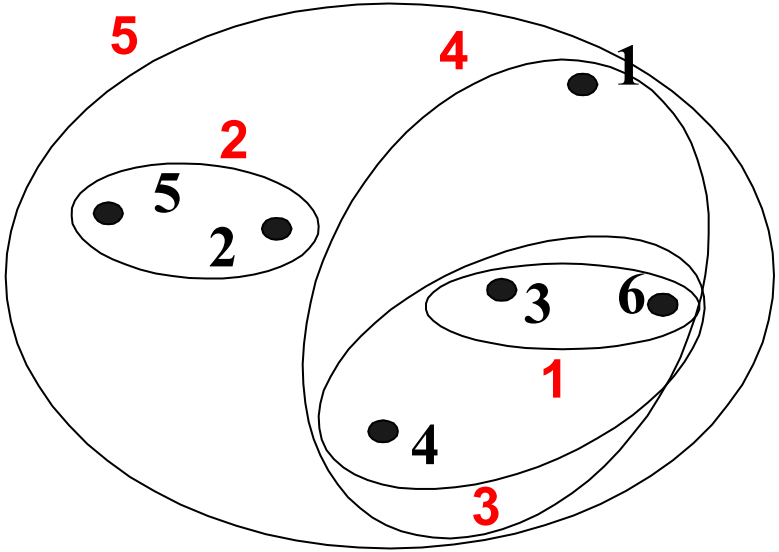


Nested Clusters

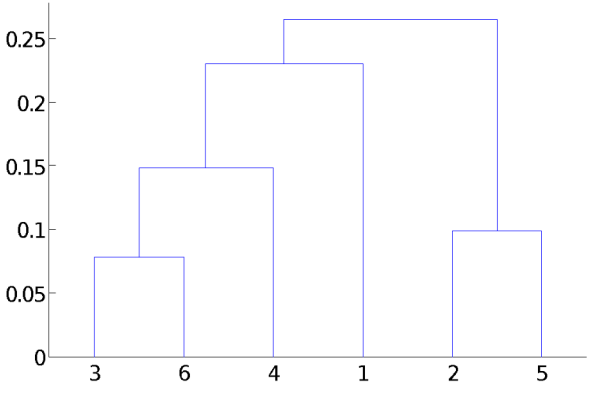


Dendrogram

HAC - Average-link



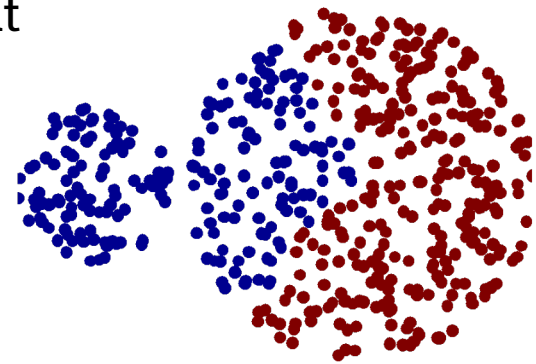
Nested Clusters



Dendrogram

HAC: Limitations

- Once two clusters are combined, it **cannot be undone**
- No **global objective function** is directly minimized
- Typical Problems:
  - Sensitivity to noise
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters



Density-based Clustering - DBSCAN

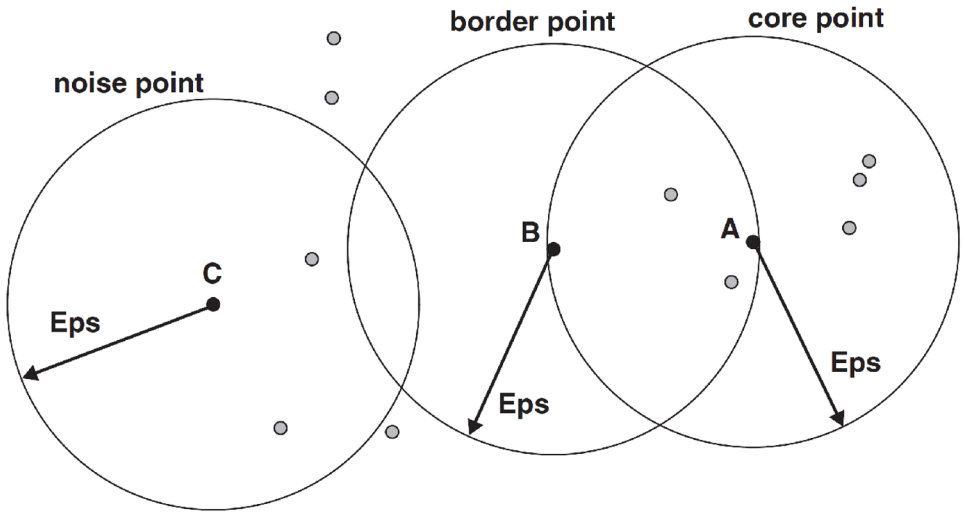
- **Main Idea**: Clusters are **regions of high density** that are **separated** from one another by **regions on low density**.
- **Density** = number of points within **a specified radius (Eps)**
  - Core point
  - Border point
  - Noise point

DBSCAN: Algorithm

DBSCAN algorithm.

1: Label all points as core, border, or noise points.
2: Eliminate noise points.
3: Put an edge between all core points that are within *Eps* of each other.
4: Make each group of connected core points into a separate cluster.
5: Assign each border point to one of the clusters with its associated core points.

How to Determine Points?

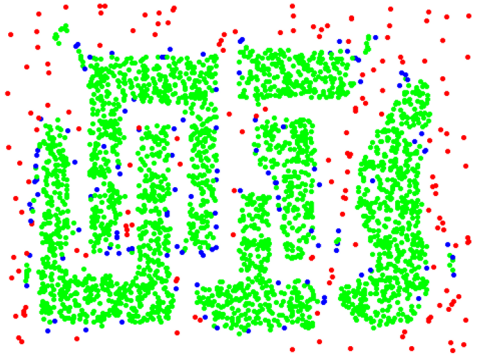


MinPts = 7

- **Core point:** Has at least a specified number of points (MinPts) within Eps
- **Border point**: not a core point, but is in the neighborhood of a core point
- **Noise point**: any point that is not a core point or a border point

DBSCAN: Core, Border and Noise Points

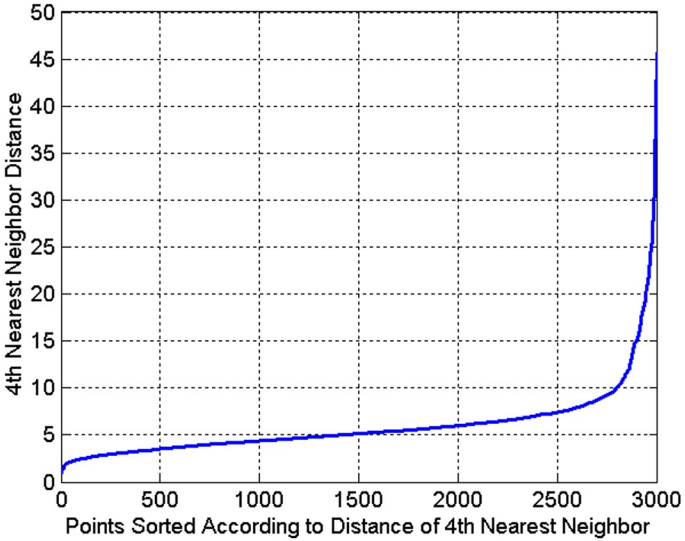
Original Points



Point types: core, border and noise

Eps = 10, MinPts = 4

DBSCAN: How to Determine Eps, MinPts?

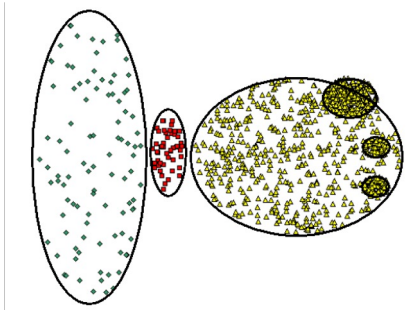


Intuition:

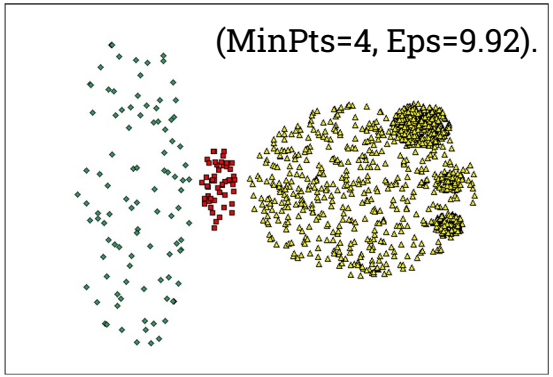
- Core point: the k-th nearest neighbors are at a close distance.
- Noise point: the k-*th* nearest neighbors are at a far distance.

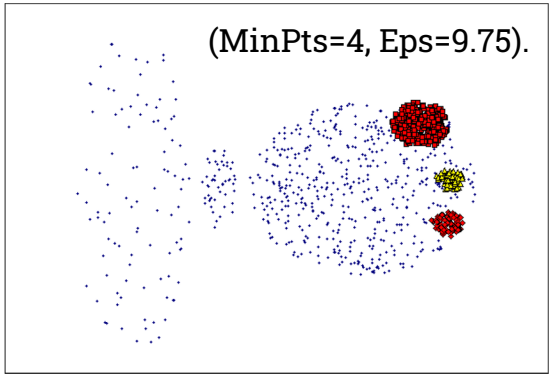
Plot sorted distance of every point to its k-th nearest neighbor

DBSCAN: Limitations



Original Points

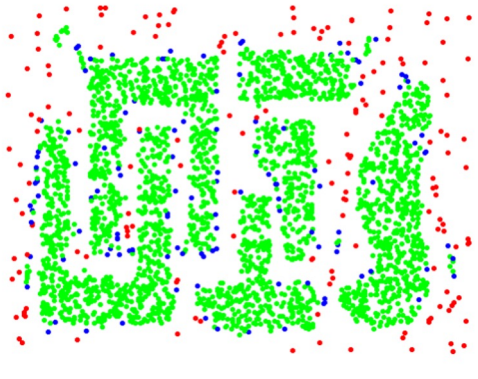




- **Varying densities**
- **High-dimensional data**

Which Clustering Algorithm?

- Type of Clustering
- Type of Cluster
  - Prototype vs connected regions vs density-based
- Characteristics of Clusters
- Characteristics of Data Sets and Attributes
- Noise and Outliers
- Number of Data Objects
- Number of Attributes
- Algorithmic Considerations



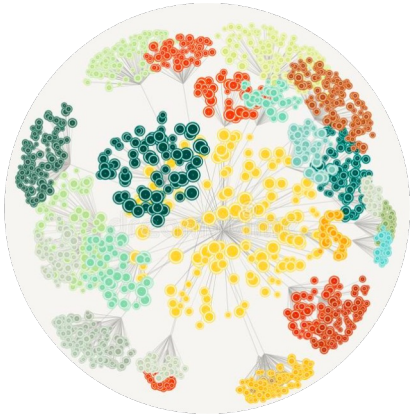
A Comparison on Clustering Algorithms

Criteria	Hierarchal clustering	K-mean	K-mediod	DBSCAN
Initial condition	No	Yes	Yes	Yes
Termination condition	Not precise	Precise	Precise	Precise
Arbitrary value	No requirement	Numeric attribute	Numeric attribute	Numeric attribute
Effect on Size of data sets	Not good	Good	Not good	Not good
Shape of data set	Arbitrary	Convex	Convex	Arbitrary
Granularity	Flexible	K and initial point	K and initial point	Threshold
Result optimization	Optimization	Rebuild optimization	Rebuild optimization	Rebuild optimization
Handling dynamic data	No	Yes	Yes	Yes
Behavior on noisy data	No influences	Influences	Influences	Not much influences
Distance measurement	Any	Distance at normal space	Distance at normal space	Density
Implementation	Simple	Simple	Complicated	Simple

Source: Text Clustering Algorithms: A Review

Summary

- **General Concepts of Clustering**
  - Definition
  - Real-life Applications
  - Types of Clustering
- **Typical Clustering Algorithms**
  - K-Means
  - HAC
  - DBSCAN

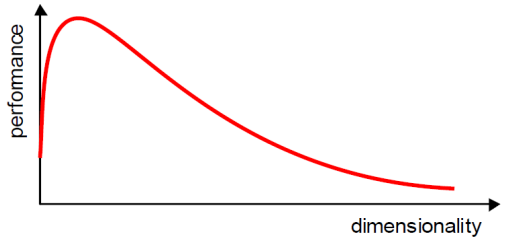


Dimensionality Reduction

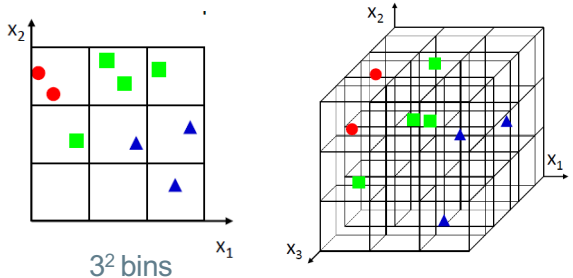
Curse of Dimensionality

The number of training examples required increases exponentially with dimensionality **d** (i.e., $k^{\mathbf{d}}$).

We have to choose the right set features



3^1 bins



3^2 bins

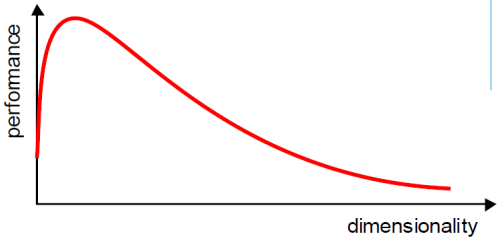
3^3 bins

k: number of bins per feature

Dimensionality Reduction

What is the objective?

Choose an optimum set of features of lower dimensionality to improve classification accuracy.



Dimensionality Reduction

Feature extraction: Extract features from sample (increase features)

Feature selection: get relevant features from the set of features

Dimensionality Reduction: get proper mapping to a lower dimensional space

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ . \\ . \\ x_{i_K} \end{bmatrix}$$

K<<N

The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

K<<N

Dimensionality Reduction

Linear combinations are faster, and easy to optimize

Given $x \in R^N$, find an N x K matrix U such that:

$$y = U^T x \in R^K \text{ where } K<N$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

U^T

This is a projection from the N-dimensional space to a K-dimensional space.

Principal Component Analysis (PCA)

- Let recall the linear algebra, assume a data point $x \in R^N$ as a linear combination of an orthonormal set of N basis vectors $<v_1, v_2, \ldots, v_N>$ in R^N :

$$v_i^T v_j = \begin{cases} 1 & if\ i = j \\ 0 & otherwise \end{cases}$$

$$\mathbf{x} = \sum_{i=1}^{N} x_i v_i = x_1 v_1 + x_2 v_2 + \ldots + x_N v_N$$

$$where \quad x_i = \frac{\mathbf{x}^T v_i}{v_i^T v_i} = \mathbf{x}^T v_i$$

- PCA seeks to represent **x** in a new space of lower dimensionality using only K basis vectors (K < N)

$$\hat{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K$$

such that $\| \mathbf{x} - \hat{\mathbf{x}} \|$ is minimized, for all $x \in D$

(i.e., minimize information loss)

Principal Component Analysis (PCA)

How should we determine the "optimal" lower dimensional space basis vectors $<u_1, u_2, \ldots, u_K>$?

The optimal space of lower dimensionality can be defined by:

(1) Finding the eigenvectors u_i of the covariance matrix of the data Σ_x

$$\Sigma_x \, u_i = \lambda_i \, u_i$$

(2) Choosing the K "largest" eigenvectors u_i (corresponding to the K "largest" eigenvalues λ_i)

PCA - Steps

Suppose we are given $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M$ N x 1 vectors

Step 1: compute sample mean

$$\bar{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i$$

Step 2: subtract sample mean (i.e., center data at zero)

$$\Phi_i = \mathbf{x}_i - \bar{\mathbf{x}} \qquad \hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

Step 3: compute the sample covariance matrix Σ_x

$$\Sigma_{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T = \frac{1}{M}AA^T$$

where $A=[\Phi_1\ \Phi_2\ ...\ \Phi_M]$

(N x M matrix)

PCA - Steps

Step 4: compute the eigenvalues/eigenvectors of Σ_x

$$\Sigma_x u_i = \lambda_i u_i$$

Assume that $\lambda_1 > \lambda_2 > \ldots > \lambda_N$ *and* $u_1, u_2, \ldots, u_N$

Question: How can we say about $\lambda_N > 0$?

Since Σ_x is symmetric, $\langle u_1, u_2, \ldots, u_N \rangle$ form an orthogonal basis in R^N, therefore:

$$\mathbf{x} - \bar{\mathbf{x}} = \sum_{i=1}^{N} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_N u_N$$

$$y_i = \frac{(\mathbf{x} - \bar{\mathbf{x}})^T u_i}{u_i^T u_i} = (\mathbf{x} - \bar{\mathbf{x}})^T u_i \qquad if \;\; \| u_i \| = 1$$

(normalized)

PCA - Steps

Step 5: Aprpoximation with the Tranformation matrix U (using the first K vectors)

$$\mathbf{x} - \overline{\mathbf{x}} = \sum_{i=1}^{N} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_N u_N$$

$$\hat{\mathbf{x}} - \overline{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K$$

or $$(\hat{\mathbf{x}} - \overline{\mathbf{x}}) = U \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$ *where* $U = [u_1 u_2 \ldots u_K]$ $N \times K$

Example

Compute the PCA for dataset

(1,2),(3,3),(3,5),(5,4),(5,6),(6,5),(8,7),(9,8)

Compute the sample covariance matrix is:

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

$$\Sigma_x = \begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}$$

The eigenvalues can be computed by finding the roots of the characteristic polynomial:

$$\Sigma_x v = \lambda v \Rightarrow |\Sigma_x - \lambda I| = 0$$
$$\Rightarrow \begin{vmatrix} 6.25 - \lambda & 4.25 \\ 4.25 & 3.5 - \lambda \end{vmatrix} = 0$$
$$\Rightarrow \boldsymbol{\lambda_1 = 9.34; \lambda_2 = 0.41}$$

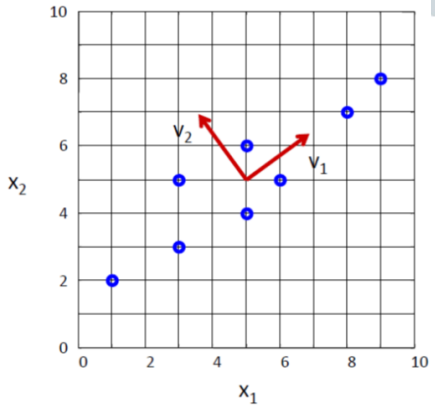
Example

The eigenvectors are the solutions of the systems:

$$\Sigma_{\mathbf{x}} u_i = \lambda_i u_i$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} \lambda_1 v_{11} \\ \lambda_1 v_{12} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = \begin{bmatrix} 0.81 \\ 0.59 \end{bmatrix}$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} \lambda_2 v_{21} \\ \lambda_2 v_{22} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} -0.59 \\ 0.81 \end{bmatrix}$$



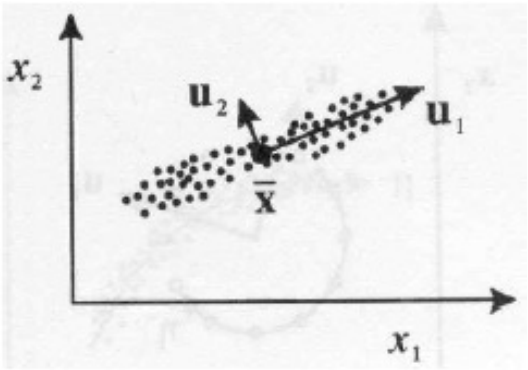
Normalize the eigenvectors vectors to unit-length.

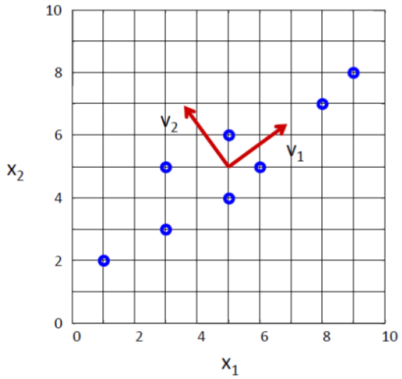
Note: if u_i is a solution, then (cu_i) is also a solution where c is any constant.

Geometric interpretation of PCA

- PCA chooses the eigenvectors of the covariance matrix corresponding to the largest eigenvalues.
- The eigenvalues correspond to the variance of the data along the eigenvector directions.
- Therefore, PCA projects the data along the directions where the data varies most.

u_1: direction of max variance
u_2: orthogonal to u_1





How do we choose K ?

- *K* is typically chosen based on how much information (variance) we want to preserve:

$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > T \quad \textit{where } T \textit{ is a threshold } (e.g., 0.9)$$

- If T=0.9, for example, we say that we "preserve" 90% of the information (variance) in the data.

- If K=N, then we "preserve" 100% of the information in the data (i.e., just a change of basis)

Approximation Error

- The approximation error (or reconstruction error) can be computed as:

$$|| \mathbf{x} - \hat{\mathbf{x}} ||$$

where $\hat{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K + \bar{\mathbf{x}}$ (reconstruction)

- It can also be shown that the approximation error can be computed as follows:

$$|| \mathbf{x} - \hat{\mathbf{x}} || = \frac{1}{2} \sum_{i=K+1}^{N} \lambda_i$$

Thanks!

trongld@vnu.edu.vn