Consensus Clustering

Javier Béjar

URL - 2024 Spring Term

CS - MAI

- The ensemble of classifiers is a well established strategy in supervised learning
- Unsupervised learning aims the same goal: Consensus clustering or Clustering ensembles
- The idea is to merge complementary perspectives of the data into a more stable partition

- Given a set of partitions of the same data $\mathcal{X}$:

$$\mathbb{P} = \{P^1, P^2, \ldots, P^n\}$$

with:

$$\begin{aligned} P^1 &= \{C^1_1, C^1_2, \ldots, C^1_{k_1}\} \\ &\vdots \\ P^n &= \{C^n_1, C^n_2, \ldots, C^n_{k_n}\} \end{aligned}$$

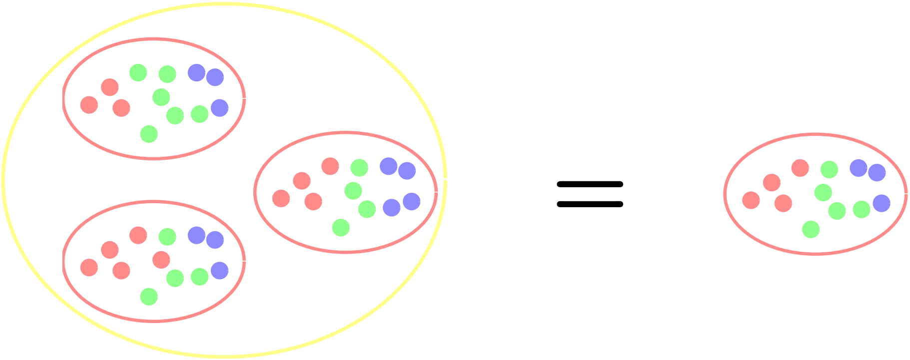
to obtain a new partition that uses the information from all n partitions

- **Robustness**, the combination has a better performance than each individual partition in some sense
- **Consistency**, the combination is similar to the individual partitions
- **Stability**, the resulting partition is less sensitive to outliers and noise
- **Novelty**, the combination is able to obtain different partitions that can not be obtained by the clustering methods that generated the individual partitions

- **Knowledge reuse**, the consensus can be computed from the partition assignments, so previous partitions using the same or different attributes can be used
- **Distributed computing**, the individual partitions can be obtained independently
- **Privacy**, only the assignments of the individual partitions are needed for the consensus

Consensus Process

- Consensus clustering is based generally in a two steps process:
  1. Generate the individual partitions to be combined
  2. Combine the partitions to generate the final partition



- **Different example representations**: Diversity by generating partitions with different subsets of attributes
- **Different clustering algorithms**: Take advantage that all clustering algorithms have different biases
- **Different parameter initialization**: Use clustering algorithms able to produce different partitions using different parameters
- **Subspace projection**: Use dimensionality reductions techniques
- **Subsets of examples**: Use random subsamples of the dataset (bootstrapping)

- **Coocurrence based methods**: Use the labels obtained from each individual clustering, and the coincidence of the labels for the examples
  - Relabelling and voting, co-association matrix, graph and hyper-graph partitioning, information theory measures, finite mixture models
- **Median partition based methods**: Given a set of partitions ($\mathcal{P}$) and a similarity function ($\Gamma(P_i, P_j)$), find the partition (P_c) that maximizes the similarity to the set:

$$P_c = \arg\max_{P \in \mathbb{P}_x} \sum_{P_i \in \mathcal{P}} \Gamma(P, P_i)$$

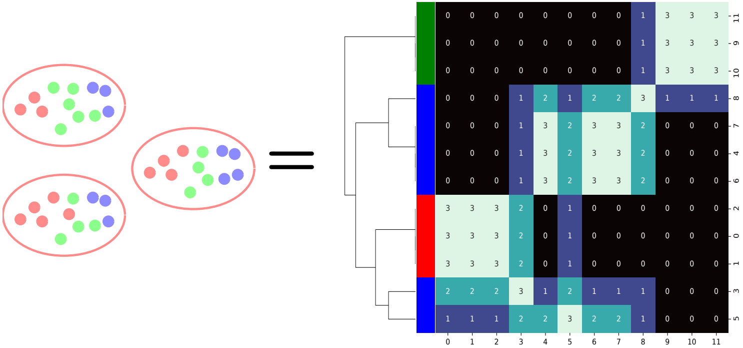
Coocurrence based methods

- First, solve the labeling correspondence problem
- After that, determine the consensus using different voting strategies

Dimitriadou Weingessel, Hornik **Voting-Merging: An Ensemble Method for Clustering**
Lecture Notes in Computer Science, 2001, 2130

1. Generate a clustering
2. Determine the correspondence with the current consensus
3. Each example gets a vote from their cluster assignment
4. Update the consensus

- **Co-Association matrix**: Count how many times a pair of examples are in the same cluster
- Use the matrix as a similarity or a new set of characteristics
- Apply a cluster algorithm to the information from the co-association matrix



- Define consensus as a graph partitioning problem
- Different methods to build a graph or hyper-graph from the partitions

Strehl, Ghosh **Cluster ensembles- A knowledge reuse framework for combining multiple partitions** Journal of Machine Learning Research, MIT Press, 2003, 3, 583-617

- Cluster based Similarity Partitioning Algorithm (CSPA)
- HyperGraph-Partitioning Algorithm (HGPA)
- Meta-CLustering Algorithm (MCLA)

- Compute a similarity matrix from the clusterings
- **Hyper-edges matrix**: For all clusterings, compute an indicator matrix (H) that represents the links among examples and clusters (Hyper-graph)
- Compute the similarity matrix as:

$$S = \frac{1}{r} H H^T$$

  where r is the number of clusterings
- Apply a graph partitioning algorithm to the distance matrix (METIS)
- Drawback: Quadratic cost in the number of examples $O(n^2 kr)$

	C_1	C_2	C_3
x_1	1	2	1
x_2	1	2	1
x_3	1	1	2
x_4	2	1	2
x_5	2	3	2

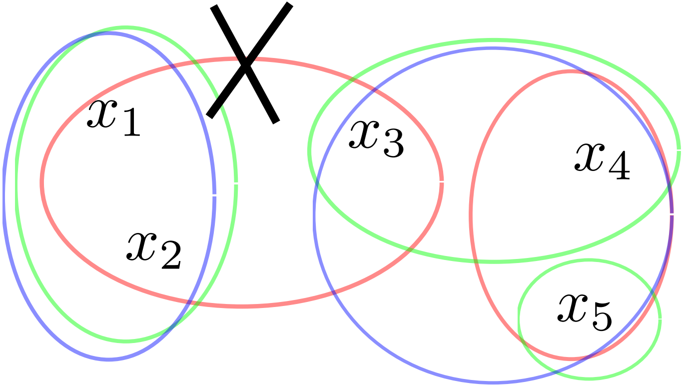
$\Longrightarrow$

	$C_{1,1}$	$C_{1,2}$	$C_{2,1}$	$C_{2,2}$	$C_{2,3}$	$C_{3,1}$	$C_{3,2}$
x_1	1	0	0	1	0	1	0
x_2	1	0	0	1	0	1	0
x_3	1	0	1	0	0	0	1
x_4	0	1	1	0	0	0	1
x_5	0	1	0	0	1	0	1

$S =$

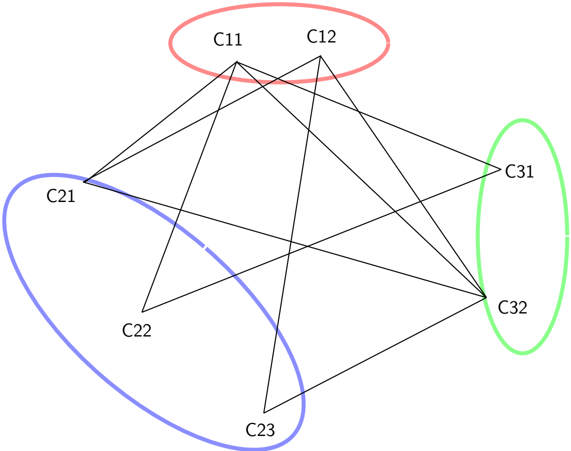
	x_1	x_2	x_3	x_4	x_5
x_1	1	1	1/3	0	0
x_2	1	1	1/3	0	0
x_3	1/3	1/3	1	2/3	1/3
x_4	0	0	2/3	1	2/3
x_5	0	0	1/3	2/3	1

- Partitions the hyper-graph generated by the examples and their clusterings
- The indicator matrix is partitioned into k clusters of approximately the same size
- The HMETIS hyper-graph partitioning algorithm is used
- Linear in the number of examples $O(nkr)$



- Group and collapse hyper-edges and assign the objects to the hyper-edge in which they participate the most
- Algorithm
  1. Build a meta-graph with the hyper-edges as vertices (edges have the vertices similarities as weights, Jaccard)
  2. Partition the hyper-edges into k meta-clusters
  3. Collapse the hyper-edges of each meta-cluster
  4. Assign examples to their most associated meta-cluster
- Linear in the number of examples $O(nk^2r^2)$

	$C_{1,1}$	$C_{1,2}$	$C_{2,1}$	$C_{2,2}$	$C_{2,3}$	$C_{3,1}$	$C_{3,2}$
x_1	1	0	0	1	0	1	0
x_2	1	0	0	1	0	1	0
x_3	1	0	1	0	0	0	1
x_4	0	1	1	0	0	0	1
x_5	0	1	0	0	1	0	1



Median partition based methods

- Given a set of partitions ($\mathcal{P}$) and a similarity function among partitions $\Gamma(P_i, P_j)$, the **Median Partition** P_c is the one that maximizes the similarity to the set

$$P_c = \arg\max_{P \in \mathbb{P}_x} \sum_{P_i \in \mathcal{P}} \Gamma(P, P_i)$$

- Has been proven to be a NP-hard problem for some similarity functions Γ

- Based on the agreements and disagreements of pairs of examples between two partitions
  - Rand index, Jaccard coefficient, Mirkin distance (and their randomness adjusted versions)
- Based on set matching
  - Purity, F-measure
- Based on information theory measures (how much information two partitions share)
  - NMI, Variation of Information, V-measure

- Best of k (the partition of the set that minimizes the distance)
- Optimization using local search: Hill Climbing, Simulated Annealing, Genetic Algorithms
  - Perform a movement of examples between two clusters of the current solution to improve the partition
- Non Negative Matrix Factorization
  - Find the partition matrix closest to the averaged association matrix of a set of partitions

This Python Notebook has examples of consensus clustering

- Consensus clustering Notebook (click here to open the notebook in colab)

If you download the notebook you will be able to use it locally (run jupyter notebook to open the notebooks)