1 Clustering Methods

Clustering methods are used to group data when no class labels are present. You thereby want to learn an intrinsic structure of the data.

1.1 K-Means Clustering

Goal: Divide data into K clusters so that the variance within the clusters is minimized. The objective function:

$$V(D) = \sum_{i=1}^{k} \sum x_j \in C_i (x_j - \mu_i)^2, \tag{1}$$

where V is the variance, C_i is a cluster, μ_i is a cluster mean, x_j is a datapoint. The algorithm works as follows:

1. Assign the data to k initial clusters
2. Calculate the mean of each cluster
3. Assign the data points to the closest cluster mean
4. If a point changed its cluster, repeat from step 2

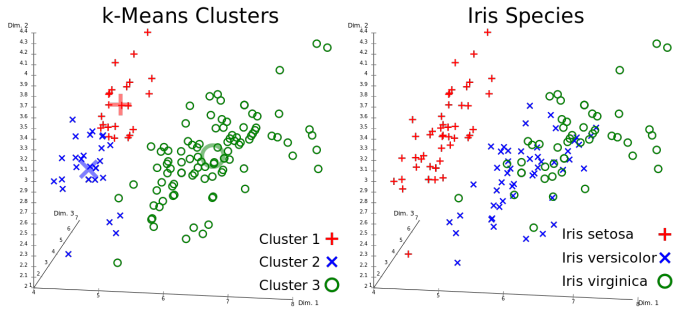

Fig. 1: Data from the *Iris flower data set* clustered into 3 clusters using k-Means. On the right the data points have been assigned to their actual species. *Figure from user Chire on wikimedia.org.*

1.2 Graph-Based Clustering

You represent data set D as a graph $G = (V, E)$ and divide it up in connected sub-graphs that represent your clusters. Each edge e_{ij} (between nodes v_i and v_j) has a weight w_{ij} (which is commonly a similarity or distance measure).

Basic Graph-Based Clustering The basic algorithm works like this:

1. Define a weight-threshold θ
2. For all edges: if $w_{ij} > \theta$: remove e_{ij}
3. If nodes are connected by a path (via *depth first search*): Assign are them to the same cluster

DBScan *Density-Based Spatial Clustering of Applications with Noise* is a more noise robust version of graph-based clustering.

Cut-Based Clustering You introduce a **adjacency/similarity** matrix W (measures similarity between data points) and define the number of clusters k. You now try to minimize the weight of edges between the clusters (equal to cutting edges between nodes that are least similar):

$$\min \frac{1}{2} \sum_{a=1}^{k} \sum_{b=1}^{k} \kappa(C_a, C_b)$$
$$\text{where } \kappa(C_a, C_b) = \sum_{v_i \in C_a, v_j \in C_b, a \neq b} W_{ij}$$
$$\text{and } \kappa(C_a, C_a) = 0$$

$\rightarrow$ You only add up the similarities/edge-weights between your clusters (but not within your clusters).

Q $\rightarrow$ For constructing the similarity matrix, different kernels can be used (commonly the linear kernel or the Gaussian kernel).

1.3 Spectral Clustering

The goal is again, to cluster the points, such that the similarity of points of different clusters is minimal. Spectral clustering employs three steps: Preprocessing, decomposition and grouping.

Preprocessing We create a Laplacian matrix L (laplacian operator in matrix form, measuring how strongly a vertex differs from nearby vertices (because the edges are similarity measures)):

$$L = D - W$$
$$D_{ij} = \begin{cases} \sum_{j=1}^{N} W_{ij} \\ 0 \text{ if } i \neq j \end{cases}$$

where D is the degree matrix (the degree of each node is on the diagonal).

Decomposition The class assignment is encoded in vectors c_k. We normalize them and get e.g.:

$$u_a = \frac{c_a(i)}{||c_a||} = \begin{cases} \frac{1}{||c_a||} \text{ if } v_i \in C_a \\ \frac{0}{||c_a||} \text{ if } v_i \notin C_a \end{cases}$$

As in cut-based clustering we search for a minimum k-cut, but also add a constraint that $u_a^T u_a = 1$

$$\min \frac{1}{2} \sum_{a=1}^{k} u_a^T L u_a + \lambda_a \sum_{a=1}^{k} 1 - ||u_a||^2$$

We find the optimal u_a by eigenvalue decomposition:

$$Lu_a = \lambda u_a$$

The final data is now represented as a matrix of k eigenvectors (of the least dominant eigenvalues) as column-vectors.

Grouping You get the final cluster assignments by normalizing the now k-dimensional data and applying k-means clustering to it.

1.3.1 Sparse Subspace Clustering (SSP)

The underlying assumption of SSP is that the different clusters reside in different subspaces of the data. Clusters are therefore perpendicular to each other and points in a cluster can only be reconstructed by combinations of points in the same cluster ($\rightarrow$ self-expressiveness, the reconstruction vectors ought to be sparse). For each point you try to find other points that can be used to recreate that point - these then form the same cluster. Doing that for all points gives you a data matrix X and a matrix of reconstruction vectors V:

$$X = X * V \text{ s.t. } \text{diag}(V) = 0.$$

You now try to minimize the V-matrix according to the L1-norm (giving you a sparse matrix). This matrix can then be (multiplied with its transpose and) used for e.g. spectral clustering.

1.4 Soft-assignment Clustering

1.4.1 Expectation Maximization (EM) Clustering

[Coming soon]

1.4.2 Gaussian Mixture Models

[Coming soon]

1.5 Unsupervised Multiple Kernel Learning (UMKL)

See chapter **??**.

1.6 Artificial Neural Networks for Clustering

See chapter *Neural Networks* (**??**)