Sommario

Clustering

- Broad set of techniques used to find unknown subgroups, or clusters, in a dataset
- Require the definition of a dissimilarity (distance) measure
- Process of dividing a set of input data into possibly overlapping subsets, where elements in each subset are considered related by some similarity measure.
- Allow to group observations according to their features characteristics
- Allow to group features/predictors/variables according to similarities in the observations
- Usually, observations are clustered on the basis of the features in order to identify subgroups among the features. However, it is also possible to cluster the features on the basis of the observations (to discover subgroups among the features) simply transposing the data matrix

Agglomerative clustering:
- Bottom up clustering
- Each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy

Divisive clustering
- Top down clustering
- All observations start in one cluster, and splits are performed recursively as one moves down the hierarchy

Hard clustering

- Each datapoint belongs to only 1 cluster

Soft / Fuzzy clustering

- Each datapoint can belong to multiple clusters
- Datapoints are not explicitly assigned to clusters, but they receive the probabilities to belong to the cluster
- Degree of membership: a probability or likelihood of the data point to be in a specific cluster
- Algorithms:
  - Fuzzy c-means
  - Mixture models
  - Dirichlet process mixtures
  - HDBSCAN

Parametric clustering

- AMOEBA
- K-means, K-median
- K-medoids
- Self-Organizing Maps
- Hierarchical clustering
- Mixture Models

Non-parametric clustering

- Dirichlet process mixtures (based on non-parametric density estimation)
- Mean shift clustering

Density-based models

- DBSCAN
- OPTICS
- HDBCAN

Similarity-based / Graph clustering

- Spectral clustering
- Affinity propagation
- Graph cuts

Hybrid clustering

- Consist mixing different clustering along the modelling chain previous to obtain the final cluster solution.
- Example:
  - Starting using k-means to get few tens of cluster, then apply hierarchical clustering on each of those cluster to find additional clusters
  - Start with a large SOM and then cluster together the neurons of the SOM with another clustering algorithm

Algorithm where the number of cluster K must be not specified

- DBSCAN & HDBSCAN
- Affinity propagation
- Mean shift
- Dirichlet Process

Clustering in practice

Finding the number of clusters k

- Run the clustering algorithm for different k and inspect changes in ‘cluster goodness’ metrics
- ‘Cluster goodness’ metrics:
  - Silhouette coefficient
  - Dunn Index
  - Gap statistic
  - Intra / Inter cluster distance
  - Connectivity index
  - **Others**

General considerations

- The choice if the variables should be centered, scaled to have standard deviation 1 or standardized must be carefully thoughts because impacts some dissimilarity measure!!!
- Comparing clusters with respect to an additional predictor/feature not considered in the formation of the clusters can help in the interpretation of clusters
- Since clustering can be non-robust, it is recommended to cluster subsets of data in order to get a sense of robustness of the clusters obtained.

Outliers considerations

- K-means and hierarchical clustering force every observation into a cluster. Thus, outliers may heavily distort the clusters!
- Mixture models accommodate better to the presence of such outliers.

Big data considerations

- k-means and SOM algorithm are very fast, well-suited to large data sets
- SOM allow for online training (subset of data at a time)
- Hierarchical clustering and distance-matrix based methods are not very adequate because the distance matrix is computationally too expensive

High-dimensional data considerations

- All datapoints can become equidistant from all other points so Euclidean distance is not an appropriate metric

Assumption of the shape of clusters (table…TODO)

Computational performance

- K-means, SOM > DBSCAN, HDBSCAN > Hierarchical clustering > Spectral Clustering > Affinity Propagation

Clustering evaluation

- Cluster stability by bootstrapping data
- Cluster stability using different clustering algorithms
- Confusion matrix (contingency table) between 2 clustering algorithms
- Cluster agreement
  - Entropy of observations
  - Purity
  - Mutual information
  - Normalized mutual information (NMI)
  - Adjusted Rand Index (ARI) (Yeung & Ruzzo 2001)
  - Rand index
  - Matching index
  - **FPC R package**

Normalized Mutual Information (NMI)

- A symmetric measure for the degree of dependency between the clustering and the manual classification.
- Based on the notion of cluster purity pi, which measures the quality of a single cluster Ci

K-means

- Hard clustering
- The number of clusters k must be specified
- The classical algorithm uses the squared Euclidean distance as dissimilarity measure
- Partition the observations into K distinct non-overlapping globular clusters
- Originally also called Forgy algorithm
- Conceptual idea: Little difference between observations belonging to the same cluster and big difference between observations of different clusters
- The within-cluster variation (WCSS) is a measure of the amount by which the observations within a cluster differ from each other.
- K-means try to minimize the total WCSS (The sum of all clusters WCSS)
- Founding the partitioning which minimize the total WCSS is a very difficult optimization problem since there are almost k^n partitioning possibilities
- K-means applies a (simple) iterative algorithm to find a local optimum and the procedure is repeated many times to hopefully find the global optimum. Such type of optimization procedure is called greedy (greedy approach) (sub-optimal optimization)
- The clustering results are dependent upon initialization
- K-means may often converge to a local minimum (sub-optimal solution)
- Bad initialization can lead to:
  - Poor convergence speed
  - Bad overall clustering
- In complex high-dimensional problems, the solution may vary according to the initial values!

- Find the k clusters centers and assign the objects to the nearest cluster center (mean=centroid), such that the sum of squared distances within each cluster (WCSS) are minimized
- Clusters the data into n clusters of equal variance minimizing the within-cluster Euclidean distance
- Cluster the observations according to the Voronoi cells generated by the cluster means (centroids)

- Use cases:
  - Clusters have globular shape
  - If the dataset has outliers, such points are lumped in the clusters and may impact the correct formation of clusters
- Advantages
  - High performance and applicability to big data
  - Easily customizable: different distance metrics, k-medians, robust WCSS
- Disadvantage
  - The algorithm seeks and identifies globular (essentially spherical) clusters
  - Tends to find clusters of comparable spatial extent
  - Can underperform with clusters of different size and density
  - Gives more “weight” to larger cluster (because they contribute more into the MSE)
  - Small clusters end-up far away from any center, while it uses those centers to “split up a much larger cluster.

Fuzzy C-Means

- Soft clustering
- The number of clusters k must be specified
- The algorithm tries to figure out the membership fraction that minimize the expected distance to each centroid.

K-medoids

- Hard clustering
- The number of clusters k must be specified
- *k*-medoids chooses randomly datapoints as centers
- Attempt to minimize the distance between points labeled to be in a cluster and datapoints designated as the center of that cluster (medoid).
- Works with a generalization of the Manhattan Norm to define the distance between datapoints
- Is more robust to noise and outliers as compared to *k*-means because it minimizes a sum of pairwise dissimilarities instead of a sum of squared Euclidean distances.
- The most time-consuming part of the k-medoids algorithm is the calculation of the distances between objects
- The distances matrix can be precomputed to achieve consequent speed-up.
- Partitioning Around Medoids (PAM)

AMOEBA

- Hard clustering
- Clustering based on Delaunay triangulation
- Partitioning algorithm
- Require definition of a distance threshold
- Clusters have Delaunay edges which considerably differ from the mean length of all Delaunay edges
- Differences are only interesting if they are shorter than the mean; the other points can be interpreted as noise
- Points of no interest (noise points) should be removed, such that further clusters can be detected within the cluster (sub clusters)

Hierarchical Clustering

- Also called bottom-up (agglomerative) linkage clustering
- Hard clustering
- The number of clusters k must be specified
- Sequentially reduce the number of clusters by 1, unifying two cluster or observations
- Clusters are sequentially enlarged
- Provide a tree-based visual representation of the observations (dendrogram)
- One single dendrogram can be used to obtain any number of clusters between 1 and n
- The height value represents the dissimilarity value between two (groups) observations
- The height of cut of the dendrogram controls the number of clusters obtained
- Observations that fuse at the very bottom of the tree are quite similar to each other
- Observations that fuse to the top of the tree tend to be quite different
- Conclusions about the similarity of two (groups of) observations must be done only based on the location on the vertical axis!
- Linkage defines the dissimilarity between two groups of observations:

- Complete: Use the largest pairwise dissimilarity between the two groups of observations. Tends toward the formation of small (respectively equally sized) clusters
- Single: Use the small pairwise dissimilarity between the two groups of observations Tends to yield “trailing” clusters: very large cluster onto which individual observations attach one-by-one (highly skewed dendrograms)
- Average: The distance between 2 clusters is the average of all pairwise dissimilarities between the two groups of observations
- Centroid: Use the dissimilarity between the centroids of the two groups of observations. The centroid of each new formed cluster is the (un/weighted) mean of the clustered centroids. Weighting may be used to account for different size of the unified clusters!
- Ward: Clusters the observations which results in the smallest increase of the total sum of squared deviation (dissimilarity) from the mean of each cluster

* Linkage does not influence leaf-to-leaf fusion
* Usually complete and average linkage tend to yield balanced and attractive clusters!
* Single linkage usually does not reflect the true underlying data structure (unless the clusters are well-separated)

Procedure:

1. Compute the distance matrix
2. Cluster together the observations/clusters with smallest distance
3. Recompute the distance matrix for the new clusters/observations
4. Repeat 2-3 until all observations are clustered in a single cluster

Dendrogram:

* Y axis the value of the dissimilarity at the fusion step
* X axis the n observations that are grouped with each other in different clusters

Use cases:

* Clusters are not necessarily circular (or hyper-spherical)
* The number of clusters is not known in advance

Disadvantages

* Clusters obtained by cutting the dendrogram at a given height are necessarily nested within the clusters obtained by cutting the dendrogram at any greater height. This hierarchical structure assumption might be unrealistic
* The computation of the distance matrix is expensive with large datasets

DBSCAN

* Density-Based Clustering (of Application with Noise)
* Hard clustering
* The number of clusters k is determined by the algorithm
* Does not require that every point be assigned to a cluster
* Groups together datapoints that are closely packed together
* Clusters are intended as regions of high data point density
* Extracts the ‘dense’ clusters and leaves sparse background classified as ‘noise’.
* Algorithm parameters
  - epsilon: the size of the neighbourhoods (‘reachability' distance)

  - minPts: the minimum number of points for the resulting set to be considered a cluster
- Datapoints that lack neighbours (the number of neighbours is lower than Minpoints) do not belong to any cluster and are classified as noise
- Algorithm
  - DBSCAN start from a random datapoint
  - It checks if in the 'epsilon'-defined neighbourhoods there are more than 'minPts' datapoints. If yes, such datapoints are all assigned to a cluster and their neighbourhoods are scanned to see if other datapoints can be added to the cluster.
  - Then DBSCAN moves to the other unvisited datapoints. If a new datapoint as enough neighbours and these neighbours are not already assigned to another cluster, it creates a new cluster
  - Datapoints that does not have enough neighbours are classified as noise.

Interpretation
- DBSCAN transforms the space according to the density of the data and then applies agglomerative single linkage clustering to the transformed space.
- The dendrogram of hierarchical clustering applied in this transformed space can be used to define 'the epsilon' parameter (the distance at which to cut)

Advantages
- Minimal requirements of domain knowledge to determine input parameter
- Discovery of clusters with arbitrary shape
- Can deal with different cluster shapes
- Good efficiency on large databases

Disadvantage
- Weak in detecting meaningful clusters in dataset with varying density in the feature/observation/geographic space
- Border points can flip between clusters, depending on the order by which the data are processed, meaning different executions can return different outputs
- DBSCAN suffers from the curse of dimensionality. Distance functions become less meaningful in higher dimensions, as all points are 'far away' from each other.
- In high-dimension it can be hard to determine the appropriate values of epsilon and minPts
- If the clusters exhibit varying density, it may be impossible to find a decent epsilon value, as one single value can't perform well on each cluster.

OPTICS

- Ordering Points To Identify the Clustering Structure
- Solve some problems of DBSCAN when clusters exhibit varying density

HDBSCAN

- Improves DBSCAN when clusters have varying density
- Extends DBSCAN by converting it into a hierarchical clustering algorithm (in the density-based transformed space)
- The dendrogram can used to select the most stable or persistent clusters
- The 'epsilon' parameter of DBSCAN is no longer needed

Mixture models / Model-based (EM) clustering

- Soft clustering
- The number of clusters k must be specified
- EM = Expectation maximization
- EM estimates some latent variables so as to maximize the log-likelihood of the observed data

Gaussian mixture model

- A probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters
- The latent variables are the mean and covariance matrix of a multivariate normal distribution
- The probability that a point belongs to a cluster is given by a multivariate Gaussian probability distribution
- Each cluster is described by its centroid (mean), covariance (so that we can have elliptical clusters), and weight (the size of the cluster)

Disadvantages

- The underlying probabilistic distribution must be specified, which is often irretrievable especially for large and high-dimensional datasets

Dirichlet process mixtures

- Soft clustering
- The number of clusters k is determined by the algorithm
- Generalization of mixture models
- Example: GMM with an infinity of components

Mean Shift Clustering

- Hard clustering
- The number of clusters k is determined by the algorithm
- The underlying idea of the Mean Shift Clustering is that there exists some probability density function from which the data are drawn, and the algorithms tries to place the cluster centroids at the maxima of that density functions.
- Mean Shift describes a general non-parametric technique that move particles in the direction of local increasing density.
- Mean Shift Clustering simply refers to its application to the task of clustering
- Requires the specification of a kernel bandwidth parameter which determines the size of neighborhood over which the density is computed
- The density equates to the number of points inside each kernel
- The cluster centers are identified as the region with local maxima density
- The kernel bandwidth parameter influences convergence rate and the number of clusters
- Kernel types: flat/uniform, gaussian, …
- $O(n^2)$ but embarrassingly parallelizable
- Adaptive mean shift clustering: the bandwidth varies for each datapoint
- Advantages
  - The number of clusters is not required
  - No assumptions on the shape of clusters
- Disadvantages
  - The bandwidth parameter is difficult to tune
  - Without visual validation it can be hard to know how wrong it may be

Spectral Clustering

- Hard clustering
- The number of clusters k must be specified
- Require the definition of a symmetric similarity matrix
- Can best be thought of as a graph clustering
- It looks at the eigenvectors of the Laplacian of the graph to attempt to find a good (low dimensional) embedding of the graph into Euclidean space.
- Once data are projected into the new space, usually k-means is applied

- Pairs of observations (vertices) are connected (by an edge) if their similarity exceeds a specified threshold
- Edges are weighted by s_{ij}
- Outliers become separate clusters
- Use case:
  - Arbitrary cluster shapes
- Disadvantage

- Computation of the similarity matrix is expensive with large datasets

Affinity Propagation

- Hard clustering
- K is determined by the algorithm
- Use datapoints as representative of clusters (like in k-medoids)
- Requires the definition of a similarity matrix S (also non-symmetric!)
- Takes as input measures of similarity between pairs of data points, and simultaneously considers all data points as potential exemplars
- Chooses the number of clusters based on the similarity matrix.
- The number of clusters can be adjusted modifying the preferences

Similarities

- Represent how well-suited a point is to another
- If there is no similarity between two datapoints, they cannot belong to the same cluster

Preferences

- The diagonal elements of the similarity matrix
- Determines how likely a datapoint is to become an exemplar
- The datapoint suitability to be an exemplar
- Adjustment of the preferences modify the number of clusters
- Larger S(i,i) → more likely the i datapoint will become an exemplar
- Setting preferences to a higher value lead to more clusters. Each point is 'more certain' of its suitability to be an exemplar and is therefore harder to 'beat' and include it under some other point's 'domination'
- Setting preferences to a lower value lead to less clusters
- Is by default usually set to the median value of the outer diagonal similarity matrix values

Availability A(i,k) :

- how much support there is from other items for k to be an exemplar
- how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar

Responsibility R(i,k) :

- how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i
- how fit is k to represent i, as compared to another possible candidate k

Clustering steps

- The algorithm runs through a number of iterations, until it converges.
- 'Exchanging messages between points' is the same thing as manipulating matrices, and it's only a matter of perspective and implementation.
- Initialization:
  - All availabilities are set to zero
- Each iteration has two message-passing steps:
  1. Calculating responsibilities R(i,k) and sent it from data point i to candidate exemplar point k
  2. Calculate availability A(i,k) and sent from candidate exemplar point k to point i
  3. Define the exemplar:
  - Compute I = A + R
  - For each datapoint i, the k with maximum I(i,k) represents point i's exemplar.
  - Datapoints i which I(i,i) is larger than 0 are exemplars
- Possible stopping criteria

- Changes in values fall below some threshold
- The maximum number of iterations is reached

Use cases:

- Many clusters, uneven cluster size, non-flat geometry

Advantages

- The algorithm is relatively insensitive to high dimensional data if the measure of similarity is robust in higher dimensions
- It allows for non-metric dissimilarities (that don't obey to the triangle inequality)
- Is purely deterministic

Disadvantages

- Not scalable with the large number of datapoints
- The similarity matrix is computationally expansive
- Choosing the correct parameters (preferences) is difficult

Graph Cuts

- Observations: Graph nodes
- Similarity S_{ij} = weight of edge I,j
- Remove edges
- Clusters are the connected components
- A graph-partitioning problem
- Goal: Partition the graph such:
  - edges between different groups have low weight
  - edges within a group have high weight
- Normalized cuts ?
- Ncut algorithm ?

Self-organizing maps (SOM)

- SOMs can be thought of as a spatially constrained form of k-means clustering, where the clusters are topologically constrained and projected into a 2D space
- It is a clustering algorithm with very appealing properties for exploratory data analysis
- Useful to finds representative exemplars of datasets with large observations
- Clusters that are similar (based on Euclidean distance) are represented close to each other
- Cluster = Map unit = Prototype vector = Weight vector = Code(book) vector = Node = Neuron
- Object = Instances = Observations = Datapoints
- Learning rate = Adaptation parameter = Gain
- The "weights" are the values of the SOM units
- Each observation is assigned to the map unit that has the most similar weight vector.
- The SOM unit which has the minimum distance to the observation input vector is called the Best Matching Unit (BMU). During training, when some pattern is presented to the SOM, the SOM unit with most similar weight values is considered the winner (BMU) and its weights (and those of its immediate neighborhood) are adapted to the pattern
- As soon as an observation is assigned to a map unit (cluster), it also changes the cluster's weight vector and the weight vectors of the neighboring clusters located within the radius of r from the winner's map unit.
- The initialization of the clusters' weight vectors can influence remarkably the outcome of the clustering

Problem definition

- (Observation) Input vector: $\mathrm{X_i} = [x_1, x_2, \dots, x_p]$
- SOM unit = Neuron=Cluster = Weight vector = $\mathrm{W_j} = [w_1, w_2, \dots, w_p]$
- Topological constrain = Neighborhood function $h^t(n_m, n_j)$ --> i.e Gaussian RBF kernel
- Parameters: Learning rate $\alpha(t)$, neighborhood size parameter $\lambda(t)$

Training options

- Stochastic, sequential, online SOM: The map units are updated after each single datapoint
- Batch SOM: The map unit are updated after having seen the entire dataset
- Mini-batch: The map units are updated after having seen the batch

Stochastic SOM

1. Sample an observations input vector ($\mathrm{X_i}$) from the dataset
2. Find the best matching unit (BMU) : $\arg\min_j D(W_j, X_i)$ where $D(W_j, X_i) = \sum_{l=1}^{p} a_l\, d_l(w_l, x_l)$
3. Retrieve the neighbors units of the BMU (and assign a weight to them) : $h^t(n_m, n_j) = \exp(\frac{-||n_j - n_m||}{\lambda(t)})$
4. Update the weight vectors: $W^{t+1} = W^t + \alpha(t)\ \boldsymbol{h^t(BMU, n_j)}\ (W^t - X_i)$
5. Repeat points from 1 to 4 by progressively reducing $\alpha(t)$ and $\lambda(t)$

Similar algorithms

- Learning Vector Quantization
  - Also called Simple Competitive Learning algorithm
  - Equivalent to Stochastic SOM when the topological constraints is removed (neighborhood function is set to 1)
- Emergent SOM: when the SOM is made of thousands of map units
- Growing self-organizing map: starts 4 nodes and grows the required new nodes on the boundary

Topological k-means SOM (TKM-SOM) (traditional SOM)

- It's a clustering procedure rather than a dimension reduction technique
- Every map unit (neuron) corresponds to a "cluster" of input observations
- Usually the number of map units is smaller than 30
- The number of clusters is defined by the size of the grid

Emergent SOM

- Could be seen as a dimension reduction procedure rather than a clustering technique
- Clustering algorithms applied on top of ESOM are called emergent
- Differs from a traditional SOM in that a very large number of neurons usually several thousands ($\geq$ 4000) are used
- Considered as a visualization technique
- Tools for visualization of structural features of the data space.
- The structural features of the data space are usually visualized using U-Matrix or P-Matrix

Sammon mapping of SOM nodes … animation
Initialization with PCA can be used only when p >= number of map units
Hypothesis

- SOM is less prone to local optima than k-means.
- The search space is better explored by SOM. This is due to the effect of the neighborhood parameter which forces units to move according to each other in the early stages of the process. This characteristic can be seen as an "annealing schedule" which provides an early exploration of the search space (Bodt, Cottrell et al. 1999).
- When the neighborhood radius is small:
  - the SOM acts more like K-means
  - the clusters close to each other aren't necessary similar
- SOM and k-means algorithms are rigorously identical when the radius of the neighborhood function in the SOM equals zero (Bodt, Verleysen et al. 1997)
- It is said that the topology preservation of the traditional SOM projection is of little use when using small maps: the performance of a small SOM is argued to be almost identical to that of a k-means clustering, with k equal to the number of nodes in the map