INFO411 Lecture 2 - Clustering

Jeremiah Deng

University of Otago

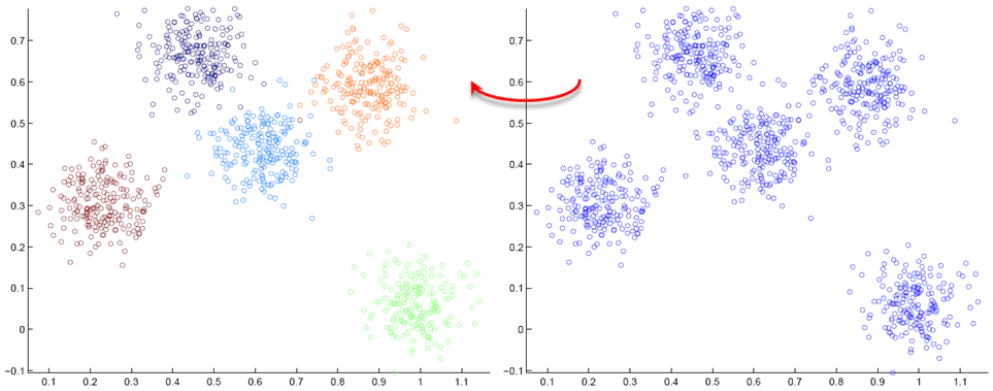
18 July 2017

Outline

What is Clustering?

'Labeling of data clouds' based on similarity

- Partitioning of a data set into subsets (clusters).
- Clusters are unknown a priori (i.e. unsupervised learning).



Main Approaches of Clustering

- Partitioning algorithms: Construct various partitions and then evaluate them by some criterion
- Density-based algorithms: based on connectivity and density functions
- Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model
- Hierarchical algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion

Partitioning Methods

- The idea: Construct a partition of a dataset of data entities into a set of k clusters
- Given a k, find a partition of k clusters that optimizes the chosen partitioning criterion
- Global optimal: exhaustively enumerate all partitions
- Heuristic methods: using a few rules
  - k-means (MacQueen'67): Each cluster is represented by the centre of the cluster
  - k-medoids or PAM (Partition around medoids): Each cluster is represented by one of the data items in the cluster

Clustering as Optimization

- How to partition data points $\mathbf{X}$ into k *clouds*?
- The goal is to optimize a score function that links to the compactness of each cloud.
- The most commonly used is the square error criterion:

$$\mathcal{F} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mathbf{m}_i\|^2$$

- Finding the best $\mathbf{m}_i$: set $\frac{\delta \mathcal{F}}{\delta \mathbf{m}_i} = 0$.
- Easy: given all $\mathbf{x}$ within a cloud, the centroid gives the minimal $\mathcal{F}$: $\mathbf{m}_i = E_{\mathbf{x} \in C_i}(\mathbf{x})$.
  - We proved this in Lecture 1, but is the problem solved?

A Deadlock?

- How do you know about the right number of clusters?
  - k is often unknown
  - Let's just assume a value for k
- However, where shall we start?
  - We need to know the membership of each data point so as to find the centroids.
  - But don't we need to know the centroids to find the right membership for each data point?
- Deadlock?

$\Rightarrow$ A random start will always converge!

- However, convergence is only to a local minimum. ☹

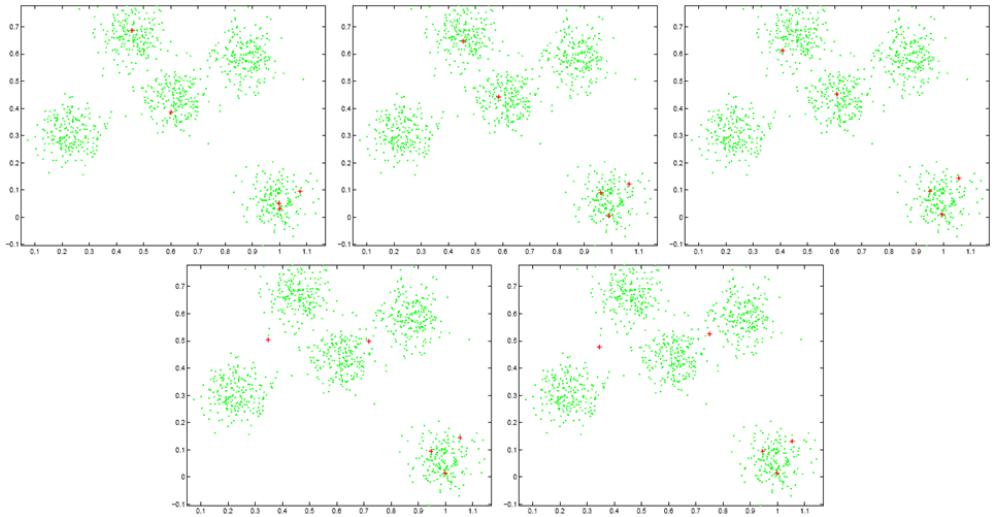
The k-means algorithm

- Given k, the k-means algorithm is implemented in the following steps:
  1. Partition data items into k non-empty subsets.
  2. Obtain the centroids as the centers (mean points) of the partitions.
  3. Obtain new partitions: assign each data item to the cluster of the nearest centroid.
  4. Stop when no more new assignment is found; otherwise go back to Step 2.
- The algorithm may need to go through many iterations before it terminates or converges.

Initialization

- We can randomly pick a few samples from the dataset as initial centroids (Forgy).
- We can randomly set the membership of all the samples, and compute the centroids thereafter (Random partition).
- We can initialize centroids with (small) random values.
- Multiple runs are expected.

Things may go wrong!



k-means++

- David Arthur & Sergei Vassilvitskii, 2007: "k-means++: the advantages of careful seeding"
- A random initialization algorithm that improves both the speed and quality of k-means

1. Take one centre $\mathbf{c}_1$, chosen uniformly at random from $\mathcal{X}$: $\mathcal{C} = \{\mathbf{c}_1\}$.
2. $\forall \mathbf{x} \in \mathcal{X}$, calculate $D(\mathbf{x}) = \min_{\mathbf{c} \in \mathcal{C}} \|\mathbf{x} - \mathbf{c}\|$: the shortest distance from a data point $\mathbf{x}$ to the closest centre.
3. Take a new center, choosing $\mathbf{x} \in \mathcal{X}$ with probability $\frac{D(\mathbf{x})^2}{\sum_{\mathbf{x} \in \mathcal{X}} D(\mathbf{x})^2}$: $\mathcal{C} \leftarrow \mathcal{C} \cup \mathbf{x}$
4. Repeat Steps 2 & 3 until k cluster centres have been taken.
5. Return: the initialized cluster centres $\mathcal{C}$

k-means: Pros and Cons

- Strength
  - Simple to implement.
  - Quite efficient. ...
- Weakness
  - *Often* terminates at a local optimum.
  - Applicable only when 'means' are defined. What about categorical data?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes

A Little Variation

- What if we use the city-block distance for cluster centroids to search for members?
- A different optimization objective using L_1:

  $\mathcal{F} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{m}_i - \mathbf{x}\|_1$
- To optimize $\mathcal{F}$, we have $\frac{\delta \mathcal{F}}{\delta \mathbf{m}_k} = 0$:
  - $\sum_{\mathbf{x} \in C_k} \text{sign}(\mathbf{x} - \mathbf{m}_k) = 0$
  - i.e., $\mathbf{m}_k = \text{median}\{\mathbf{x} \in C_k\}$.
- The median/medoid of a group of points is straightforward to calculate and less prone to be affected by outliers.

Choosing k

- Largely heuristic
- Could depend on the application
- Visual assessment: plot data (e.g. using principal component analysis) and check for groups
- Incremental evaluation: Add one at a time until reaching an "elbow" (reconstruction error/log likelihood/intergroup distances)
- The Linde-Buzo-Gray algorithm (LBG): progressively splits existing clusters into two

- ER: Hamerly & Elkan, "Learning the k in k-means", 2003

Fuzzy c-means

- Uses soft membership: $0 \leq \mu_{ij} \leq 1$
- Modified optimization criterion:
  $\mathcal{F} = \sum_{i=1}^{k} \sum_{j=1}^{N} \mu_{ij} \|\mathbf{x}_j - \mathbf{c}_i\|^2$.
- Fuzzy means: $\mathbf{m}_i = \dfrac{\sum_{j=1}^{N} \mathbf{x}_j \mu_{ij}}{\sum_{j=1}^{N} \mu_{ij}}$
- Soft membership update: $\mu_{ij} = \frac{\|\mathbf{x}_j - \mathbf{c}_i\|^{-2/(r-1)}}{\sum_{i=1}^{k} \|\mathbf{x}_j - \mathbf{c}_i\|^{-2/(r-1)}}, r \geq 1$

Model-based algorithms

- Assume data are generated from k probability distributions.
  - Typically, Gaussian
- Soft or probabilistic version of k-means clustering
- Need to learn the distribution parameters.
- Example: the EM Algorithm

The E-M algorithm

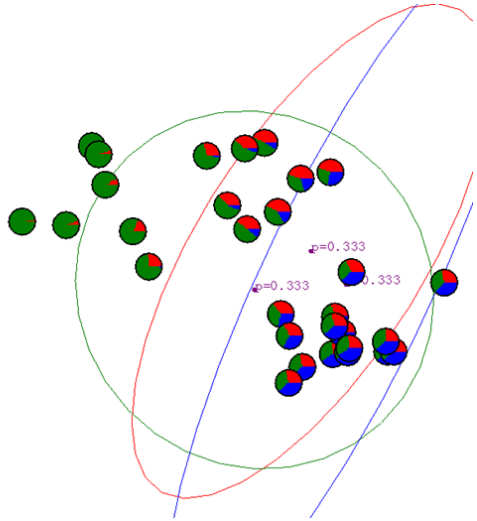
- EM stands for Expectation-Maximization.
- Each cluster is modelled by a Gaussian probability density function $p(\mathbf{x}|c_i)$.
- Steps:
  1. Initialize each cluster model randomly (e.g. set 'prior' $P(c_i)$ and 'conditional' $p(\mathbf{x}|c_i)$ randomly).
  2. Expectation, assign points to clusters: compute 'posterior' $P(c_i|\mathbf{x})$ for each $\mathbf{x}$ and c_i using Bayes' rule.
  3. Maximization, re-estimate model parameters: re-compute $P(c_i)$ and $p(\mathbf{x}|c_i)$ based on $P(c_i|\mathbf{x})$.
  4. Repeat steps 2 and 3 until convergence.

EM in action: Initialization

Example by A. Moore

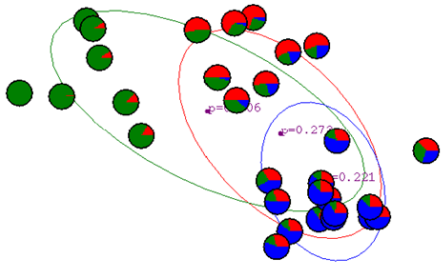
Indicators:

- Ellipses: Gaussian models
- Pies: data points with respective $P(c_i|\mathbf{x})$
- p: priors

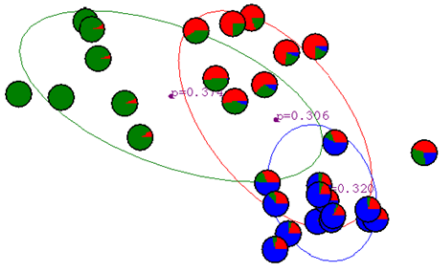


Source: Andrew Moore's tutorial

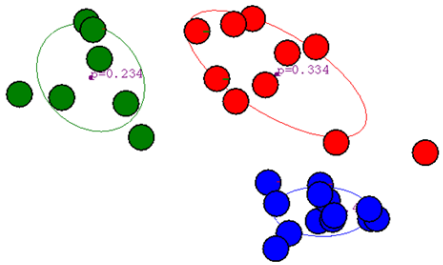
EM in action: Iteration 1



EM in action: Iteration 2



EM in action: Iteration 20



Density-based algorithms

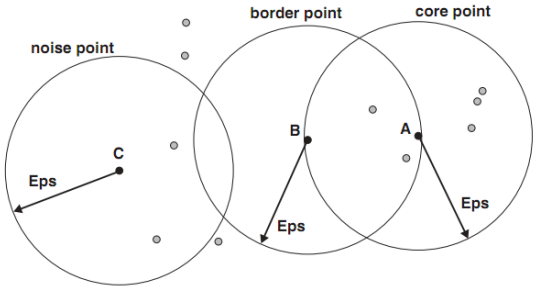
- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Several interesting algorithms: DBSCAN, OPTICS, DENCLUE, CLIQUE etc.

k-Medoids algorithms

- Find representative objects, called medoids, in clusters
- PAM (Partitioning Around Medoids, 1987) starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - PAM works effectively for small data sets, but does not scale well for large data sets
- CLARA (Kaufmann & Rousseeuw, 1990)
- CLARANS (Ng & Han, 1994): Randomized sampling focusing + spatial data structure (Ester et al., 1995)

DBSCAN: the idea

- Centre-based density (Ester, et al., KDD'96)
- Classify data points into
  - Core points: number of points around the point within a distance threshold *Eps* exceeds a threshold *MinPts*.
  - Border points: non-core point falling within the neighbourhood of a core point.
  - Noise points: neither a core point nor a border point.
- Clusters represented by *density-reachable* core points.



DBSCAN: the algorithm

1. Classify all points as core (K), border (B), or noise points.
2. Eliminate noise points.
3. $\forall(K_i, K_j), i \neq j, d(K_i, K_j) \leq Eps$, put an edge in between.
4. Form clusters as groups of all connected core points.
5. Assign each border point to one of the clusters of its associated core points.

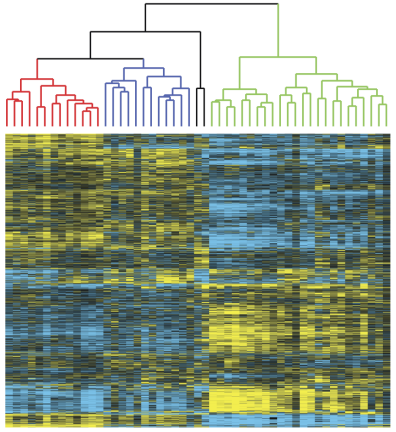
DBSCAN vs. CLARANS

CLARANS

DBSCAN

Hierarchical clustering

- Use distance matrix as clustering criteria
- Adopt either an agglomerative or divisive approach
- Decompose data objects into a several levels of nested partitioning, called a 'dendrogram'
- Clustering of the data objects is obtained by cutting the dendrogram at the desired level.



Example: clustering of gene data.

The need of clustering validation

- How good is the clustering?
- The clustering outcome, as a hypothesis, is acceptable?
- From multiple runs, which one is the best?
- Related: what is the best k?
- While cost functions can be useful in comparing results, we still don't know how good a clustering outcome is.
- When groundtruth is known ("external validation"): a few metrics can be used: adjusted rand index, average mutual information etc.

Silhouette coefficient

- Useful when data labels are not available
- For data item x_i, suppose it belongs to cluster C, let's measure the compactness of C:

$$a_i = E_{x_j \in C}(d(x_i, x_j))$$

- And the separation from other clusters:

$$b_i = \min_{x_j \notin C}(d(x_i, x_j))$$

- Silhouette coefficient is defined as

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

- Average silhouette coefficient reports the quality of clustering.

S-Dbw

- Halkidi & Vazirgiannis, ICDM'01
- Scatter: average cluster deviation over overall deviation

$$\text{Scat}(c) = \frac{1}{c}\frac{\sum_i \sigma(c_i)}{\sigma(S)}$$

- Inter-cluster density

$$\text{Dens\_bw}(c) = \frac{1}{c(c-1)} \sum_{i}^{c} \sum_{j \neq i}^{c} \frac{\text{density}(\mu_{ij})}{\max(\text{density}(v_i), \text{density}(v_i))}$$

- Validity index $\text{S\_Dbw}(c) = \text{Scat}(c) + \text{Dens\_bw}(c)$

Clustering: recap

- Normally iterative
- Initialization/parameter setting can be tricky
- Approaches: partitioning, model-based, density-based, hierarchical ...
- Unfinished business
  - How do we decide k =?
  - What if computing has to be done in a *distributed* manner?
  - And, what about efficiency?
  - Still more: subspace clustering, biclustering
  - What about streaming data? (To be continued)

Readings

- Alpaydin Chapter 7 (2nd/3rd Edition)
- DHS, Chapter 10.
- Andrew Moore, Statistical Data Mining Tutorials, `http://www.autonlab.org/tutorials/index.html`, esp. on Gaussian mixture models, k-means and hierarchical clustering.
- Liu et al., "Understanding of internal clustering validation measures", ICDM'10