Machine learning - HT 2016
10. Clustering

Varun Kanade

University of Oxford

March 4, 2016

Announcements

- Practical Next Week - No submission
- Final Exam: Pick up on Monday
- Material covered next week is not required for exam
- Will shut down Piazza on Sunday

Outline

Today we will see some approaches to clustering

- Defining clustering objective
- k-Means for clustering
- Hierarchical Clustering
- Spectral Clustering

Clustering

Often data can be grouped together into subsets that are coherent. However, this grouping may be subjective. It is hard to define a general framework.

Two types of clustering algorithms

1. Feature-based - Points are represented as vectors in $\mathbb{R}^n$
2. (Dis)similarity-based - Only know pairwise (dis)similarities

Two types of clustering methods

1. Flat - Partition the data into k clusters
2. Hierarchical - Organise data as clusters, clusters of clusters, and so on

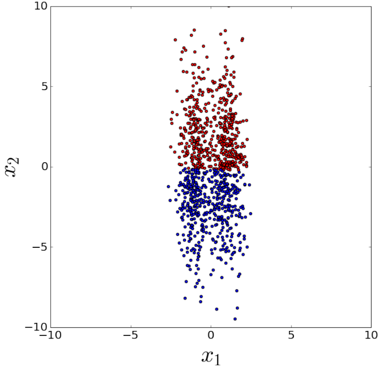
Defining Dissimilarity

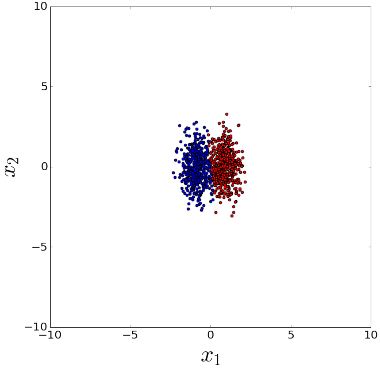
- Weighted distance between (real-valued) attributes

$$D(\mathbf{x}, \mathbf{x}') = f\left(\sum_{i=1}^{n} w_i d_i(x_i, x_i')\right)$$

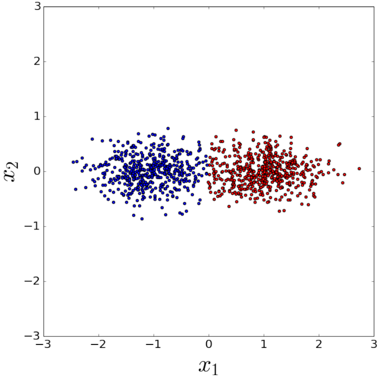
- In the simplest setting $w_i = 1$ and $d_i(x_i, x_i') = (x_i - x_i')^2$ and $f(z) = \sqrt{z}$
- Weights allow us to emphasise different variables differently
- If features are ordinal or categorical then define distance suitably
- Standardisation (mean 0, variance 1) may or may not help

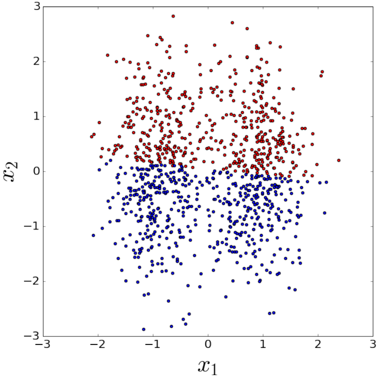
Helpful Standardisation





Unhelpful Standardisation





Partition Based Clustering : k-Means

Want to partition the data into subsets $C_1, \dots, C_k$, where k is fixed in advance

Define quality of a partition by

$$W(C) = \frac{1}{2} \sum_{j=1}^{k} \sum_{i,i' \in C_j} d(\mathbf{x}_i, \mathbf{x}_{i'})$$

If we use $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2$, then

$$W(C) = \sum_{j=1}^{k} \sum_{i \in C_j} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2$$

where $\boldsymbol{\mu}_j = \frac{1}{|C_j|} \sum_{i \in C_j} \mathbf{x}_i$

The objective is minimising the sum of squares of distances to the mean within each cluster

k-Means Objective

Minimise jointly over partitions $C_1, \ldots, C_k$ and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$

$$W(C) = \sum_{j=1}^{k} \sum_{i \in C_j} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2$$

This problem is NP-hard even for $k = 2$ for points in $\mathbb{R}^n$

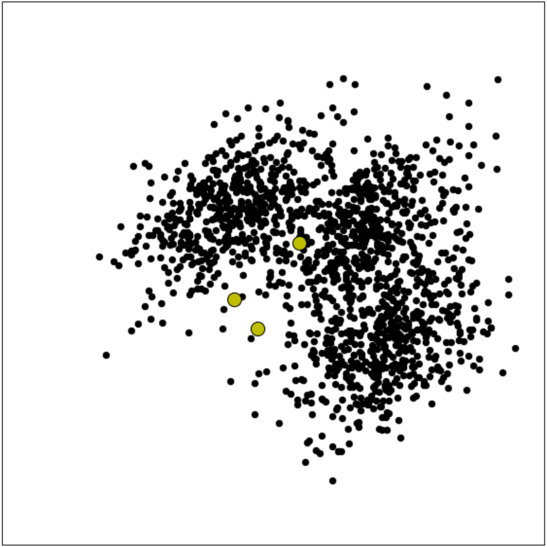
If we fix $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_j$, finding a partition $(C_j)_{j=1}^{k}$ that minimises W is easy

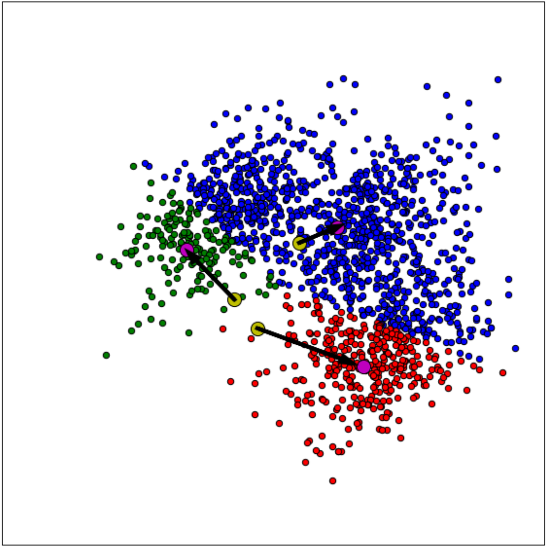
$$C_j = \{i \mid \|\mathbf{x}_i - \boldsymbol{\mu}_j\| = \min_{j'} \|\mathbf{x}_i - \boldsymbol{\mu}_{j'}\|\}$$

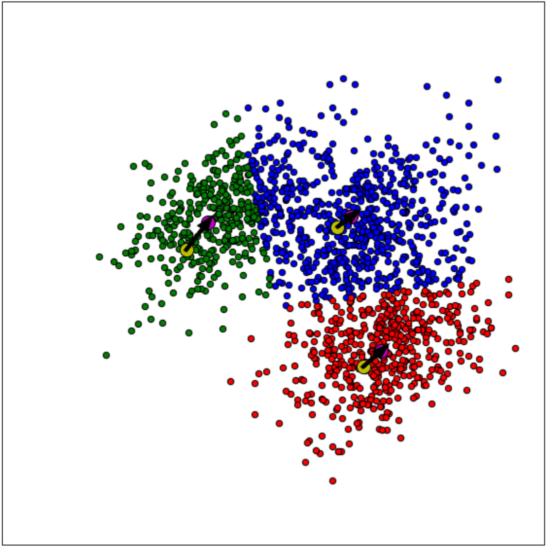
If we fix the clusters $C_1, \ldots, C_k$ minimising W with respect to $(\boldsymbol{\mu}_j)_{j=1}^{k}$ is easy

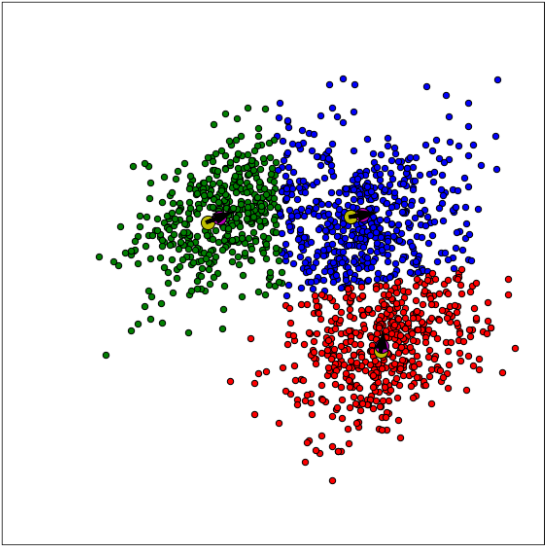
$$\boldsymbol{\mu}_j = \frac{1}{|C_j|} \sum_{i \in C_j} \mathbf{x}_i$$

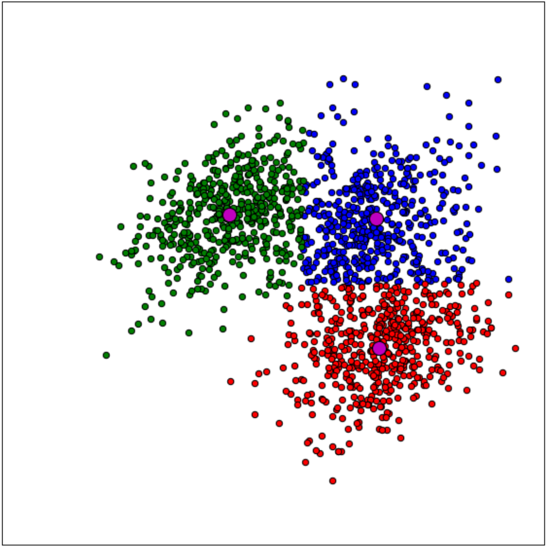
Iteratively run these two steps - assignment and update

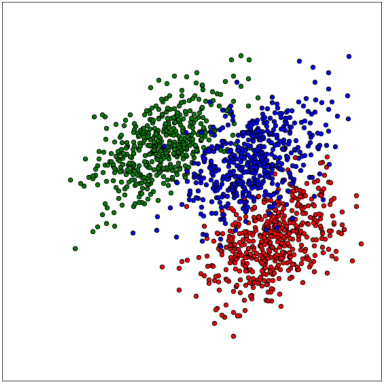


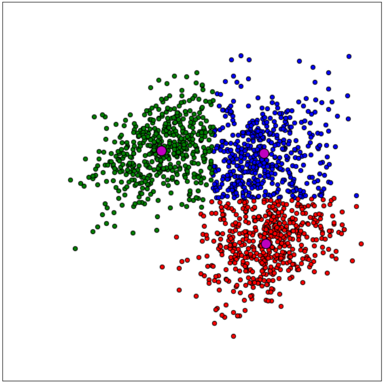












The k-Means Algorithm

Does the algorithm always converge?

Yes, because the W function decreases every time a new partition is used; there are only finitely many partitions

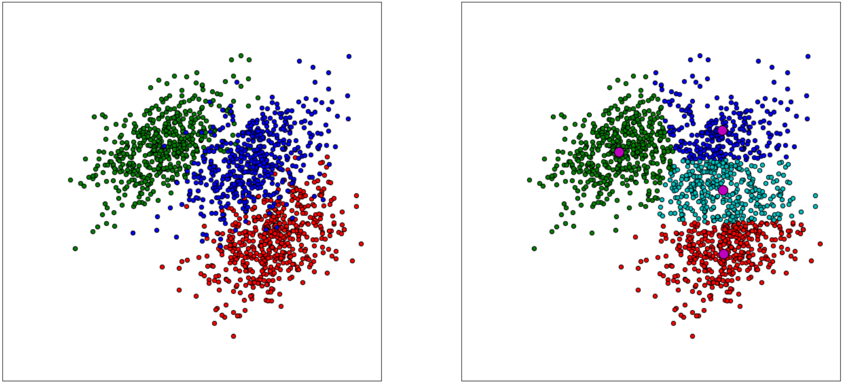
Convergence may be very slow in the worst-case, but typically fast on real-world instances

Convergence is probably to a local minimum

Run multiple times with random initialisation

Can use other criteria: k-medoids, k-centres, etc.

Selecting the right k is not easy: plot W against k and identify a "kink"



Clustering

US election 2016: Mitt Romney warns Trump not fit to run country

Paper that says human hand was ’designed by Creator’ sparks concern

Mystery of cosmic radio bursts grows even more intriguing

Lionel Messi combines with Neymar to score vs. Rayo

Hierarchical Clustering

Hierarchical structured data exists all around us

- Measurements of different species and individuals within species
- Top-level and low-level categories in news articles
- Country, county, town level data

Two General Strategies

- Agglomerative: Bottom-up, clusters formed by merging smaller clusters
- Divisive: Top-down, clusters formed by splitting larger clusters

Visualise this as a dendogram or tree

Measuring Dissimilarity at Cluster Level

To find hierarchical clusters we need to define dissimilarity at cluster level, not just at datapoints

Suppose we have dissimilarity at datapoint level, *e.g.*, $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|$

Few different proposals at cluster level, say C and C'

- Single Linkage

$$D(C, C') = \min_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$$

- Complete Linkage

$$D(C, C') = \max_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$$

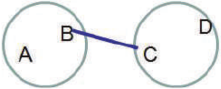
- Average Linkage

$$D(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$$

Measuring Dissimilarity at Cluster Level

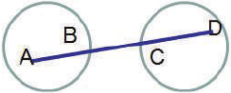
- Single Linkage

$$D(C,C') = \min_{\mathbf{x}\in C,\mathbf{x}'\in C'} d(\mathbf{x},\mathbf{x}')$$



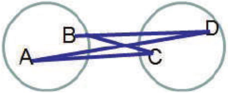
- Complete Linkage

$$D(C,C') = \max_{\mathbf{x}\in C,\mathbf{x}'\in C'} d(\mathbf{x},\mathbf{x}')$$



- Average Linkage

$$D(C,C') = \frac{1}{|C|\cdot|C'|}\sum_{\mathbf{x}\in C,\mathbf{x}'\in C'} d(\mathbf{x},\mathbf{x}')$$



Agglomerative Clustering Algorithm

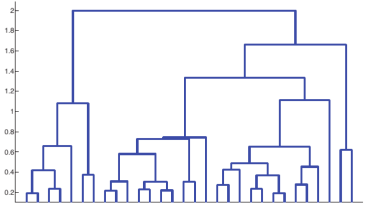
1. Initialise clusters as singletons $C_i = \{i\}$
2. Initialise clusters available for merging $S = \{1, \ldots, m\}$
3. Repeat
    a. Pick 2 most similar clusters, $(j, k) = \operatorname*{argmin}_{j,k \in S} D(j, k)$
    b. Let $C_l = C_j \cup C_k$
    c. If $C_l = \{1, \ldots, m\}$, break;
    d. Set $S = (S \setminus \{j, k\}) \cup \{l\}$
    e. Update $D(i, l)$ for all $i \in S$

Dendogram

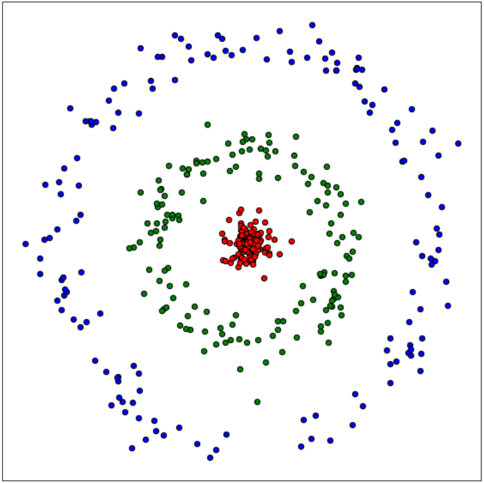
Binary tree, representing clusters as they were merged

The height of a node represents dissimilarity

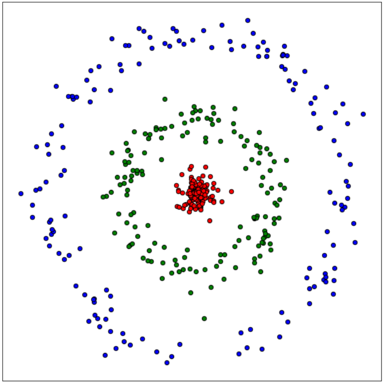
Cutting the dendogram at some level gives a partition of data

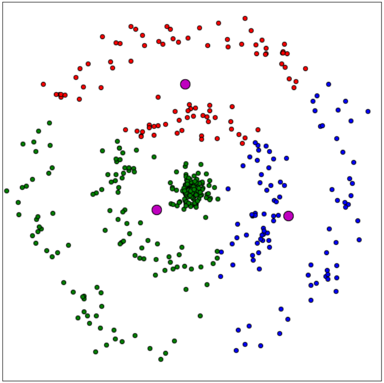


Spectral Clustering



Spectral Clustering





Limitations of k-means

k-means will typically form clusters that are spherical, elliptical, convex

Kernel PCA followed by k-means

Spectral clustering is a (related) alternative that often works better

Spectral Clustering

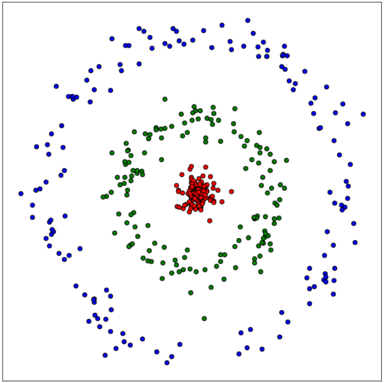
Construct a graph from data; one node for every point in dataset

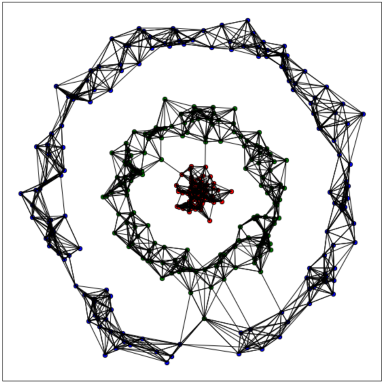
Use similarity measure, ***e.g.,*** $s_{i,j} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma)$

Construct mutual K-nearest neighbour graph, ***i.e.,*** (i, j) is an edge if either i is among the K nearest neighbours of j or vice versa

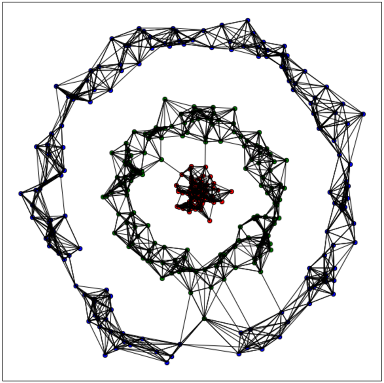
The weight of edge (i, j), if it exists is $s_{i,j}$

Spectral Clustering





Spectral Clustering



Use graph partitioning algorithms

Mincut can give bad cuts (only one node on one side of the cut)

Multi-way cuts, balanced cuts, are typically NP-hard to compute

Relaxations of these problems give eigenvectors of Laplacian

$\mathbf{W}$ is the weighted adjacency matrix

$\mathbf{D}$ is (diagonal) degree matrix: $D_{ii} = \sum_j W_{ij}$

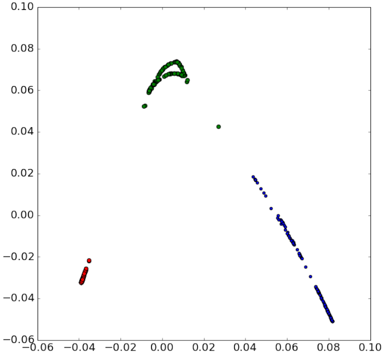
Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$

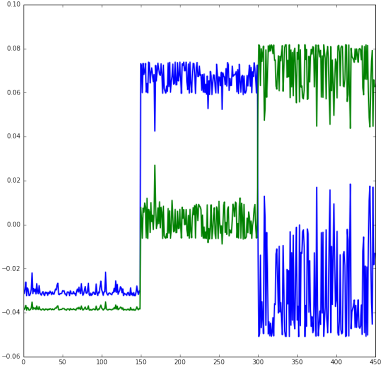
Normalised Laplacian: $\tilde{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}$

Spectral Clustering

Let us study the eigenvectors of the Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$

Spectral Clustering





Summary: Clustering

Clustering is grouping together similar data in a larger collection of heterogeneous data

Definition of good clusters often user-dependent

Clustering algorithms in feature space, *e.g.*, k-Means

Clustering algorithms that only use (dis)similarities: k-Medoids, hierarchical clustering

Spectral clustering when clusters may be non-convex

Next Time

- Reinforcement Learning