Lecture 9: Unsupervised Learning

COMP90049

Semester 1, 2022

Lea Frermann, CIS

Acknowledgement: Jeremy Nicholson, Tim Baldwin & Karin Verspoor

Roadmap

So far...

- Classification
- Evaluation of supervised machine learners
- Feature Selection

Today

- Unsupervised learning: Clustering
- Concepts
- Methods
- Evaluation

Clustering

A possible clustering of the weather dataset

Outlook	Temperature	Humidity	Windy	Cluster
sunny	**hot**	**high**	FALSE	?
sunny	**hot**	**high**	TRUE	?
overcast	**hot**	**high**	FALSE	?
rainy	mild	high	FALSE	?
rainy	**cool**	**normal**	FALSE	?
rainy	**cool**	**normal**	TRUE	?
overcast	**cool**	**normal**	TRUE	?
sunny	mild	**high**	FALSE	?
sunny	**cool**	**normal**	FALSE	?
rainy	**mild**	**normal**	FALSE	?
sunny	**mild**	**normal**	TRUE	?
overcast	**mild**	high	TRUE	?
overcast	**hot**	normal	FALSE	?
rainy	**mild**	high	TRUE	?

Clustering over the weather dataset (cf. outputs)

Outlook	Temperature	Humidity	Windy	Cluster	Play
sunny	hot	high	FALSE	0	no
sunny	hot	high	TRUE	0	no
overcast	hot	high	FALSE	0	yes
rainy	mild	high	FALSE	1	yes
rainy	cool	normal	FALSE	1	yes
rainy	cool	normal	TRUE	1	no
overcast	cool	normal	TRUE	1	yes
sunny	mild	high	FALSE	0	no
sunny	cool	normal	FALSE	1	yes
rainy	mild	normal	FALSE	1	yes
sunny	mild	normal	TRUE	1	yes
overcast	mild	high	TRUE	1	yes
overcast	hot	normal	FALSE	0	yes
rainy	mild	high	TRUE	1	no

Clustering

- Clustering is ***unsupervised***
- The class of an example is not known (or at least not used)
- Finding groups of items that are *similar*
- Success often measured subjectively
- Applications in pattern recognition, spatial data analysis, medical diagnosis, . . .

Clustering, basic contrasts

Exclusive vs. overlapping

- Can an item be in more than one cluster?

Deterministic vs. probabilistic (Hard vs. soft clustering)

- Can an item be partially or weakly in a cluster?

Hierarchical vs. partitioning

- Do the clusters have subset relationships between them? e.g. nested in a tree?

Partial vs. complete

- In some cases, we only want to cluster some of the data

Heterogenous vs. homogenous

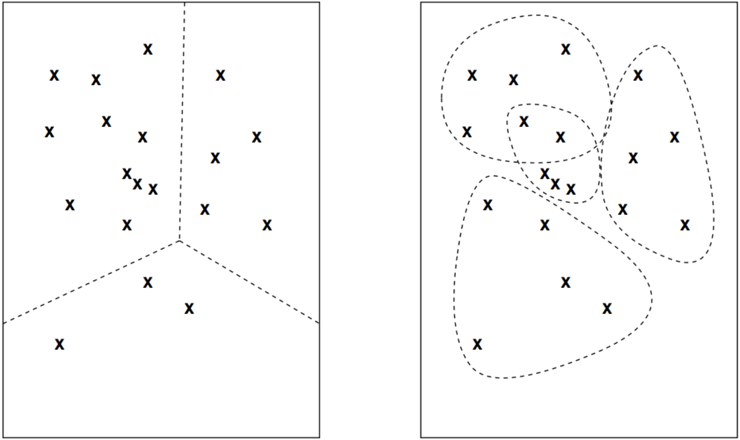
- Clusters of widely different sizes, shapes, and densities

Incremental vs. batch

- Is the whole set of items clustered in one go?

Exclusive vs. overlapping clustering

Can an item be in more than one cluster?



Deterministic vs. probabilistic clustering

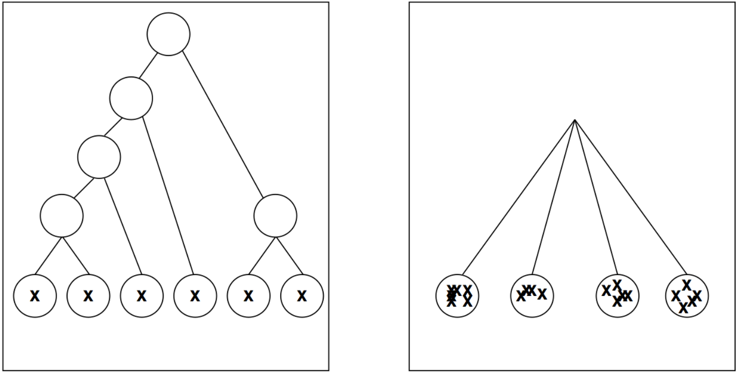
Can an item be partially or weakly in a cluster?

Instance	*Cluster*
1	2
2	3
3	2
4	1
5	2
6	2
7	4
⋮	⋮

	Cluster			
Instance	1	2	3	4
1	0.01	0.87	0.12	0.00
2	0.05	0.25	0.67	0.03
3	0.00	0.98	0.02	0.00
4	0.45	0.39	0.08	0.08
5	0.01	0.99	0.00	0.00
6	0.07	0.75	0.08	0.10
7	0.23	0.10	0.20	0.47
⋮	⋮			

Hierarchical vs. partitioning clustering

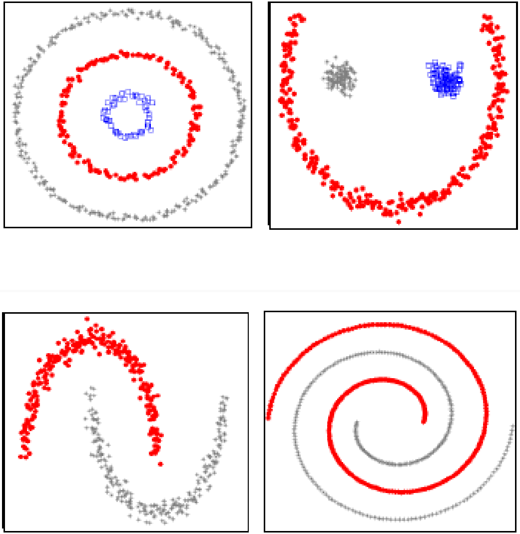
Do the clusters have subset relationships between them? e.g. nested in a tree?



Partial vs. complete

In some cases, we only want to cluster some of the data

Clusters of widely different sizes, shapes, and densities

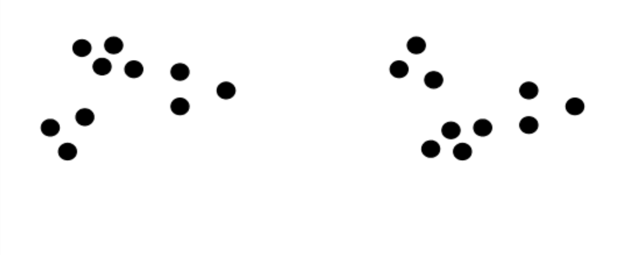


Incremental vs Batch clustering

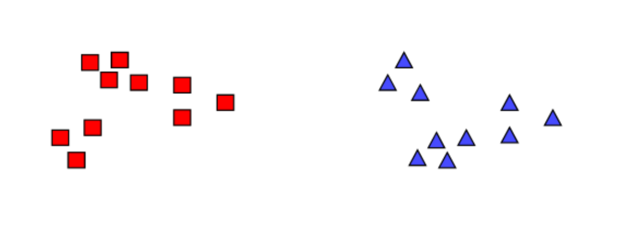
Clustering, Desiderata

- Scalability; high dimensionality
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Able to deal with noise and outliers
- Insensitive to order of input records

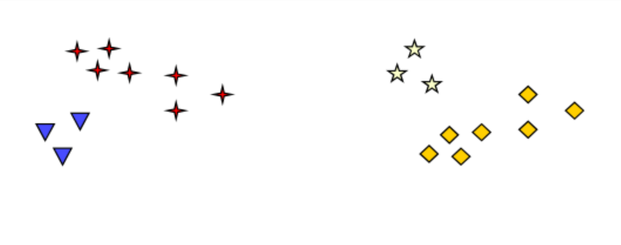
What is a good clustering?



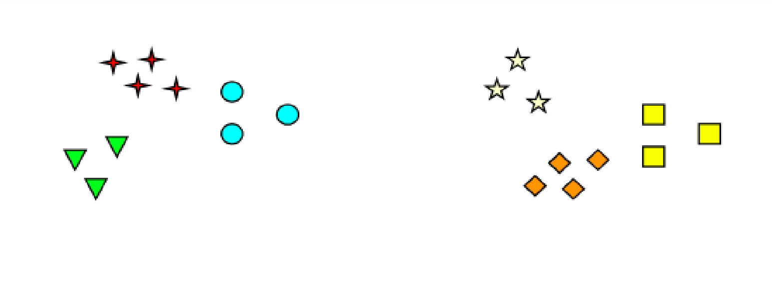
Two clusters?



Four clusters?



Six clusters?



Types of Evaluation

Unsupervised

- Measures the goodness of a clustering structure without respect to external information. Includes measures of **cluster cohesion** (compactness, tightness), and measures of **cluster separation** (isolation, distinctiveness).

Supervised

- Measures the extent to which the clustering structure discovered by a clustering algorithm matches some external structure. For instance, *entropy* can measure how well cluster labels match externally supplied class labels.

Unsupervised Evaluation I

A "good" cluster should have one or both of:

- **High cluster cohesion**: instances in a given cluster should be closely related to each other

$$cohesion(C_i) = \frac{1}{\sum_{x,y \in C_i} Distance(x, y)}$$

- **High Cluster Separation** instances in different clusters should be distinct from each other

$$separation(C_i, C_j) = \sum_{x \in C_i, y \in C_{j \neq i}} Distance(x, y)$$

Unsupervised Evaluation II

Most common measure for evaluating cluster quality is **Sum of Squared Error (SSE)** or *Scatter*

$$\sum_{i=1}^{k} \sum_{x \in C_i} dist^2(m_i, x)$$

- x: is a data point in cluster C_i
- m_i: is the representative point for cluster C_i
- For each point, the error is the distance to the nearest cluster
- To get SSE, we square these errors and sum them.

Unsupervised Evaluation II

Most common measure for evaluating cluster quality is **Sum of Squared Error (SSE)** or *Scatter*

$$\sum_{i=1}^{k} \sum_{x \in C_i} dist^2(m_i, x)$$

- Can show that the m_i that minimises SSE corresponds to the center (mean) of the cluster
- At 'application time': Given two clusters, we can choose the one with the smallest error
- One easy way to reduce SSE is to increase k, the number of clusters
- However, a good clustering with smaller k can have a lower SSE than a poor clustering with higher k

THE UNIVERSITY OF MELBOURNE

Sum of squared errors: Example

SSE for Categorical Attributes

Cluster 1 centroid:				
sunny	mild	high	no	
sunny	hot	high	no	$1^2 = 1$
sunny	hot	high	yes	$2^2 = 4$
overcast	hot	high	no	$2^2 = 4$
rainy	mild	high	no	$1^2 = 1$
sunny	mild	high	no	0
overcast	mild	high	yes	$2^2 = 4$
rainy	mild	high	yes	$2^2 = 4$

$SSE_1 = 18$

Cluster 2 centroid:			
overcast	cool	normal	yes
rainy	cool	normal	yes
overcast	cool	normal	yes
sunny	cool	normal	no
overcast	mild	normal	no
sunny	mild	normal	yes
overcast	hot	normal	no
rainy	cool	normal	no

$SSE_2 = 20$

$$SSE = SSE_1 + SSE_2 = 38$$

Supervised Evaluation

If labels are available, evaluate the degree to which class labels are consistent within a cluster and different across clusters

- **Purity:** Probability of the class with highers probability (representation) in each cluster. Higher is better.

$$purity = \sum_{i=1}^{k} \frac{|C_i|}{N} \max_j P_i(j)$$

- **Entropy:** Entropy probability of distribution over labels per cluster. Lower is better.

$$entropy = \sum_{i=1}^{k} \frac{|C_i|}{N} H(x_i)$$

- where x_i is the distribution of class labels in cluster i

Supervised Evaluation Example I

Ex. 1: Calculate the entropy and purity of the following cluster output

Cluster	Play = yes	Play = no
1	4	0
2	4	4

$$purity = \sum_{i=1}^{k} \frac{|C_i|}{N} \max_j P_i(j) \qquad entropy = \sum_{i=1}^{k} \frac{|C_i|}{N} H(x_i)$$

Supervised Evaluation Example II

Ex. 2: Calculate the entropy and purity of the following cluster output

Cluster	Play = yes	Play = no
1	2	0
2	6	4

$$\begin{aligned}
entropy_1 &= -1 \times log(1) - 0 \times log(0) = 0 \\
entropy_2 &= -0.6 \times log(0.6) - 0.4 \times log(0.4) = 0.97 \\
purity_1 &= max(1, 0) = 1 \\
purity_2 &= max(0.6, 0.4) = 0.6
\end{aligned}$$

Supervised Evaluation Example II

Ex. 2: Calculate the entropy and purity of the following cluster output

Cluster	Play = yes	Play = no	Entropy	Purity
1	2	0	0	1
2	6	4	0.97	0.6
Total:			**0.81**	**0.67**

$$entropy = \frac{2}{12} \times 0 + \frac{10}{12} \times 0.97 = 0.81$$

$$purity = \frac{2}{12} \times 1 + \frac{10}{12} \times 0.6 = 0.67$$

THE UNIVERSITY OF MELBOURNE

Methods

Similarity / Proximity / Closeness

Clustering finds groups of instances in the dataset which

- are similar or close to each other within a group
- are being different or separated from other clusters/clusters.

A key component of any clustering algorithm is a measurement of the **distance between any points**.

Measuring Similarity / Proximity / Closeness

Data points in Euclidean space

- Euclidean (L2) distance
- Manhattan (L1) distance

Discrete values

- Hamming distance (discrepancy between the bit strings)

d	a	b	c
a	0	1	1
b	1	0	1
c	1	1	0

For two bit strings, the number of positions at which the corresponding symbols are different

Text documents

- Cosine similarity
- Jaccard measure

Other measures

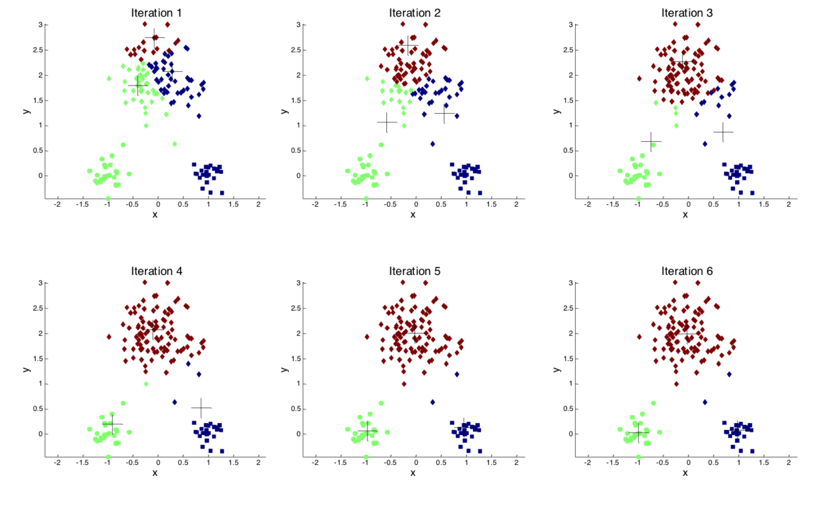
- Correlation
- Graph-based measures

Given k, the *k*-means algorithm is implemented in four steps:

1. Select *k* points to act as seed cluster centroids
2. **repeat**
3. Assign each instance to the cluster with the **nearest centroid**
4. Recompute the centroid of each cluster
5. **until** the centroids don't change

It is an exclusive, deterministic, partitioning, batch clustering method

Example, Iterations

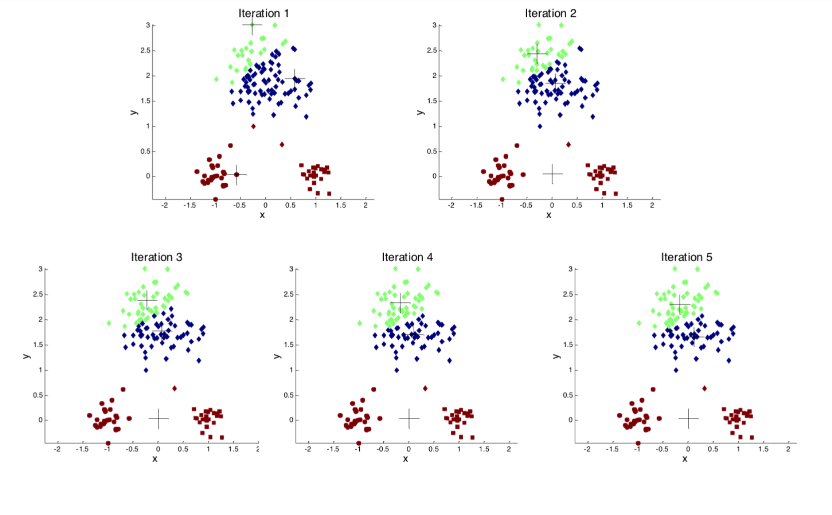


Demo: `https://www.naftaliharris.com/blog/visualizing-k-means-clustering/`

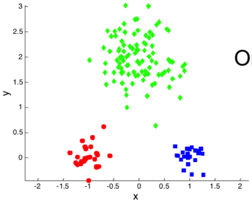
k-means Clustering – Details

- **Initial centroids** are often chosen randomly.
    - Clusters produced vary from one run to another.
- The **centroid** is (typically) the **mean** of the points in the cluster.
- ‘Nearest’ is based on proximity/**similarity metric**.
- K-means will **converge** for common similarity measures mentioned above.
    - Most of the convergence happens in the first few iterations.
    - Often the stopping condition is changed to *‘Until relatively few points change clusters’*
      (this way the stopping criterion will not depend on the type of similarity or dimensionality)

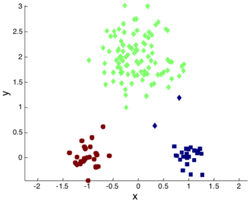
Example, Impact of initial seeds



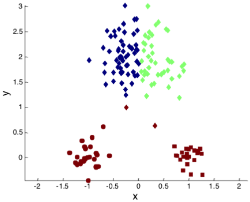
Example, Different outcomes



Original Points

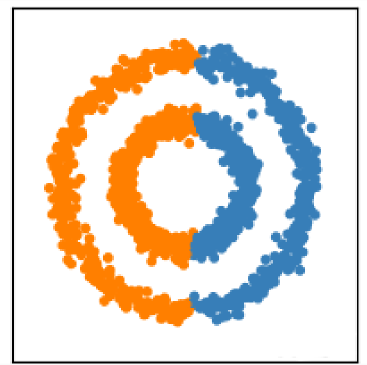


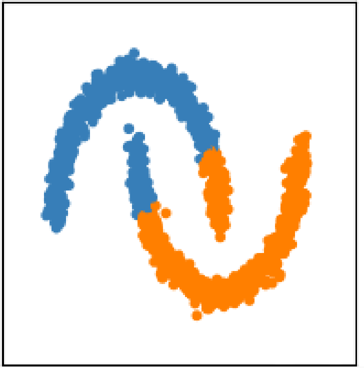
Optimal Clustering



Sub-optimal Clustering

Shortcomings of *k*-means





k-means, Pros and Cons

Strengths

- relatively **efficient**:
  - $O(ndki)$, where n is no. instances, d is no. attributes, k is no. clusters, and i is no. iterations; normally $k, i \ll n$
  - Unfortunately we cannot a priori know the value of i!
- can be extended to hierarchical clustering

Weaknesses

- tends to converge to **local minimum**; sensitive to seed instances
- need to **specify** k **in advance**
- not able to handle non-convex clusters, or clusters of differing densities or sizes
- “mean” ill-defined for nominal or categorical attributes
- may not work well when the data contains outliers

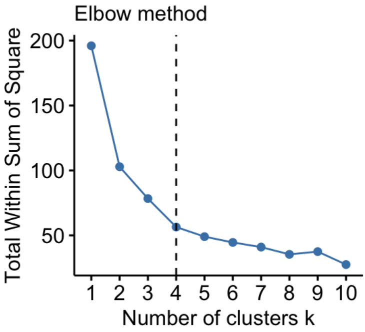
How to choose the number of clusters?

- Calculate *SSE* for different number of clusters K

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} (x - m_i)^2$$

- As K increases, we will have a smaller number of instances in each cluster $\rightarrow$ SSE decreases

- **Elbow method**: K increases to $K+1$, the drop of SSE starts to diminish



Hierarchical Clustering

Bottom-up (= agglomerative) clustering

- Start with single-instance clusters
- Iteratively **merge** clusters in a **bottom-up** fashion

Top-down (= divisive) clustering

- Start with one universal cluster
- Iteratively **split** clusters in a **top-down** fashion

In contrast to k-means clustering, hierarchical clustering only requires a measure of similarity between *groups* of data points. No seeds, no k value.

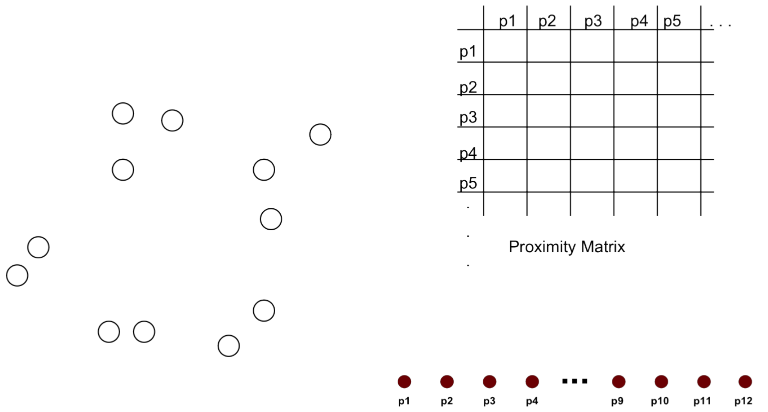
Agglomerative Clustering (bottom-up)

Input: A distance function

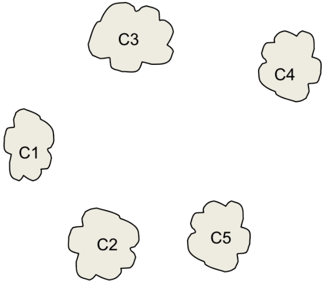
1. Compute the proximity matrix, if necessary.
2. **repeat**
3. Merge the closest two clusters
4. Update the proximity matrix to reflect the proximity between the new cluster and the original clusters
5. **until** Only one cluster remains

Output: A **dendrogram**: tree that represents the hierarchical clustering of all instances into successivly smaller groups.

Example, Step 1

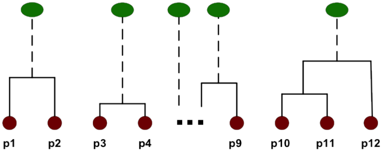


Example, Step 2

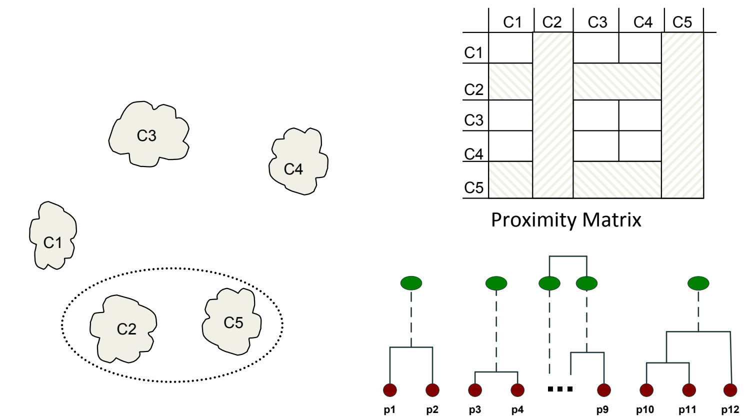


	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

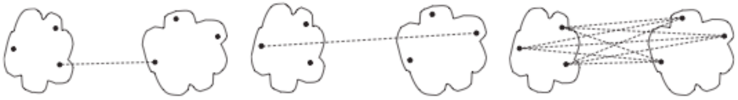
Proximity Matrix



Example, Step 3



Graph-based measure of Proximity



(a) MIN (single link.) (b) MAX (complete link.) (c) Group average.

Updating the proximity matrix:

- **Single Link**: *Minimum* distance between any two points in the two clusters. (most similar members)
- **Complete Link**: *Maximum* distance between any two points in the two clusters. (most dissimilar members)
- **Group Average**: *Average* distance between all points (pairwise).

Agglomerative Clustering Example

An initial **proximity matrix** with five clusters (aka. five points)

	1	2	3	4	5
1	1.00	0.90	0.10	0.65	0.20
2	0.90	1.00	0.70	0.60	0.50
3	0.10	0.70	1.00	0.40	0.30
4	0.65	0.60	0.40	1.00	0.80
5	0.20	0.50	0.30	0.80	1.00

What are the two closest points?

Agglomerative Clustering Example

An initial **proximity matrix** with five clusters (aka. five points)

	1	2	3	4	5
1	1.00	0.90	0.10	0.65	0.20
2	0.90	1.00	0.70	0.60	0.50
3	0.10	0.70	1.00	0.40	0.30
4	0.65	0.60	0.40	1.00	0.80
5	0.20	0.50	0.30	0.80	1.00

Merge points 1 & 2 into a new cluster: $\{1,2\}$

Agglomerative Clustering Example

An initial **proximity matrix** with five clusters (aka. five points)

	1	2	3	4	5
1	1.00	0.90	0.10	0.65	0.20
2	0.90	1.00	0.70	0.60	0.50
3	0.10	0.70	1.00	0.40	0.30
4	0.65	0.60	0.40	1.00	0.80
5	0.20	0.50	0.30	0.80	1.00

Merge points 1 & 2 into a new cluster: {1,2}

Update (single link):

	1	2	3	4	5	{1,2}
{1,2}						

Update (complete link):

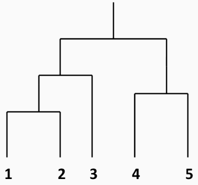
	1	2	3	4	5	{1,2}
{1,2}						

Agglomerative Clustering Example

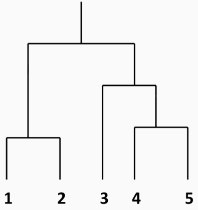
An initial **proximity matrix** with five clusters (aka. five points)

	1	2	3	4	5
1	1.00	0.90	0.10	0.65	0.20
2	0.90	1.00	0.70	0.60	0.50
3	0.10	0.70	1.00	0.40	0.30
4	0.65	0.60	0.40	1.00	0.80
5	0.20	0.50	0.30	0.80	1.00

The two resulting **dendrograms**:



Single link



Complete link

Quick coding demo!

`https://docs.scipy.org/doc/scipy-1.2.1/reference/cluster.hierarchy.html`

SciPy.org Sponsored By ENTHOUGHT

Scipy.org | Docs | SciPy v1.2.1 Reference Guide | Clustering package (`scipy.cluster`)

index | modules | next | previous

This is documentation for an old release of SciPy (version 1.2.1). Read this page in the documentation of the latest stable release (version 1.6.3).

Hierarchical clustering (scipy.cluster.hierarchy)

These functions cut hierarchical clusterings into flat clusterings or find the roots of the forest formed by a cut by providing the flat cluster ids of each observation.

fcluster(Z, t[, criterion, depth, R, monocrit])	Form flat clusters from the hierarchical clustering defined by the given linkage matrix.
fclusterdata(X, t[, criterion, metric, …])	Cluster observation data using a given metric.
leaders(Z, T)	Return the root nodes in a hierarchical clustering.

These are routines for agglomerative clustering.

linkage(y[, method, metric, optimal_ordering])	Perform hierarchical/agglomerative clustering.
single(y)	Perform single/min/nearest linkage on the condensed distance matrix `y`.
complete(y)	Perform complete/max/farthest point linkage on a condensed distance matrix.
average(y)	Perform average/UPGMA linkage on a condensed distance matrix.
weighted(y)	Perform weighted/WPGMA linkage on the condensed distance matrix.
centroid(y)	Perform centroid/UPGMC linkage.
median(y)	Perform median/WPGMC linkage.
ward(y)	Perform Ward's linkage on a condensed distance matrix.

These routines compute statistics on hierarchies.

cophenet(Z[, Y])	Calculate the cophenetic distances between each observation in the hierarchical clustering defined by the linkage `Z`.
from_mlab_linkage(Z)	Convert a linkage matrix generated by MATLAB(TM) to a new linkage matrix compatible with this module.
inconsistent(Z[, d])	Calculate inconsistency statistics on a linkage matrix.
maxinconsts(Z, R)	Return the maximum inconsistency coefficient for each non-singleton cluster and its children.
maxdists(Z)	Return the maximum distance between any non-singleton cluster.
maxRstat(Z, R, i)	Return the maximum statistic for each non-singleton cluster and its children.
to_mlab_linkage(Z)	Convert a linkage matrix to a MATLAB(TM) compatible one.

Routines for visualizing flat clusters.

dendrogram(Z[, p, truncate_mode, …])	Plot the hierarchical clustering as a dendrogram.

These are data structures and routines for representing hierarchies as tree objects.

ClusterNode(id[, left, right, dist, count])	A tree node class for representing a cluster.
leaves_list(Z)	Return a list of leaf node ids.
to_tree(Z[, rd])	Convert a linkage matrix into an easy-to-use tree object.
cut_tree(Z[, n_clusters, height])	Given a linkage matrix Z, return the cut tree.
optimal_leaf_ordering(Z, y[, metric])	Given a linkage matrix Z and distance, reorder the cut tree.

These are predicates for checking the validity of linkage and inconsistency matrices as well as for checking isomorphism of two flat cluster assignments.

Table Of Contents

Previous topic

scipy.cluster.vq.kmeans2

Next topic

scipy.cluster.hierarchy.fcluster

Quick search

search

Top-down clustering: Bi-secting k-means

Variant of K-means that can produce a partitional or a hierarchical clustering.

Basic Idea: to obtain K clusters, split the set of all points into two clusters, select one of these clusters to split, and so on, until K clusters have been produced.

1. Initialize the list of clusters to contain the cluster consisting of all points.
2. **repeat**
3. Pick a cluster from the current dendrogram.
4. {Perform several 'trial' bisections of the chosen cluster.}
5. **for** i = 1 to number of trials do
6. Bisect the selected cluster using basic k-means.
7. **end for**.
8. Select the two clusters from the bisection with the lowest total SSE.
9. Add these two clusters to the dendrogram.
10. **until** We have produced (a dendrogram of) k clusters.

Final Thoughts

Clustering is in the eyes of the beholder

- "The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art [...]"[1] Cluster validation an open research area!

Other variants of **unsupervised learning** (beyond the scope of the subject)

- **Density Estimation**: Estimate the underlying (unobservable) probability distribution that explains the observed variables (features). E.g., Gaussian mixture models (also: 'soft' K-means)
- **Dimensionality Reduction**: Map original features to a smaller set of features that still explain the observed data well (either a subset, or a new set of features). E.g., Principle Component Analysis

[1] `http://homepages.inf.ed.ac.uk/rbf/BOOKS/JAIN/Clustering_Jain_Dubes.pdf`

Roadmap

Today

- Unsupervised learning: Clustering
- Concepts
- Methods
- Evaluation

Next up...

- Semi-supervised learning

References

Resources:

Tan, Steinbach, Kumar (2006) Introduction to Data Mining. Chapter 8, Cluster Analysis `http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf`

Jain, Dubes (1988) Algorithms for Clustering Data. `http://homepages.inf.ed.ac.uk/rbf/BOOKS/JAIN/Clustering_Jain_Dubes.pdf`