Machine Learning

Clustering Algorithms

Clustering

What is Clustering?

Clustering is the grouping of objects into classes in such a way that:

- Objects in the same group are similar.
- Objects in different groups are dissimilar.

Tough question:

How do you measure similarity?

What is Similarity?

Similarity is hard to define…

What is Similarity? (2)

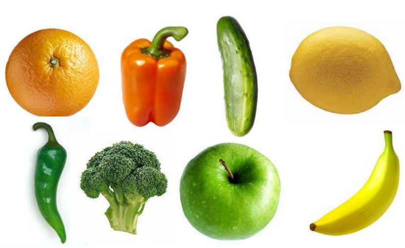
There is no single definition of similarity or dissimilarity between data objects.

The definition depends upon:

- The type of the data being considered
- What kind of similarity we are seeking

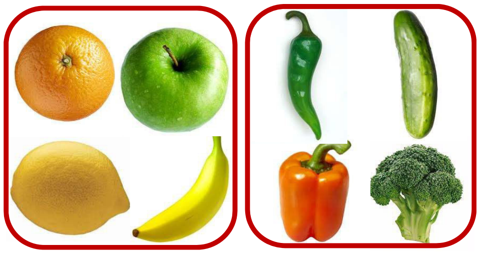
Clustering and Similarity

What are the “natural” groupings of these objects?



Clustering and Similarity (2)

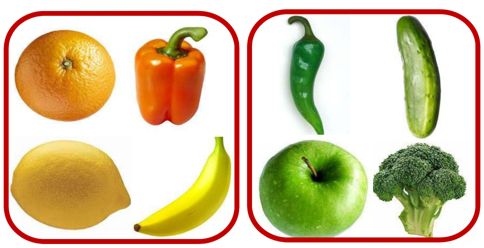
clustering by type



Matt Johnson, Ph.D. 7

Clustering and Similarity (3)

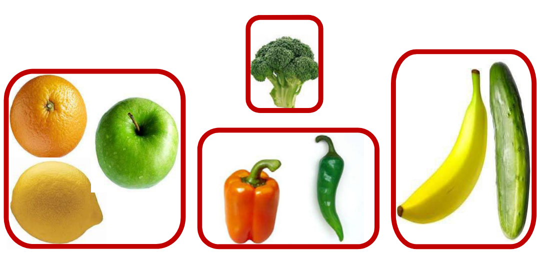
clustering by color



Matt Johnson, Ph.D. 8

Clustering and Similarity (4)

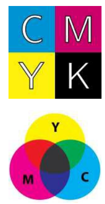
clustering by shape



Matt Johnson, Ph.D. 9

Example: Clustering By Color

Item	Cian	Magenta	Yellow	Black
Chili	72	0	51	57
Cucamber	11	0	45	19
Broccoli	15	0	23	31
Apple	25	0	74	20
Paprika	0	52	100	11
Lemon	0	20	93	5
Orange	0	18	65	3
Banana	0	1	100	1



Example: Clustering By Color (2)

Item	Cian	Magenta	Yellow	Black	Cluster
Chili	72	0	51	57	Cluster 1
Cucamber	11	0	45	19	Cluster 1
Broccoli	15	0	23	31	Cluster 1
Apple	25	0	74	20	Cluster 1
Paprika	0	52	100	11	Cluster 2
Lemon	0	20	93	5	Cluster 2
Orange	0	18	65	3	Cluster 2
Banana	0	1	100	1	Cluster 2

Applications of Clustering

Marketing: Categorizing customers based on behavior

Banking: ATM Fraud detection (outlier detection)

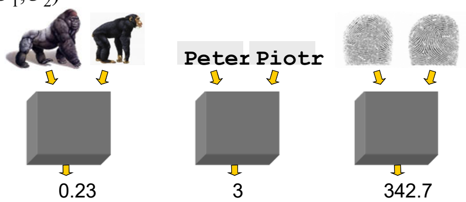
Image processing: Identifying objects on an image (such as face detection)

Insurance: Identifying groups of car insurance policy holders with a high average claim cost

Houses: Categorizing houses according to their house type, value, and geographical location

Distance Measure

Let O_1 and O_2 be two objects from the universe of all possible objects. The **distance** (dissimilarity) between O_1 and O_2 is a real number denoted by $D(O_1,O_2)$



Distance Measure (2)

What properties should a distance measure have?

- $D(A,B) = D(B,A)$ *Symmetry*
- $D(A,A) = 0$ *Constancy of Self-Similarity*
- $D(A,B) = 0$ iff A=B *Positivity (Separation)*
- $D(A,B) \leq D(A,C) + D(B,C)$ *Triangular Inequality*

Distance Measure Properties

$D(A,B) = D(B,A)$

Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex".

$D(A,A) = 0$

Otherwise you could claim "Alex looks more like Bob than Bob does".

Distance Measure Properties (2)

D(A,B) = 0 iff A=B

Otherwise there are objects in your world that are different, but you cannot tell them apart.

$D(A,B) \le D(A,C) + D(B,C)$

Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl".

Types of Clustering Algorithms

Hierarchical algorithms

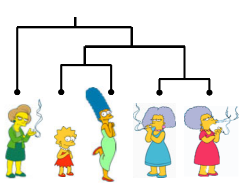
Create a hierarchical decomposition of the set of objects using some criterion

Partitional algorithms

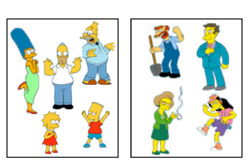
Construct various partitions and then evaluate them by some criterion

Types of Clustering Algorithms (2)

Hierarchical



Partitional

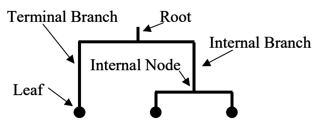


Hierarchical Clustering

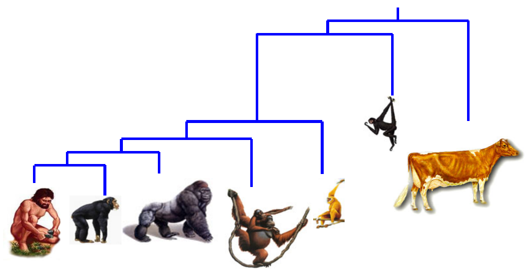
Dendograms

A **dendogram** is means of summarizing distance measurements.

The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.



Example Heirarchy

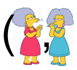


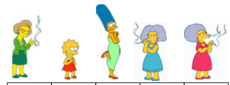
(Bovine:0.69395, (Spider Monkey 0.390, Gibbon:0.36079, (Orang:0.33636, (Gorilla:0.17147, (Chimp:0.19268, Human:0.11927):0.08386):0.06124):0.15057):0.54939)

Distance Matrix

We begin with a distance matrix which contains the distances between every pair of objects in our database.

D() = 8

D() = 1



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

Dendograms

The number of dendrograms with n leafs =

$$(2n-3)!/[(2^{(n-2)})(n-2)!]$$

Number of Leafs	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	…
10	34,459,425

Since we cannot tractably test all possible trees, we will need to use heuristic search…

Hierarchical Clustering

There are two types of hierarchical strategies:

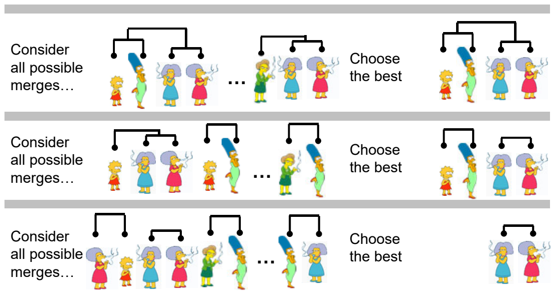
Bottom-Up (**Agglomerative**)

Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Top-Down (**Divisive**)

Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.

Example: Agglomerative Clustering



Linkage

We know how to measure the distance between two objects.

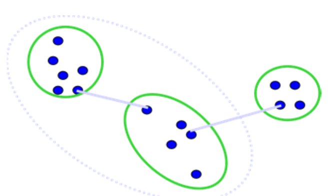
How do you define the distance between an object and a cluster, or define the distance between two clusters?

There are three basic methods for determining this:

- Single Linkage
- Complete Linkage
- Average Linkage

Single Linkage Method

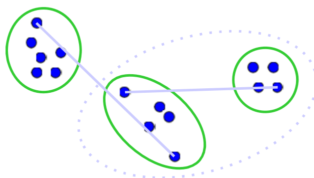
Using the **single linkage** or **nearest neighbors** method, the cluster distance is the distance between the two closest members in each cluster.



This method can create potentially long and skinny clusters.

Complete Linkage Method

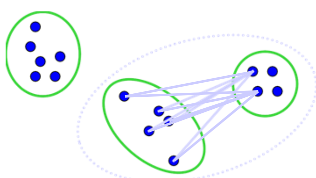
Using the **complete linkage** or **furthest neighbors** method, the cluster distance is the greatest distance between any two members in each cluster.



This method creates very tight clusters.

Average Linkage Method

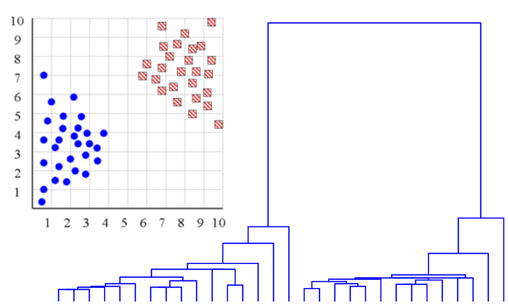
Using the **group average linkage** method, the cluster distance is the average distance between all pairings of objects from both clusters.



This is the most widely used method.

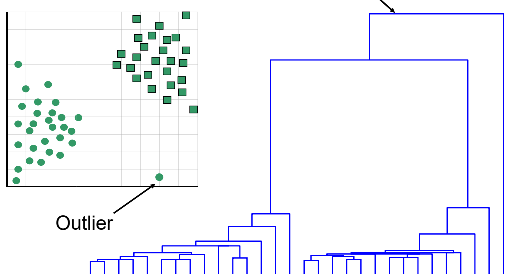
It is very robust against noise.

How Many Clusters?



Detecting Outliers

The single isolated branch is suggestive of a data point that is very different to all others.



What is an Outlier?

An **outlier** is a data point that is very far away from other data points.

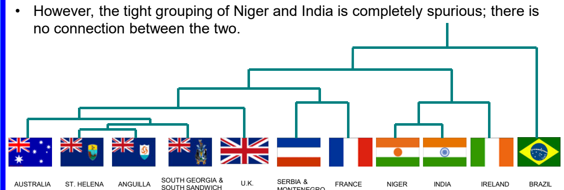
- Outliers could be errors in the data recording.
- Outliers could be some special data points with very different values.

Detecting and handling outlier data points is a significant challenge for all clustering algorithms.

Spurious Groupings

Hierarchal clustering can sometimes show patterns that are meaningless or spurious.

- The tight grouping of Australia, Anguilla, St. Helena, etc… is meaningful; all these countries are former UK colonies.
- However, the tight grouping of Niger and India is completely spurious; there is no connection between the two.



Hierarchical Methods Summary

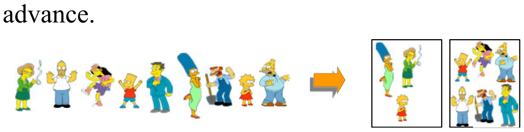
- No need to specify the number of clusters in advance
- Hierarchal nature maps nicely onto human intuition for some domains
- They do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Like any heuristic search algorithms, local optima are a problem
- Interpretation of results is very subjective

Partitional Clustering

Partitional Clustering

Partitional clustering is nonhierarchical, so each instance is placed in exactly one of *k* non-overlapping clusters.

Since the output is one set of clusters, the user must specify the desired number of clusters *k* in advance.



The K-Means Algorithm

K-means (MacQueen, 1967) is the most commonly used partitional clustering algorithm.

Given *k,* the *k-means* algorithm works as follows:

1. Choose *k* (random) data points to be the initial **centroids** or cluster centers
2. Assign each data point to the closest centroid
3. Re-compute the centroids using the current cluster memberships
4. If a convergence criterion is not met, repeat steps 2 and 3

What is Convergence?

One of the following convergence criterion is selected for the algorithm:

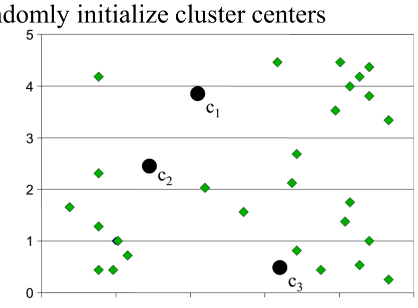
- no re-assignment of data points to different clusters
- no change of centroids
- A minimum decrease in the sum of squared error (SSE)

$$SSE = \sum_{j=1} \sum_{\mathbf{x} \in C_j} d(\mathbf{x}, \mathbf{m}_j)^2$$

- C_j is the jth cluster
- $\mathbf{m}_j$ is the centroid of cluster C_j
- $d(\mathbf{x}, \mathbf{m}_j)$ is the Euclidian distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$

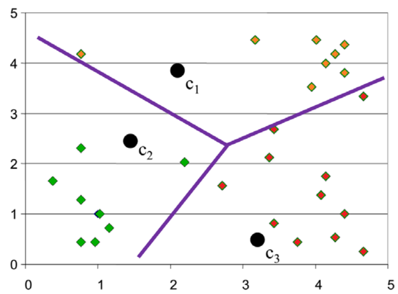
Step 1

Randomly initialize cluster centers



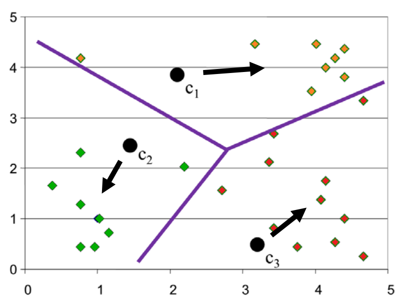
Step 2:

Determine cluster membership for each input



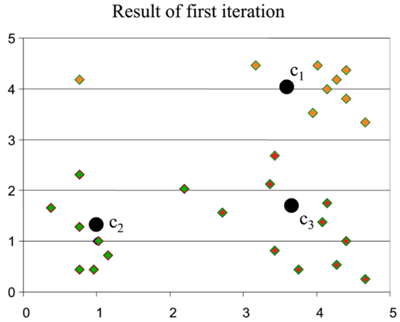
Step 3:

Re-calculate cluster centers:

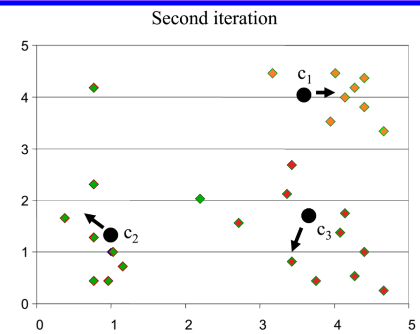


Step 4: Convergence?

Result of first iteration

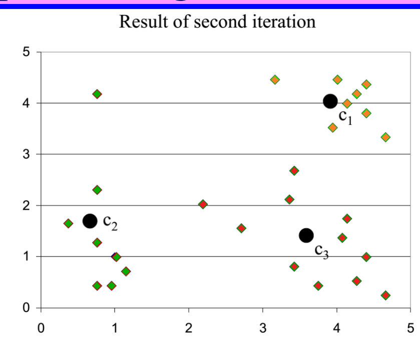


Steps 2-3:



Matt Johnson, Ph.D. 43

Step 4: Convergence?



Matt Johnson, Ph.D. 44

Strengths of K-Means

- Simple: easy to understand and to implement
- Efficient: Time complexity: $O(tkn)$, where
  - n is the number of data points
  - k is the number of clusters
  - t is the number of iterations
  - Since both k and t are small, k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.

Matt Johnson, Ph.D. 45

Weaknesses of K-Means

- The user needs to specify *k*.
- The algorithm is sensitive to outliers.
- It terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

Summary of K-Means

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity and efficiency.
- There is no clear evidence that any other clustering algorithm performs better in general.
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!