Introduction

Cluster analysis is a multivariate method which aims to classify a sample of subjects (or objects) on the basis of a set of measured variables into a number of different groups such that similar subjects are placed in the same group. An example where this might be used is in the field of psychiatry, where the characterization of patients on the basis of clusters of symptoms can be useful in the identification of an appropriate form of therapy. In marketing, it may be useful to identify distinct groups of potential customers so that, for example, advertising can be appropriately targeted.

WARNING ABOUT CLUSTER ANALYSIS

Cluster analysis has no mechanism for differentiating between relevant and irrelevant variables. Therefore the choice of variables included in a cluster analysis must be underpinned by conceptual considerations. This is very important because the clusters formed can be very dependent on the variables included.

Approaches to cluster analysis

There are a number of different methods that can be used to carry out a cluster analysis; these methods can be classified as follows:

- Hierarchical methods
  - Agglomerative methods, in which subjects start in their own separate cluster. The two 'closest' (most similar) clusters are then combined and this is done repeatedly until all subjects are in one cluster. At the end, the optimum number of clusters is then chosen out of all cluster solutions.
  - Divisive methods, in which all subjects start in the same cluster and the above strategy is applied in reverse until every subject is in a separate cluster. Agglomerative methods are used more often than divisive methods, so this handout will concentrate on the former rather than the latter.
- Non-hierarchical methods (often known as k-means clustering methods)

Types of data and measures of distance

The data used in cluster analysis can be interval, ordinal or categorical. However, having a mixture of different types of variable will make the analysis more complicated. This is because in cluster analysis you need to have some way of measuring the distance between observations and the type of measure used will depend on what type of data you have.

A number of different measures have been proposed to measure 'distance' for binary and categorical data. For details see the book by Everitt, Landau and Leese. Readers are also referred to this text for details of what to do if you have a mixture of different data types. For interval data the most common distance measure used is the Euclidean distance.

Hierarchical agglomerative methods

Within this approach to cluster analysis there are a number of different methods used to determine which clusters should be joined at each stage. The main methods are summarized below.

- Nearest neighbour method (single linkage method)

  In this method the distance between two clusters is defined to be the distance between the two closest members, or neighbours. This method is relatively simple but is often criticised because it doesn't take account of cluster structure and can result in a problem called chaining whereby clusters end up being long and straggly. However, it is better than the other methods when the natural clusters are not spherical or elliptical in shape.

- Furthest neighbour method (complete linkage method)

  In this case the distance between two clusters is defined to be the maximum distance between members — i.e. the distance between the two subjects that are furthest apart. This method tends to produce compact clusters of similar size but, as for the nearest neighbour method, does not take account of cluster structure. It is also quite sensitive to outliers.

- Average (between groups) linkage method (sometimes referred to as UPGMA)

  The distance between two clusters is calculated as the average distance between all pairs of subjects in the two clusters. This is considered to be a fairly robust method.

• Centroid method

Here the centroid (mean value for each variable) of each cluster is calculated and the distance between centroids is used. Clusters whose centroids are closest together are merged. This method is also fairly robust.

• Ward's method

In this method all possible pairs of clusters are combined and the sum of the squared distances within each cluster is calculated. This is then summed over all clusters. The combination that gives the lowest sum of squares is chosen. This method tends to produce clusters of approximately equal size, which is not always desirable. It is also quite sensitive to outliers. Despite this, it is one of the most popular methods, along with the average linkage method.

K-means Clustering

In these methods the desired number of clusters is specified in advance and the 'best' solution is chosen. The steps in such a method are as follows:

1. Choose initial cluster centres (essentially this is a set of observations that are far apart — each subject forms a cluster of one and its centre is the value of the variables for that subject).
2. Assign each subject to its 'nearest' cluster, defined in terms of the distance to the centroid.
3. Find the centroids of the clusters that have been formed
4. Re-calculate the distance from each subject to each centroid and move observations that are not in the cluster that they are closest to.
5. Continue until the centroids remain relatively stable.

Hierarchical Clustering

Given a set of N items to be clustered, and an NxN distance (or similarity) matrix, the basic process of Johnson's (1967) hierarchical clustering is this:

1. Start by assigning each item to its own cluster, so that if you have N items, you now have N clusters, each containing just one item. Let the distances (similarities) between the clusters equal the distances (similarities) between the items they contain.

2. Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one less cluster.
3. Compute distances (similarities) between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Step 3 can be done in different ways, which is what distinguishes *single-link* from *complete-link* and *average-link* clustering. In *single-link* clustering (also called the *connectedness* or *minimum* method), we consider the distance between one cluster and another cluster to be equal to the shortest distance from any member of one cluster to any member of the other cluster. If the data consist of similarities, we consider the similarity between one cluster and another cluster to be equal to the greatest similarity from any member of one cluster to any member of the other cluster. In *complete-link* clustering (also called the *diameter* or *maximum* method), we consider the distance between one cluster and another cluster to be equal to the longest distance from any member of one cluster to any member of the other cluster. In *average-link* clustering, we consider the distance between one cluster and another cluster to be equal to the average distance from any member of one cluster to any member of the other cluster. A variation on average-link clustering is the UCLUS method of D'Andrade (1978) which uses the median distance.

DBSCAN Clustering

DBSCAN, (Density-Based Spatial Clustering of Applications with Noise), captures the insight that clusters are dense groups of points. The idea is that if a particular point belongs to a cluster, it should be near to lots of other points in that cluster.

It works like this: First we choose two parameters, a positive number epsilon and a natural number minPoints. We then begin by picking an arbitrary point in our dataset. If there are more than minPoints points within a distance of epsilon from that point, (including the original point itself), we consider all of them to be part of a "cluster". We then expand that cluster by checking all of the new points and seeing if they too have more than minPoints points within a distance of epsilon, growing the cluster recursively if so.

Eventually, we run out of points to add to the cluster. We then pick a new arbitrary point and repeat the process. Now, it's entirely possible that a point we pick has fewer than minPoints

points in its epsilon ball, and is also not a part of any other cluster. If that is the case, it's considered a "noise point" not belonging to any cluster.

(There's a slight complication worth pointing out: say minPoints=4, and you have a point with three points in its epsilon ball, including itself. Say the other two points belong to two different clusters, and each has 4 points in their epsilon balls. Then both of these dense points will "fight over" the original point, and it's arbitrary which of the two clusters it ends up in. To see what I mean, try out "Example A" with minPoints=4, epsilon=1.98. Since DBSCAN considers the points in an arbitrary order, the middle point can end up in either the left or the right cluster on different runs. This kind of point is known as a "border point").

To illustrate the "epsilon ball rules", before the algorithm runs I superimpose a grid of epsilon balls over the dataset you choose, and color them in if they contain more than minPoints points. To get an additional feel for how this algorithm works, check out the "DBSCAN Rings" dataset, which consists of different numbers of points in different sized circles.

Note that in the actual DBSCAN algorithm, epsilon and minPoints remain the same throughout. But I thought it'd be fun to play around with changing them while the algorithm is running, so I've left the option in to do so.