

Lecture 8 - Discovering Groups

Summary

Being able to meaningfully cluster data into groups using **Clustering** or **Cluster Analysis** is a key part of the process of exploratory and descriptive data mining. Clustering techniques are all a form of unsupervised machine learning. Numerous techniques for clustering exist; in this lecture we'll look at two of the most common and useful: **Hierarchical Clustering** and **K-Means Clustering**. We'll also look briefly at a more advanced, but computationally intensive algorithm called **Mean Shift Clustering** that both produces clusters and finds the *modes* of the data.

Being able to group data into clusters is a good basis for understanding that data, however, in many cases these clusters can be difficult to interpret. An alternative approach is to attempt to produce 2D visualisations (images) that highlight the key relationships in the data by projecting the data from a high dimensionality to two dimensions. We'll look at three key algorithms: **Principal Component Analysis (PCA)**, **Multidimensional Scaling (MDS)** and **Self Organising Maps (SOMs)**

Key points

- Clustering is an unsupervised machine learning technique, that learns to group data without prior knowledge of what the groups should look like.

Hierarchical Clustering

- **Hierarchical Clustering** attempts to iteratively break data into a hierarchy of clusters
- **Hierarchical Agglomerative Clustering** builds a binary tree of clusters from the leaf nodes upwards towards the root
 - Known as a bottom-up approach
 - Requires three things:
 - a set of items to cluster
 - a distance measure to measure how close items are to each other
 - e.g. an L_p distance, a similarity measure converted to a distance (i.e. 1-Pearson or 1-cosine)
 - Doesn't necessarily have to be a distance computed over a vector; some forms of agglomerative clustering allow only need a matrix of distances or similarities computed between all items as input (see below)
 - a **linkage criterion** which measures dissimilarity of clusters as a function of the pairwise distances of items in the clusters
 - Basic approach:
 - Initially every item is in a cluster of its own
 - While there is more than one single cluster:
 - The closest pair of clusters according to the linkage criterion are merged into a bigger cluster

- By recording the merges at each step a binary tree structure linking the clusters can be formed
 - Often a useful way of utilising this is by drawing a diagram known as a dendrogram that shows the structure of the tree
- Two categories of linkage criterion:
 - Centroid-based linkage functions that measure similarity between clusters based on the distance between their centroids
 - Requires that each item is represented by a numeric feature vector that can be interpreted as a position in space
 - Examples:
 - **Weighted Centroid Clustering** (*WPGMC - Weighted Pair Group Method with Centroids; often also known as the “median” method*)
 - When two clusters s and t are combined into a new cluster u , the average of centroids s and t give the new centroid u
 - **Unweighted Centroid Clustering** (*UPGMC - Unweighted Pair Group Method with Centroids*)
 - When two clusters s and t are combined into a new cluster u , the average of the positions of all the items within s and t give the new centroid u
 - Distance-based linkage functions that measure distances between clusters as a function of the distances between items within those clusters.
 - Clustering can be performed purely as a function of a **distance matrix** in which each element $\mathbf{D}_{i,j}$ represents the distance, $d(i,j)$, between items i and j
 - Commonly used linkage criteria between two sets (clusters) of items A and B include:
 - **Minimum or single-linkage clustering**: $\min\{d(a,b) : a \in A, b \in B\}$
 - **Maximum or complete-linkage clustering**: $\max\{d(a,b) : a \in A, b \in B\}$
 - **Mean or average linkage clustering** (*UPGMA - Unweighted Pairwise Group Method with Arithmetic Mean*):

$$\frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$$
 - In general, complexity is $O(n^3)$, which can be a problem for large data sets, however there are some $O(n^2)$ variants for the single-linkage and complete-linkage cases
 - Divisive clustering algorithms (“top-down” approaches), which start with all the data in the root node and recursively split do exist
 - Not widely used in practice.
 - One major reason is that in general complexity is $O(2^n)$, which is worse than the agglomerative methods.

K-Means Clustering

- The **K-Means algorithm** (also known as *Lloyds algorithm*) is a simple, but powerful, approach to clustering that attempts to group data in a feature space into K groups or clusters represented by centroids (i.e. the mean point of the class in feature-space).
 - Algorithm:
 - The K-value must be chosen *a-priori* (beforehand)
 - To begin, K initial cluster centres are chosen (typically randomly or from a sample of the existing data points, although note that better initialisation procedures exist - e.g. the KMeans++ algorithm)

- Then the following process is performed iteratively until the centroids don't move between iterations (or the maximum number of iterations is reached):
 - Each point is assigned to its closest centroid
 - The centroid is recomputed as the mean of all the points assigned to it. If the centroid has no points assigned it is randomly re-initialised to a new point.
 - The final clusters are created by assigning all points to their nearest centroid.
- K-Means always converges, but not necessarily to the most optimal solution

Mean Shift Clustering

- **Mean Shift** is a standard algorithm to efficiently find the modes of a **Probability Density Function (PDF)** from a set of samples of that PDF (i.e. the featurevectors representing a set of items).
 - The only variable of the mean shift algorithm is the **kernel** and the **kernel bandwidth** of a **kernel density estimator**.
 - Clustering is an application of the mean shift procedure
 - Automatically chooses the number of clusters!
- The probability density function (PDF) of a continuous random variable is a function that describes the relative likelihood for this random variable to take on a given value
 - The PDF is non-negative everywhere and sums to 1
- In the context of a feature space, the PDF is a function that tells you how likely it is that a featurevector is *drawn* from a specific location in a feature space.
 - A feature vector drawn from part of the space where there are lots of similar items would have a higher probability density than if the drawn feature vector were from a part of the space with very few similar items
 - or in other words, dense parts of the space with more items have a higher probability density
 - Generally speaking, for arbitrary features describing a set of items, the PDF cannot be described empirically
 - Must be estimated using some other method
 - Simple, but crude, way to do this would be to quantise the feature space into bins in order to build a histogram
 - Each bin would contain the count of the number of items with feature vectors falling into that bin divided by the number of total items
 - Major disadvantage of this approach is that it isn't *continuous* and only gives a discrete approximation of the PDF
 - Better way to do this is to use a **Kernel Density Estimator** (also known as a **"Parzen Window"**)
 - Letting $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ represent the set of samples (e.g. feature vectors) in a d -dimensional space R^d from an unknown density f , then:
$$f(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^n K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)$$

where $K(\cdot)$ is the **kernel** (a non-negative function that integrates to one and has mean zero), and $h > 0$ is a smoothing parameter called the **kernel bandwidth**.

 - Common choice for the kernel is a multivariate Gaussian with zero mean and unit s.d.
 - For radially symmetric kernels, it suffices to define the profile of the kernel $k(\mathbf{x})$ satisfying $K(\mathbf{x}) = c_{k,d} k(\|\mathbf{x}\|^2)$
 - Intuitively one wants to choose h as small as the data will allow
 - there is always a trade-off between the bias of the estimator

and its variance however

- The **Mean Shift procedure** attempts to find the modes of the density function – that is the points where the gradient is 0: $\nabla f(\mathbf{x})=0$

- Assuming a radially symmetric kernel, then the gradient is:

$$\begin{aligned}\nabla f(\mathbf{x}) &= \frac{2c_{k,d}}{nh^{d+2}} \sum_{i=1}^n (\mathbf{x} - \mathbf{x}_i) g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right) \\ &= \frac{2c_{k,d}}{nh^{d+2}} \left[\sum_{i=1}^n g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right) \right] \left[\frac{\sum_{i=1}^n \mathbf{x}_i g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^n g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x} \right] \text{ where } \\ g(s) &= -k'(s).\end{aligned}$$

- The first term in the above is proportional to the density estimate at \mathbf{x} computed with a kernel $G(\mathbf{x}) = c_{g,d}^{-1} e^{-\frac{1}{2} \|\mathbf{x}\|^2}$, and the second term

$$\mathbf{m}_h(\mathbf{x}) \frac{\sum_{i=1}^n \mathbf{x}_i g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^n g\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}$$

is the **mean shift**.

- The mean shift vector always points toward the direction of the maximum increase in the density.

- The mean shift procedure, obtained by successive

- computation of the mean shift vector $\mathbf{m}_h(\mathbf{x}_t)$,
- translation of the window $\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{m}_h(\mathbf{x}_t)$

is guaranteed to converge to a point where the gradient of density function is zero

- Mean Shift Clustering** works as follows:

- for each feature vector:
 - apply the mean shift procedure until convergence and store the resultant mode
 - the set of featurevectors that converge to the same mode define the *basin of attraction* of that mode; all features that converged to the same mode belong to the same cluster

Visualising Data in Two Dimensions

- Sometimes rather than clustering data explicitly, we just want a way to visualise (through an image or diagram) which items are similar to each other and which are highly dissimilar
 - Lots of techniques allow us to do this

Principal Component Analysis

- Principal component analysis** allows us to project hight dimensional data into a lower dimensional space
 - Could use PCA to create a scatter plot where the x and y axis are the first and second principal components
 - Unfortunately this potentially has problems if the items in the data are not distributed such that the largest axes of variance are good discriminators of similarity

Multidimensional Scaling

- Multidimensional Scaling (MDS)** is another approach

Further Reading

- k-means++: the advantages of careful seeding (<http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>) . Arthur and Vassilvitskii. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics Philadelphia, PA, USA. pp. 1027–1035. 2007.
- Mean shift: A robust approach toward feature space analysis (<http://www.caip.rutgers.edu/riul/research/papers/pdf/mnshft.pdf>) . Comaniciu and Meer. IEEE Trans. Pattern Anal. Machine Intell., 24:603–619, 2002.