Learning never exhausts the mind

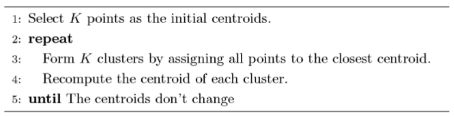
May 16, 2014

TSK - Chapter 8 - Cluster Analysis

Cluster analysis could be defined as "Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups."
In short, cluster analysis divides the data into groups(clusters) that are meaningful, useful or both.
There are many applications of cluster analysis to practical problems.

- **Clustering for Understanding**
    - Biology - creating taxonomy
    - Information Retrieval - grouping search results from the web
    - Climate - finding patterns in the atmosphere
    - Psychology and medicine - identify illness
    - Business - segment customers
- **Clustering for Utility**
    - Summarization - apply costly algorithms to cluster prototypes instead of the entire dataset.
    - Compression - prototypes are used when data objects are similar and loss is acceptable.
    - Efficiently finding nearest neighbors - use prototypes to reduce number of pairwise distance computations.
- **8.1 Overview**
    - **8.1.1 What is Cluster Analysis**
        - Defined above.
        - Groups data objects based only on information found in the data that describes the objects and their relationships.
        - Notion of a cluster is ambiguous
        - Classification is "Supervised Classification" and Cluster analysis is "Unsupervised Classification"
        - "Segmentation" and "Partitioning" are also sometimes synonymous with Clustering
    - What is not Cluster Analysis
        - Supervised Classification
        - Simple segmentation
        - Results of a query
        - Graph partitioning

- **8.1.2 Different types of Clusterings**
    - Hierarchical vs Partitional
        - Hierarchical (nested) - leaves are singletons
        - Partitional (unnested) - non overlapping subsets
    - Exclusive vs Overlapping vs Fuzzy
        - Exclusive - assign each object to a single cluster
        - Overlapping - objects can simultaneously belong to more than one cluster
        - Fuzzy - every object belongs to every cluster with a membership weight between 0(absolutely doesn't belong) and 1(abs belongs)
            - caveat : it doesn't address true multi-class situations since the weights need to add up to 1.
    - Complete vs Partial
        - Complete - assigns every object to a cluster
        - Partial - doesn't assign every object to a cluster
    - Heterogeneous vs Homogeneous
- **8.1.3 Different types of Clusters**
    - Well Separated - any point in a cluster is closer (more similar) to every other point in the cluster than to any point not in the cluster. Can have any shape.
    - Prototype based (center based) - an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster.
        - for data with continuous attributes, center is often the centroid. when centroid is not meaningful (data with categorical attributes), centre is often the medoid.
        - tends to be globular.
    - Graph based - group of objects that are connected to one another but have no connection to objects outside the group. (Connected Components)
        - Contiguity-based - 2 objects are connected only if they are within a specified distance of each other. Useful when clusters are irregular or intertwined.
        - Clique - another example of graph based cluster.
    - Density based - a cluster is a dense region of objects that is surrounded by a region of low density. Used when clusters are irregular or intertwined and when noise and outliers are present.
    - Shared property (conceptual clustering) - Finds clusters that share some common property or represent a particular concept.
    - Objective function - Finds clusters that minimize or maximize an objective function.
- Characteristics of Input data

    - Type of proximity/density measure
    - Sparseness
    - Attribute Type
    - Type of data
    - Dimensionality
    - Noise and Outliers
    - Type of Distribution
  - Clustering Algorithms
    - K-means - prototype based, partitional clustering technique - finds k clusters represented by their centroids.
    - Hierarchical Clustering - leaves are singletons, top down or bottom up.
    - DBSCAN - density based, partial that produces a partitional clustering. number of clusters is automatically determined by the algorithm.
- **8.2 K-means**
  - Partitional clustering technique in which each cluster is associated with a centroid and each point is assigned to a cluster with the closest centroid. Number of clusters K, must be given as input.
  - **8.2.1 The basic K-means algorithm**
    - 

```
1: Select $K$ points as the initial centroids.
2: repeat
3:     Form $K$ clusters by assigning all points to the closest centroid.
4:     Recompute the centroid of each cluster.
5: until The centroids don't change
```

      - Initial centroids are often chosen randomly.
      - The centroid is generally the mean of the points in the cluster
      - Closeness is measured by Euclidean distance, cosine similarity, correlation ..etc.
      - Most of the convergence happens the first few iterations
      - Complexity is O(I * K * m * n)
    - Assigning points to the closest centroid - Manhattan(L_1) distance can be used for Euclidean data, Jaccard measure for document data
    - Centroids and Objective function - e.g for objective function : minimize the squared distance of each point to its closest centroid.
    - Data in Euclidean space - Objective function used is SSE(Sum of Squared Error) also known as 'scatter'.

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

      - where *dist* is the standard Euclidean (L_2) distance b/w objects.

- Centroid that minimizes the SSE of a cluster is the mean

$$\mathbf{c}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in C_i} \mathbf{x}$$

-

- Given 2 different sets of clusters, we prefer one with lower SSE.

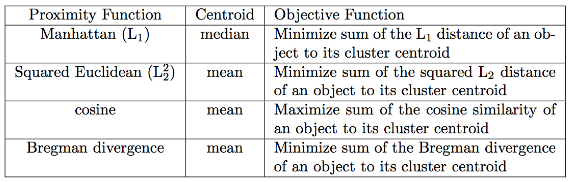
- Document data
  - represented as a DTM(document term matrix)
  - objective is to maximize similarity of the objects in the cluster to cluster centroid.
  - The analogous quantity to SSE is 'total cohesion'

$$\text{Total Cohesion} = \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} cosine(\mathbf{x}, \mathbf{c}_i)$$

  -

- The General case
  - There are many choices for the proximity function, centroid and objective function that can be used in basic k-means algorithm and that are guaranteed to converge.

Table 8.2. K-means: Common choices for proximity, centroids, and objective functions.



Proximity Function	Centroid	Objective Function
Manhattan (L_1)	median	Minimize sum of the L_1 distance of an object to its cluster centroid
Squared Euclidean (L_2^2)	mean	Minimize sum of the squared L_2 distance of an object to its cluster centroid
cosine	mean	Maximize sum of the cosine similarity of an object to its cluster centroid
Bregman divergence	mean	Minimize sum of the Bregman divergence of an object to its cluster centroid

  -

  - Bregman divergence includes squared Euclidean distance, the Mahalanobis distance and cosine similarity.

- Choosing initial centroids
  - Eg. 1 - Poor initial centroids : Randomly selected initial centroids may provide sub-optimal clustering with a higher SSE.
  - Eg. 2 - Limits of random initialization
  - If there are K 'real' clusters, chance of selecting one centroid from each is very small.

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K! n^K}{(Kn)^K} = \frac{K!}{K^K}$$

  -

- **Solutions to Initial centroids problem**
  - Perform multiple runs

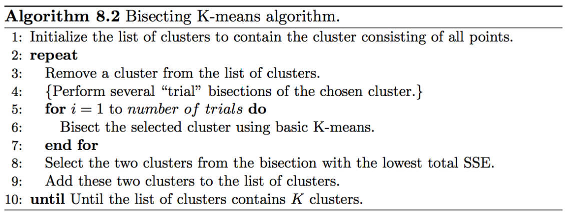
    - One way to minimize the effect or a poor random initial choice of centroids is to perform multiple runs.
    - But this also need not give optimal clustering. If a centroid falls between a pair of clusters, it will be stuck there by making one cluster out of 2.
  - Use hierarchical clustering on a sample
    - One effective approach to counter the limitations of random poor initial centroids is to take a sample of points and cluster them using hierarchical clustering technique and use those centroids as initial centroids.
    - But this is practical only if sample is relatively small and k is small compared to sample size.
  - Iterative approach
    - Select first point at random (or take centroid of all points)
    - for each successive initial centroid, select a point that is farthest from the previously selected centroids.
    - the downside is that this could select outliers.
  - Select more than K centroids
  - Post processing
  - Bisecting K-means
- Time and Space complexity
  - Storage required : O((m + K)n) , m = number of points, n = number of attributes
  - Time required : O(I * K * m * n) , I = number of iterations

- **8.2.2 K-means: Additional issues**
  - Handling empty clusters
    - Basic K-means algo can yield empty clusters.
    - Choose a replacement otherwise SSE will be large
    - Approach 1 : Choose a replacement point that is farthest from any current centroid
    - Approach 2 : Choose a replacement point from the centroid that has highest SSE. This would split the cluster and reduce overall SSE.
  - Outliers
    - If there are outliers, centroids may not be as representative as they ought to be and SSE would be higher as well.
    - One approach is to discover outliers and eliminate them beforehand. But there could be situations where outliers are important (compression, financial data)

- Another approach is to identify outliers during pre-processing. Identify points with high contributions to SSE.
- Also we could eliminate small clusters

- Reducing the SSE with post processing
    - One way to reduce SSE is to increase number of clusters (larger K). 2 strategies that decrease SSE by increasing K are -
        - Split a cluster - Choose a cluster with largest SSE or largest standard deviation for an attribute.
        - Introduce a new centroid - One approach is to choose a point farthest from any cluster center. Another is to choose a point at random.
    - 2 strategies that decrease K and try to minimize the SSE increase are -
        - Disperse a cluster - removing a centroid and assigning the points to other clusters. Disperse a cluster that doesn't contribute much to the total SSE.
        - Merge 2 clusters - Clusters with closest centroids are usually chosen. Strategy is same as that used in hierarchical clustering.
    - Commonly used approach is to alternate cluster splitting and merging phases.
- Updating centroids incrementally
    - In the basic K-means algorithm, centroids are updated after all points are assigned to a centroid. An alternative is to update the centroids after each assignment (incremental approach). Each assignment updates 0 or 2 centroids.
    - Pros
        - Guarantees that empty clusters are not produced.
        - Relative weights may be adjusted. Better accuracy and faster convergence are possible if weights are used.
    - Cons:
        - It is more expensive
        - Introduces an order dependency (clusters produced depends on the order in which points are processed) . Although this can be overcome by randomizing the order.

- **8.2.3 Bisecting K-means**
    - A straightforward extension of the basic K-means algorithm that is based on a simple 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.



Algorithm 8.2 Bisecting K-means algorithm.

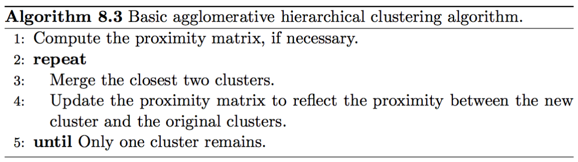
```
1: Initialize the list of clusters to contain the cluster consisting of all points.
2: repeat
3:   Remove a cluster from the list of clusters.
4:   {Perform several “trial” bisections of the chosen cluster.}
5:   for $i = 1$ to number of trials do
6:     Bisect the selected cluster using basic K-means.
7:   end for
8:   Select the two clusters from the bisection with the lowest total SSE.
9:   Add these two clusters to the list of clusters.
10: until Until the list of clusters contains $K$ clusters.
```

  - There are different ways to choose which cluster to split, say using size or SSE.
  - Usually the the clusters are refined by using the resulting centroids as initial centroids for basic k-means algorithm. This is useful since during bisecting k-means we’re using k-means algorithm ’locally’
  - Bisecting K-means is less susceptible to initialization problems.
  - Also we can use bisecting k-means for hierarchical clustering.

- **8.2.4 K-means and different types of clusters**
  - K-means has difficulty in detecting natural clusters when clusters have non-spherical shapes or widely different sizes or densities.
  - This could be overcome if the user is willing to increase K and have more clusters. So a cluster might have many sub clusters.
- **8.2.5 Strengths and weaknesses**
  - Strengths : Used for wide variety of data types, Very efficient
  - Weaknesses : Cannot handle non-globular clusters, not good with data that has outliers.
  - **8.2.6 K-means as an optimization problem**
  - Gradient descent - involves picking an initial solution and then repeating the fol- lowing two steps: compute the change to the solution that best optimizes the objective function and then update the solution.
  - Derivation of K-means as an algorithm to minimize the SSE (sum of squared errors): Proof that the best centroid for minimizing the SSE of a cluster is the mean of the points in the cluster.
  - Derivation of K-means for SAE (sum of absolute errors) : Ck is the median of the cluster.

- **8.3 Agglomerative Hierarchical Clustering**
  - 2 approaches of hierarchical clustering -
    - Agglomerative - Start with the points as individual clusters and, at each step, merge the closest pair of clusters.
    - Divisive - Start with one, all-inclusive cluster and, at each step, split a cluster until only singleton clusters of individual points remain.
  - Often displayed graphically using a tree-like diagram called dendogram, which displays sub-cluster relationships and also the order in which the clusters were merged.
  - Don’t have to assume any particular number of clusters.
  - May correspond to meaningful taxonomies.

- **8.3.1 Basic Agglomerative Hierarchical Clustering Algorithm**



Algorithm 8.3 Basic agglomerative hierarchical clustering algorithm.

```
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.
```

-
    - Key operation is the computation of proximity
- **Defining proximity b/w clusters**
    - Cluster proximity is typically defined with a particular type of cluster in mind. For example, many agglomerative hierarchical clustering techniques, such as MIN, MAX, and Group Average, come from a graph-based view of clusters.
- **Time and Space complexity**
    - Proximity matrix requires the storage of 1/m^2 proximities. The space needed to keep track of the clusters is proportional to the number of clusters. Hence, the total space complexity is O(m^2).
    - There are m steps and at each step the size, m^2, proximity matrix must be updated and searched. So time complexity is O(m^3). It can be reduced to O(m^2 log m) by keeping the data in a sorted list or heap
- **8.3.2 Specific Techniques of similarity measures**
    - MIN (single link)
        - Similarity of two clusters is based on the two most similar (closest) points in the different clusters. Determined by one pair of points, i.e., by one link in the proximity graph.
        - Strength : Can handle non-eliptical shapes.
        - Weakness : Sensitive to noise and outliers.
    - MAX (complete link)
        - Similarity of two clusters is based on the two least similar (most distant) points in the different clusters. Determined by all pairs of points in the two clusters.
        - Strength : Less susceptible to noise and outliers
        - Weakness : Tends to break large clusters, Biased towards globular clusters
    - Group Average
        - Proximity of two clusters is the average of pairwise proximity between points in the two clusters.
        - Compromise between Single and Complete Link
        - Strength : Less susceptible to noise and outliers
        - Weakness : Biased towards globular clusters

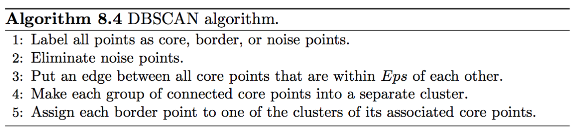
- Ward's method
  - Similarity of two clusters is based on the increase in squared error when two clusters are merged.
  - Similar to group average if distance between points is distance squared.
  - Hierarchical analogue of K-means - Can be used to initialize K-means.
  - Strength : Less susceptible to noise and outliers
  - Weakness : Biased towards globular clusters
- Distance between centroids : Similarity of two clusters is based on the distance between the centroids in the different clusters.

- **8.3.3 The Lance-Williams formula for Cluster Proximity**
  - A general formula for expressing hierarchical clustering proximity
  - Any of the cluster proximities that we have discussed above can be viewed as a choice of different parameters in the Lance-Williams formula
  - $$p(R,Q) = \alpha_A\, p(A,Q) + \alpha_B\, p(B,Q) + \beta\, p(A,B) + \gamma\, |p(A,Q) - p(B,Q)|$$
- **8.3.4 Key Issues in Hierarchical clustering**
  - Lack of a global objective function : It is greedy technique in which optimization is done locally at that specific moment.
  - Ability to handle different cluster sizes : In order to take care of the relative sizes of the clusters that are merged, we could take weights into account.
    - UPGMA - Unweighted Pair Group Method using Arithmetic averages.
    - WPGMA - Weighted Pair Group Method using Arithmetic averages.
  - Merging decisions are final: Once the decision is made, it cannot be undone at a later time.
- **8.3.5 Strengths and Weaknesses** : Already discussed above.

- **8.4 DBSCAN**
  - Density-based clustering locates regions of high density that are separated from one another by regions of low density. DBSCAN is a simple and effective density-based clustering algorithm.
  - **8.4.1 Traditional Density: Center-based approach**
    - In the center-based approach, density is estimated for a particular point in the data set by counting the number of points within a specified radius, Eps, of that point. This includes the point itself.
    - **Classification of Points According to Center-Based Density**
      - **Core point:** A point is a core point if it has more than a specified number of points (MinPts) within Eps.
      - **Border point:** A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

        - **Noise point:** A noise point is any point that is neither a core point nor a border point.

    - **8.4.2 The DBSCAN algorithm**



Algorithm 8.4 DBSCAN algorithm.

```
1: Label all points as core, border, or noise points.
2: Eliminate noise points.
3: Put an edge between all core points that are within Eps of each other.
4: Make each group of connected core points into a separate cluster.
5: Assign each border point to one of the clusters of its associated core points.
```

    -

    - Time and Space complexity
        - The basic time complexity of the DBSCAN algorithm is O(m × time to find points in the Eps-neighborhood), where m is the number of points. In the worst case, this complexity is O(m^2). If a different data structure is used, it could be as low as O(m logm).
        - The space requirement is O(m).
    - Selection of DBSCAN Parameters
        - The big question for DBSCAN is "how to determine the parameters Eps and MinPts ?"
        - The basic approach is to look at the behavior of the distance from a point to its kth nearest neighbor(k-dist).
        - Usually points within cluster will have small k-dist and noise points will have large k-dist. So if we compute the k-dist for all the data points for some k, sort them in increasing order, and then plot the sorted values, we expect to see a sharp change at the value of k-dist that corresponds to a suitable value of Eps.
    - Clusters of Varying Density
        - DBSCAN can have trouble with density if the density of clusters varies widely.
        - If the Eps threshold is low and if the densities vary, the denser clusters will be picked up and the dilute ones might be categorized as noise.
    - **8.4.3 Strengths and Weaknesses**
        - Strengths
            - Resistant to Noise
            - Can handle clusters of different shapes and sizes
        - Weaknesses
            - Doesn't work well for clusters with varying densities
            - Doesn't work well for High-dimensional data
- **8.5 Cluster Evaluation**
    - For supervised classification we have a variety of measures to evaluate how good our model is - Accuracy, precision, recall
    - For cluster analysis, the analogous question is "how to evaluate 'the goodness' of the resulting clusters?"

- The difficulty - Many times clustering is done as part of exploratory data analysis. So validation may not be needed. Also, different clustering algorithms might need different evaluation techniques - SSE might be suited for K-means but not for DBSCAN.
- **8.5.1 Overview**
  - **Aspects of Cluster Validation**
    1. Determining the clustering tendency of a set of data, (distinguishing whether non-random structure actually exists in the data).
    2. Determining the correct number of clusters.
    3. Evaluating how well the results of a cluster analysis fit the data without reference to external information.
    4. Comparing the results of a cluster analysis to externally known results, such as externally provided class labels.
    5. Comparing two sets of clusters to determine which is better.
  - Items 1, 2, and 3 do not make use of any external information - they are unsupervised techniques - while item 4 requires external information. Item 5 can be performed in either a supervised or an unsupervised manner.
  - The evaluation measures (indices), that are applied to judge various aspects of cluster validity are traditionally classified into the following three types -
    - **Unsupervised** (Internal indices) -
      - Measures the goodness of a clustering structure without referring to external information.
      - e.g SSE
      - Further divided into 2 classes - measures of cluster cohesion, cluster separation
    - **Supervised** (External indices)
      - Measures the extent to which the clustering structure discovered by a clustering algorithm matches some external structure.
      - e.g Entropy (measures how well cluster labels match externally supplied class labels)
    - **Relative**
      - Compares different clusterings or clusters
      - e.g Two K-means clusterings can be compared using either the SSE or entropy.
- **8.5.2 Unsupervised Cluster Evaluation Using Cohesion and Separation**

$$overall\ validity = \sum_{i=1}^{K} w_i\ validity(C_i).$$

  -

    - Cluster validity for a set of K clusters is a weighted sum of the validity of individual clusters.
    - The validity function can be cohesion, separation or some combination of these. The weights will vary depending on the validity measure. For cohesion, higher values are better, for separation, lower values are better.
- **Graph-Based View of Cohesion and Separation**

$$\begin{aligned} cohesion(C_i) &= \sum_{\substack{\mathbf{x}\in C_i\\ \mathbf{y}\in C_i}} proximity(\mathbf{x},\mathbf{y}) \\ separation(C_i, C_j) &= \sum_{\substack{\mathbf{x}\in C_i\\ \mathbf{y}\in C_j}} proximity(\mathbf{x},\mathbf{y}) \end{aligned}$$

    -
    - Cohesion of a cluster = sum of the weights of the links in the proximity graph that connect points within the cluster.
    - Separation between two clusters = sum of the weights of the links from points in one cluster to points in the other cluster.
    - The proximity function can be a similarity, a dissimilarity, or a simple function of these quantities
- Prototype-Based View of Cohesion and Separation

$$\begin{aligned} cohesion(C_i) &= \sum_{\mathbf{x}\in C_i} proximity(\mathbf{x},\mathbf{c}_i) \\ separation(C_i, C_j) &= proximity(\mathbf{c}_i,\mathbf{c}_j) \\ separation(C_i) &= proximity(\mathbf{c}_i,\mathbf{c}) \end{aligned}$$

    -
    - Cohesion of a cluster = sum of the proximities with respect to the prototype (centroid or medoid) of the cluster.
    - Separation between two clusters = proximity of the two cluster prototypes.
    - There are two measures for separation because, the separation of cluster prototypes from an overall prototype is sometimes directly related to the separation of cluster prototypes from one another
- Overall Measures of Cohesion and Separation
    - The previous measures (cohesion & separation) can be combined into an overall measure of cluster validity by using a weighted sum.

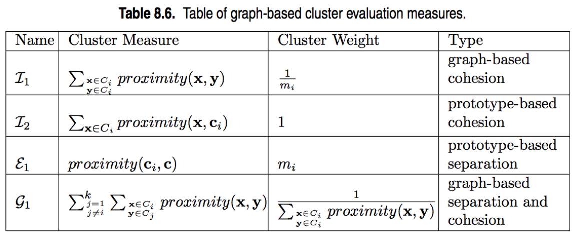


Table 8.6. Table of graph-based cluster evaluation measures.

Name	Cluster Measure	Cluster Weight	Type
$\mathcal{I}_1$	$\sum_{\substack{\mathbf{x}\in C_i\\ \mathbf{y}\in C_i}} proximity(\mathbf{x},\mathbf{y})$	$\frac{1}{m_i}$	graph-based cohesion
$\mathcal{I}_2$	$\sum_{\mathbf{x}\in C_i} proximity(\mathbf{x},\mathbf{c}_i)$	1	prototype-based cohesion
$\mathcal{E}_1$	$proximity(\mathbf{c}_i,\mathbf{c})$	m_i	prototype-based separation
$\mathcal{G}_1$	$\sum_{\substack{j=1\\ j\neq i}}^{k} \sum_{\substack{\mathbf{x}\in C_i\\ \mathbf{y}\in C_j}} proximity(\mathbf{x},\mathbf{y})$	$\frac{1}{\sum_{\substack{\mathbf{x}\in C_i\\ \mathbf{y}\in C_i}} proximity(\mathbf{x},\mathbf{y})}$	graph-based separation and cohesion

    -
- Relationship between Prototype-Based Cohesion and Graph-Based Cohesion
    - For some proximity measures, the measures of cohesion and separation are equivalent for Graph-based and prototype-based approaches even though

they seem distinct.

  - For instance, for the SSE and points in Euclidean space, it can be shown that the average pairwise distance between the points in a cluster is equivalent to the SSE of the cluster

    $$\text{Cluster SSE} = \sum_{\mathbf{x}\in C_i} dist(\mathbf{c}_i, \mathbf{x})^2 = \frac{1}{2m_i}\sum_{\mathbf{x}\in C_i}\sum_{\mathbf{y}\in C_i} dist(\mathbf{x}, \mathbf{y})^2$$

  -

- Two Approaches to Prototype-Based Separation

  $$\text{Total SSB} = \sum_{i=1}^{K} m_i \; dist(\mathbf{c}_i, \mathbf{c})^2$$

  -

  - When proximity is measured by Euclidean distance, the traditional measure of separation between clusters is the between group sum of squares (SSB), which is the sum of the squared distance of a cluster centroid.
  - The higher the total SSB of a clustering, the more separated the clusters are from one another.

- Relationship between Cohesion and Separation
  - In some cases there is a strong relationship b/w cohesion and separation. It is possible to show that the sum of the total SSE and the total SSB is a constant.
  - Total sum of Squares (TSS) = sum of squares of the distance of each point to the overall mean of the data.
  - TSS = SSE + SSB
  - The importance of this result is that minimizing SSE (cohesion) is equivalent to maximizing SSB (separation).

- Evaluating Individual Clusters and Objects
  - Many of the measures that we studied above can be used to evaluate individual clusters and objects.
  - We could rank individual clusters according to their specific value of cluster validity (cohesion or separation) and make some observations -
  - Since higher value of cohesion may be better than lower, we could decide to split a cluster with lower value.
  - If separation between clusters is low, we could decide to merge the clusters.
  - Objects that contribute more to the cohesion and separation are probably near the interior of the cluster and others are probably near the edge.

- The Silhouette Coefficient
  - How to compute Silhouette coefficient
    - For the i th object, calculate its average distance to all other objects in its cluster. Call this value a_i.
    - For the i th object and any cluster not containing the object, calculate the object's average distance to all the objects in the

given cluster. Find the minimum such value with respect to all clusters; call this value b_i.

  - Silhouette coefficient, $s_i = (b_i - a_i)/\max(a_i, b_i)$.

- The value of the silhouette coefficient can vary between –1 and 1.
- When $a_i = 0$, $s_i = 1$

- **8.5.3 Unsupervised Cluster Evaluation Using the Proximity Matrix**
  - Measuring Cluster Validity via Correlation
    - Input -> Similarity matrix for a data set and the cluster labels from a cluster analysis of the data set.
    - Output -> Evaluate "goodness" of the clustering by looking at the correlation between the similarity matrix and the cluster labels.
    - While computing the correlation, since the matrices are symmetric, only the correlation between n (n - 1) / 2 entries needs to be calculated.
    - High correlation indicates that points that belong to the same cluster are close to each other.
  - Judging a Clustering Visually by Its Similarity Matrix
    - If we sort the rows and columns of the similarity matrix so that all objects belonging to the same class are together, then an ideal similarity matrix has a block diagonal structure.
    - Well-separated clusters show a very strong, block- diagonal pattern in the reordered similarity matrix.
  - Computation of the proximity matrix takes O(m ^ 2) time
- **8.5.4 Unsupervised Evaluation of Hierarchical Clustering**
  - Previous approaches are for paritional clusterings.
  - Cophenetic correlation is a popular evaluation measure for hierarchical clusterings
  - **Cophenetic distance between two objects** = Proximity at which an agglomerative hierarchical clustering technique puts the objects in the same cluster for the first time.
  - The **CoPhenetic Correlation Coefficient (CPCC)** is the correlation between the entries of the cophenic distance matrix and the original dissimilarity matrix and is a standard measure of how well a hierarchical clustering (of a particular type) fits the data.
- **8.5.5 Determining the Correct Number of Clusters**
  - Plot SSE and Silhouette co-efficient for different values of k.
  - Choose k where there is a distinct knee in the SSE and a distinct peak in the silhouette coefficient
- **8.5.6 Clustering Tendency**
  - Clustering algorithms produce clusters for any input data. But can the data even be clustered ?

    - One approach is to evaluate the resulting clusters and only claim that a data set has clusters if at least some of the clusters are of 'good' quality. To make sure that the poor quality is not due to wrong choice of algorithm, we could use multiple algorithms and again evaluate the quality of the resulting clusters.
    - Another approach is to try to evaluate whether a data set has clusters without clustering. Most common approach, especially for data in Euclidean space, has been to use statistical tests for spatial randomness

- **8.5.7 Supervised Measures of Cluster Validity**
    - If we have external information about data, say in the form of class labels for the data objects, we could measure the degree of correspondence between the cluster labels and the class labels.
    - We could have different approaches to measure the extent to which two objects that are in the same class are in the same cluster. There are 2 main approaches - classification oriented and similarity oriented.
    - Classification-Oriented Measures of Cluster Validity
        - Uses measures from classification, such as entropy, purity, and the F-measure. These measures evaluate the extent to which a cluster contains objects of a single class.
        - Entropy : The degree to which each cluster consists of objects of a single class
        - Purity: Another measure of the extent to which a cluster contains objects of
        - a single class.
        - Precision : The fraction of a cluster that consists of objects of a specified class.
        - Recall : The extent to which a cluster contains all objects of a specified class.
        - F-measure : A combination of both precision and recall that measures the extent to which a cluster contains only objects of a particular class and all objects of that class.
    - Similarity-Oriented Measures of Cluster Validity
        - We can view this approach to cluster validity as involving the comparison of two matrices:
            - (1) the ideal cluster similarity matrix discussed previously, which has a 1 if two objects are in the same cluster and 0 otherwise, and
            - (2) an ideal class similarity matrix defined with respect to class labels, which has 1 if objects are in the same class and 0 otherwise.
        - Rand statistic and Jaccard coefficient are two of the most frequently used cluster validity measures.

$$\text{Rand statistic} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

- $$\text{Jaccard coefficient} = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

- Cluster Validity for Hierarchical Clusterings
  - Supervised evaluation of a hierarchical clustering is more difficult for a variety of reasons, including the fact that a preexisting hierarchical structure often does not exist
  - To evaluate a hierarchical clustering, we compute,
    - for each class, the F-measure for each cluster in the cluster hierarchy.
    - For each class, we take the maximum F- measure attained for any cluster.
    - Finally, we calculate an overall F-measure for the hierarchical clustering by computing the weighted average of all per-class F-measures, where the weights are based on the class sizes.

- **8.5.8 Assessing the Significance of Cluster Validity Measures**
  - How do we interpret the significance of the number given by a cluster evaluation technique ?
  - Need to be aware of the basic definitions of evaluation measures used, like - a purity of 0 is bad, while a purity of 1 is good, an entropy of 0 is good, as is an SSE of 0.
  - If we don't have an absolute standard, a common approach is to interpret the value of our validity measure in statistical terms. e.g - plotting a histogram of SSE for random data sets.

Wow ! That was a really long chapter and took 1 full day to read through! Thanks to Tan, Steinbach and Kumar for an excellent narrative.

I really cannot forget this quote -

> *Just as people can find patterns in clouds, data mining algorithms can find clusters in random data. While it is entertaining to find patterns in clouds, it is pointless and perhaps embarrassing to find clusters in noise.*

Source : Chapter 8 from

- Power Point slides : http://www-users.cs.umn.edu/~kumar/dmbook/index.php#item4
- Sample Chapters : http://www-users.cs.umn.edu/~kumar/dmbook/index.php#item2

Posted by Robinson Raju

Labels: Data Mining, Intro to Data Mining - TSK - Chapter 8, Introduction-to-Data-Mining-Tan-Steinbach-Kumar

No comments:

Post a Comment

Enter your comment...

Comment as: Robinson Raju (Sign out

Publish Preview Notify me

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

Blog Archive

- ▼ 2014 (12)
  - ► December 2014 (1)
  - ► June 2014 (5)
  - ▼ May 2014 (1)
    - TSK - Chapter 8 - Cluster Analysis
  - ► February 2014 (5)

Search This Blog

Search

Labels

Data Mining (6)
Introduction-to-Data-Mining-Tan-Steinbach-Kumar (6)
Clojure (5)
Clojure-Koans (1)
Intro to Data Mining - TSK - Chapter 1 (1)
Intro to Data Mining - TSK - Chapter 2 (1)
Intro to Data Mining - TSK - Chapter 3 (1)
Intro to Data Mining - TSK - Chapter 4 (1)
Intro to Data Mining - TSK - Chapter 8 (1)
getting-started-with-clojure-part-1 (1)
getting-started-with-clojure-part-2 (1)
getting-started-with-clojure-part-3 (1)