CS60050 Machine Learning - Weekly Report

Nisarg Upadhyaya (19CS30031)

Week 8: 29 September and 1 October

1 Topics covered

- How to determine K?
- Generalizing K-Means and Expectation Maximization
- Hierarchical Clustering
- Graph Based Approaches
- Approximate methods to solve Corrupted Cliques Problem
- DBSCAN

2 Summary

2.1 How to determine K?

2.1.1 Cluster Validity Indices

We can use external indices using some existing partiotioning information such as class labels. Different heuristics can be used like *Normalized Mutual Information* which is based on entropy measurement of the clusters, *FM Index* which is the fraction of same pairs in the same cluster or by finding matching partion pairs. We can also use internal indices by looking at the variance distribution and structure of clusters by using measures such as the *Silhoutte Index* or the *Calinski-Harabasz Index*.

2.1.2 Stability Check Based Clustering

For an appropriate K repeated clustering should have similar partitioning. This shows that the partitioning is stable. In *Wang's* method we permute the input data c times. Each time we divide it into test set and two equal training sets. K-means is performed on both training sets and the number of disagreements (a pair from test set being classified in the same or different clusters by the two training sets) are calculated. We choose the K which minimizes the average number of disagreements.

2.2 Generalizing K-Means

The mixture density is written as $P(x) = \sum_{i=1}^{K} P(x|G_i)P(G_i)$ where K is the number of components, $P(G_i)$ are the mixture proportions and $P(x|G_i)$ are the component densities. If the component densities are multivariate Gaussian then we have $P(x|G_i) \sim \mathcal{N}(\mu_i, \Sigma_i)$ and we need to estimate μ_i, Σ_i and $P(G_i)$ from the set of *iid* samples.

2.2.1 Mixture of Gaussians

When each class is Gaussian distributed, we have a Gaussian mixture. Each cluster center is augmented by a covariance matrix, whose values are re-estimated from corresponding samples. We use *Mahalanobis* distance to calculate distance from any x to its cluster center and it is given by $(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)$ where μ_k is the cluster center and Σ_k is the covariance matrix.

2.2.2 Expectation Maximization Algorithm

We assign a sample x to the cluster in which its belongingness is maximum. Starting with an initial set of parameters $\{\pi_k, \mu_k, \Sigma_k\}$. We compute probability z_{ik} of x_i belonging to the k-th cluster and assign it to the cluster where we get the maximum probability where:

$$z_{ik} = \frac{1}{Z_i} \pi_k N(x_i|\mu_k, \Sigma_k), Z_i = \sum_k \pi_k N(x_i|\mu_k, \Sigma_k)$$

Then we re-estimate the parameters from the class distribution. The re-estimation is done as follows

$$\mu_k = \frac{1}{N_k} \sum_i z_{ik} x_i, \Sigma_k = \frac{1}{N_k} \sum_i z_{ik} (x_i - \mu_k)(x_i - \mu_k)^T, \pi_k = \frac{N_k}{N}$$

Here N is the expected number of pixels in class k that is $N_k = \sum_i z_{ik}$. The above two steps are repeated till convergence.

2.3 Hierarchical Clustering

This is a non-probabilistic approach which builds a hierarchy of groups using a bottom-up approach. It uses a distance matrix among the samples. Given a distance matrix d and n clusters each with one element we initialise a graph T with a vertex for each cluster. While there is more than one cluster the following steps are repeated:

1. Find the two closest clusters C_1 and C_2 and merge C_1 and C_2 into C with $|C_1| + |C_2|$ elements.
2. Compute distance from C to all other clusters and add a new vertex C to T and connect to vertices C_1 and C_2.
3. Remove rows and columns of d corresponding to C_1 and C_2 and add a row and column to d corresponding to the new cluster C.

Different measures can be used to compute distance between pair of clusters. For example it can be taken as the smallest distance between any pair of their elements or the average distance between all pairs of their elements.

2.4 Graph Based Approaches and Distance Graphs

We can also use graph based approaches where we form a graph from the input data. In distance graphs we represent feature vectors as vertices in a graph. A distance threshold θ is chosen and those vertices are connected by an edge whose distance is less than θ. The resulting graph may contain cliques which represent clusters of closely located data points. A clique is a graph with every vertex connected to every other vertex. A clique graph is a graph where each connected component is a clique. The *Corrupted Cliques Problem* involves finding the smallest number of additions and removals of edges that will transform an arbitrary graph into a clique graph. It is a NP-Hard problem. There are some approximate methods to solve it:

1. Parallel Classification with Cores: Let S be a set of n elements, k be the number of clusters and G be the distance graph. We randomly select S', a subset from S and S'' a subset from $S - S'$, such that $|S'| = log(log(n))$ and $|S''| = log(n)$. For all k partitions in S' first we extend the clustering from S' to S'' and then from S'' to $S - (S' \cup S'')$ and choose that partition in which minimum edges are required to be added or removed from G to get a Clique graph as per the partition. This algorithm runs in $O(n^2 log(n)^{log(k)})$ time.
2. Cluster Affinity Search Technique: We define distance between feature i and cluster C as $d(i, C)$ = average distance between feature i and all other features in C. Let S be a set of elements and G be the distance graph. While S is not empty we initialise a cluster C to the vertex of maximal degree in the distance graph G. We keep adding nearest closest feature i not in C and removing the farthest distant feature i in C while either exists. We add this cluster to the partition set and update the set S and graph G by removing the vertices of cluster C.

2.4.1 DBSCAN

Density-based spatial clustering of applications with noise involves growing regions of connected core points from a seed. A core point is any such point p such that the number of points within a distance ϵ from point p is more than a minimum threshold. The points lying in this ϵ region are called neighbors. A neighbor but not a core point is called a border point. A point not reachable via recursive search from any core point is considered as noise. It uses R-tree for efficient search.

3 Challenging concepts

Expectation Maximization Algorithm

4 Interesting concepts

DBSCAN

5 Concepts not understood

None

6 Ideas

In DBSCAN the parameter ϵ plays a crucial role. In fact, it needs to be chosen appropriately based on the data. A very small value can lead to a large number of clusters and a large value can cause unrelated clusters to merge. The minimum threshold parameter also plays a crucial role. Exhaustively searching for optimal values can take a lot of time considering we have two parameters to set. However, it looks as if the effect of change of parameter is monotonous. Increasing the value of ϵ will lead to a decrease in the number of clusters. We can employ some sort of binary search in this case to arrive at optimal values of ϵ and the minimum threshold because of this monotonous behavior.