Refining Initial Points for K-Means Clustering

Clustering is an important area of application for a variety of fields. The fundamental clustering problem is that of grouping together (clustering) data items which are similar to each other. The clustering problem is sensitive to initial starting conditions. In this paper we find a method which could give us refined points and could help us to perform a better and effective clustering. The method discussed in the paper is scalable and very powerful when we're working on very large datasets. One of the advantage of the method is that there's no assumption used in it so there are no exceptional cases where the method won't work. We assume that in addition to the observed variables for each data item, there is a hidden, unobserved variable indicating the "cluster membership" of the given data item. Hence the data is assumed to arrive from a mixture model and the mixing labels (cluster identifiers) are hidden. One method for initialising the points for K-means is taking the mean of the entire data and then randomly perturbing it K times . This method does not appear to be better than random initialization in the case of EM over discrete data. Another method for clustering is to estimate the density and attempt to find the maxima ("bumps") of the estimated density function. Density estimation in high dimensions is difficult as is bump hunting so this method cannot be used. So our selected method refines the initial point to a point likely to be closer to the modes. The basic heuristic is that severely subsampling the data will naturally bias the sample to representatives "near" the modes. In general, one cannot guard against the possibility of points from the tails appearing in the subsample. We have to overcome the problem that the estimate is fairly unstable due to elements of the tails appearing in the sample. A clustering session on a data set with many dimensions and tens of thousands or millions of records can take hours to days but our method has a great advantage over here that it uses very less RAM even for huge databases because the method works on very small samples of data at a time. We did experiments on synthetic data as well as real world data. For synthetic data we created for dimension d = 2, 3, 4, 5, 10, 20, 40, 50 and 100. For a given value of d, data was sampled from 10 Gaussians. We divided the data into various type of subsamples- 10 subsamples and single random subsample and did the refinement and we compared the results with non-refined data. In most subsamples we got better result than non-refined data. Then we did the experiments on real world data. We were primarily more interested in large databases -- hundreds of dimensions and tens of thousands to millions or

records. The reason was simple: a clustering session on a large database is a time-consuming affair. Hence a refined starting condition can insure that the time investment pays off. Firstly we took Image Segmentation data set from UCI ML Repository. The amount of information gained on average by the solutions computed from the refined point was 2.6222 time that of the solution computed over the random initial point. on average, solutions computed from the refined initial points (J=10) reduced distortion by 44.41% over solutions computed from random initial points. Results on the small real-world Image Segmentation data set indicate that the K-Means solution from the refined points provide twice as much "information" than the solutions computed from the random initial point. Furthermore, the average distortion is decreased by 9%. Now we did the experiment on Reuters Information Retrieval Data Set. This data had high dimensions and this was a large dataset. . On average the distortion of a solution obtained by starting from a refined initial point was about 80% of the corresponding distortion obtained by clustering from the corresponding randomly chosen initial starting point. Computational results on the Reuters database of newswire stories in 300 dimensions indicate a drop in distortion by about 20%. Information gain was improved by a factor of 4.13 times on this data set.

There were only 2 cons which I found in the method-

1. Due to subsampling some subsamples may provide noisy estimates mostly in case of high dimensions and skewed distributions.
2. Due to noisy estimates the smoothing should be done in a very optimal way which is very tough when we have very huge datasets.

The refinement algorithm operates over small subsamples of a given database, hence requiring a small proportion of the total memory needed to store the full database and making this approach very appealing for large-scale clustering problems. The procedure is motivated by the observation that subsampling can provide guidance regarding the location of the modes of the joint probability density function assumed to have generated the data. By initializing a general clustering algorithm near the modes, not only are the true clusters found more often, but it follows that the clustering algorithm will iterate fewer times prior to convergence. This is very important as the clustering methods discussed here require a full data-scan at each iteration and this may be a costly procedure in a largescale setting.

In future we can improve the method for subsampling so that noisy estimates could be removed. We can find a way for subsampling in an optimal way so that all the similar data points which produce less variance could be refined together. We could research for the implementation for this method in other clustering methods also like Hierarchical clustering, Agglomerative clustering etc as this method could help a lot to reduce time complexity for huge databases.