CSE 5243: Introduction to Data Mining Assignment 4

Vaibhav Devekar (devekar.1)  
Akshay Nikam (nikam.5)

# Goal of the Assignment

This assignment aims at demonstrating various clustering techniques on the reuters document dataset based on the feature vectors generated in Assignment 1.

# Distance Metrics:

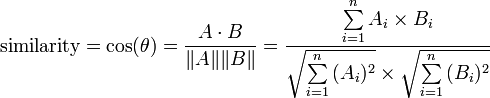
We attempted running our algorithms with two different distance metrics:

1. **Cosine distance**

Cosine similarity between two equal dimensional vectors is the cosine of the angle between the two vectors. Smaller is the angle, larger is the cosine similarity. Cosine distance is defined as:

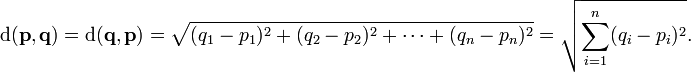
Cosine distance = 1 – cosine similarity

Where,

Cosine similarity between vector A and B is calculated as:

1. **Euclidean distance**

Euclidean distance is the actual point-to-point Cartesian distance between two points in a vector space. For an n-dimensional space, Euclidean distance (d) between points p and q is calculated as:



These metrics were used to compute distance between a pair of articles based on the feature vector generated.

# Clustering Techniques

We implemented the following clustering algorithms:

### K-means clustering:

This is a convergence based algorithm that works in two steps per iteration. The algorithm expects the number of clusters K as an input. We start with K random mean points, one mean for each cluster. The means are represented as a vector of the same dimensionality as the feature vector of the articles. Each article acts as a point for this algorithm.

In one iteration, first step is to find the closest cluster mean for each point and assign the point to that cluster. Each cluster gets a set of points nearby its mean. In the second step, all cluster means are recomputed. This is straightforward by taking a mean of each dimension and forming a mean vector.

The clustering algorithm converges when in two successive iterations no points or only a small number of points are reassigned to different clusters. This can also be represented in terms of change in the cluster means. For the computed data matrix from assignment 1, our implementation of K-means converges in less than 40 iterations.

The running time is roughly directly proportional to the number of clusters K provided as input. We took results for K=2,4,8,16,32,64.

# Results and Performance:

For results and performance computation, we considered topics and place labels of each article in each cluster and used following metrics to evaluate performance and the quality of clustering.

### Entropy:

For finding out how the algorithm performed, we used entropy as the performance metric. It is a measure of uncertainty and depicts the number of bits required to represent a point in the given data.

Computing entropy:

Entropy is first individually computed for each cluster and then a weighted average is taken based on the number of points in the clusters. Since each article has multiple topics and places, we gave each label of the article an equal weight, such that the total weight of all topics and place labels for an article is always 1. For example, if an article has 2 topics and 1 place, each are given 1/3 weight. Now, each topic’s/place’s weight is summed over all the records in that cluster. Once we get a vector of total weights of all class labels, we normalize it to find probabilities of each topic/place. Then entropy for the cluster is computed as:

   \displaystyle
   H(X)= - \sum_{i=1}^np(x_i)\log_b p(x_i)

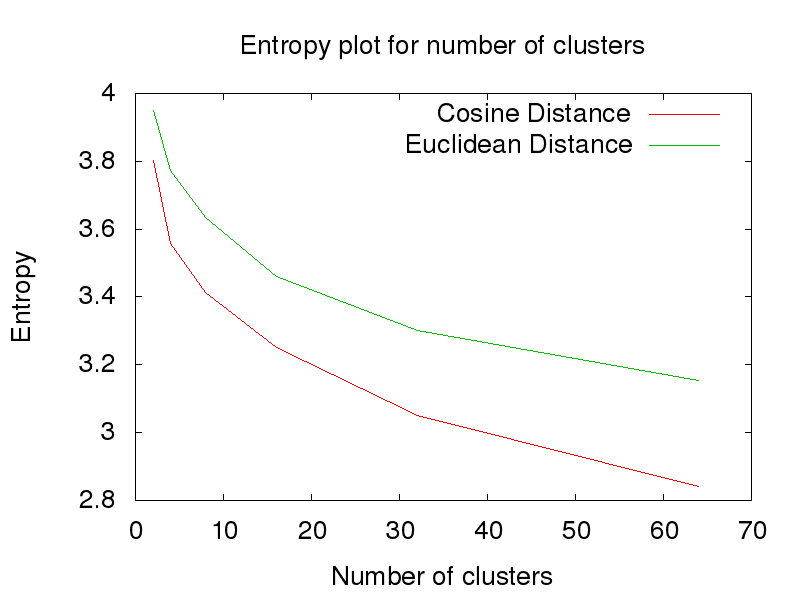

For n class values and b=2.

We found that as the number of clusters increase, entropy gradually decreases, meaning a better quality of separation of articles.

For K-means clustering algorithm, we found following entropies:

|  |  |  |
| --- | --- | --- |
| **# Clusters** | **Entropy for Cosine Distance** | **Entropy for Euclidean Distance** |
| 2 | 3.80375332771 | 3.94993482586 |
| 4 | 3.55697535925 | 3.77195219452 |
| 8 | 3.41406922379 | 3.63348425172 |
| 16 | 3.25008232272 | 3.46016354507 |
| 32 | 3.05113063415 | 3.29978197132 |
| 64 | 2.84177028704 | 3.23467349883 |

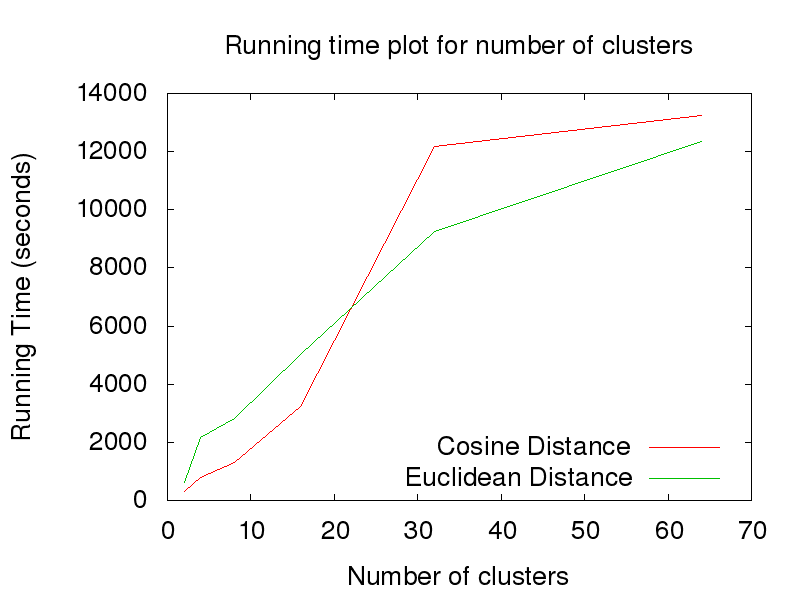
According to the following plot, it can be observed that Cosine distance produces lower entropy and hence is a better distance metric for clustering given data.



### Running Time:

Kmeans:

It is found that running time is directly proportional to the number of clusters and the number of iterations required to converge. The following plot shows the results for the same:



### Skew:

# Individual Contributions

Akshay implemented the K-mean clustering algorithm while Vaibhav implemented Max-link hierarchical clustering. Both the algorithms were implemented in Python.