2009

Effects of similarity metrics on document clustering

Rushikesh Veni
University of Nevada Las Vegas

EFFECTS OF SIMILARITY METRICS ON DOCUMENT CLUSTERING

By

Rushikesh Veni

Bachelor of Engineering in Computer Science and Engineering
Osmania University, India
May 2007

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree in Computer Science
School of Computer Science
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
August 2009

ABSTRACT

Effects of Distance Metrics on Document Clustering

by
Rushikesh Veni

Dr. Kazem Taghva, Examination Committee Chair
Professor, Department of Computer Science
University of Nevada, Las Vegas

Document clustering or unsupervised document classification is an automated process of grouping documents with similar content. A typical technique uses a similarity function to compare documents. In the literature, many similarity functions such as dot product or cosine measures are proposed for the comparison operator.

For the thesis, we evaluate the effects a similarity function may have on clustering. We start by representing a document and a query, both as a vector of high-dimensional space corresponding to the keywords followed by using an appropriate distance measure in k-means to compute similarity between the document vector and the query vector to form clusters. Based on these clusters we decide the best distance metric for the document set used. Next, we compute time complexities for different similarity functions for the same model and document set based on the number of iterations and number of clusters.

TABLE OF CONTENTS

ABSTRACT iii

LIST OF TABLES v

LIST OF FIGURES vi

ACKNOWLEDGEMENTS vii

CHAPTER 1 INTRODUCTION 1

CHAPTER 2 BACKGROUND 4
2.1 Introduction to Clustering 5
2.2 Types of Data in Cluster Analysis 6
2.3 Different Clustering Methods 8
2.3.1 Different types of Clusters 14

CHAPTER 3 K-MEANS CLUSTERING 19
3.1 K-Means Clustering Algorithm 14
3.2 A Numerical Example of K-Means Algorithm 19
3.3 Distance Metrics 25
3.3.1 Different Distance functions Used 26
3.4 Time Complexities for K-means using different metrics 30

CHAPTER 4 IMPLEMENTATION 34
4.1 Documents Collection 34
4.2 Documents Preprocessing 36
4.3 TF * IDF Calculation 43
4.4 Algorithm Implementation 46
4.4.1 Experiments over different Metrics 39

CHAPTER 5 RESULTS EVALUATION 54
5.1 Comparison based on Clusters 54
5.2 Comparison based on Execution Time 63

CHAPTER 6 CONCLUSION AND FUTURE WORK 66

BIBLIOGRAPHY 68

VITA 71

LIST OF TABLES

Table 1 Medicine Objects with both the Attributes 19
Table 2 Distances between Data points and Centroids 21
Table 3 Final Clusters formed with Data points..... 25
Table 4 Reuters-21578 Collection Categories 33
Table 5 Training Set Collection 34
Table 6 Example text; each line is one Document..... 46
Table 7 Inverted file for text in table 6 48
Table 8 No of documents in each cluster for Canberra 55
Table 9 Documents in each cluster for Bray-Curtis 56
Table 10 Documents in each cluster for Variational 57
Table 11 Documents in each cluster for Chi-square 58
Table 12 Documents in each cluster for trigonometric..... 60
Table 13 No of Documents in each cluster for euclidean..... 61
Table 14 Showing Execution Time for Distances 63

LIST OF FIGURES

Figure 1 Different clusters for the same set of points 6
Figure 2 Data Matrix 7
Figure 3 Dissimilarity Matrix 8
Figure 4a Traditional Nested Set 9
Figure 4b Traditional Dendogram 9
Figure 4c Non-Traditional Nested Set 9
Figure 4d Non-Traditional Dendogram 9
Figure 5 Steps involved in K-Means Algorithm 15
Figure 6 Initial Centroids Leading to Poor Clusters 17
Figure 6 Initial Centroids Leading to Better Clusters 18
FigurE 8 Points plotted representing medicine objects 19
FigurE 9 Plotting initial centroids of medicine objects 20
Figure 10 Recalculating centroids after first iteration 22
Figure 11 New centroids and clusters after Iteration 2 24
Figure 12 A screenshot of Reuters 21578 collection 35
Figure 13 Parsed XML document 36
Figure 14 List of tokens after tokenization 37
Figure 15 A Screenshot of List of stop words 38
Figure 16 A Screenshot of output after stemming 44
Figure 17 A Screenshot of Term frequency matrix 44
Figure 18 A Screenshot of TF * IDF matrix 45

ACKNOWLEDGEMENTS

I would like to express my gratitude to many people who helped and directed me to pursue my goal of gaining an education in United States of America. Firstly I would like to thank the faculty and staff of Department of Computer Science, University of Nevada Las Vegas. My special thanks to Dr. Kazem Taghva, who has been more than a mentor and an advisor to me. It is only because of his support and back-up I'm able to finish my research work. I sincerely thank my graduate coordinator Dr. Ajoy K Datta for his help and invaluable support through my masters program. I also would like to thank other members of my committee, Dr. Laxmi P. Gewali and Dr. Venkatesan Muthukumar. I also wish to thank Dr. Matt Pederson who has been very cooperative to me during my Teaching Assistantship under him.

I would also like to extend my appreciation towards my parents, my brother, and my cousins, for being there for me through thick and thin and always encouraging me to strive for the best. Without their endless support I would never be able to reach to the place I'm standing today in my life. Last but not the least, I thank my friends, roommates for their support in the successful completion of this work.

CHAPTER 1

INTRODUCTION

Document Clustering is a technique used in unsupervised document organization for identifying clusters or forming group of documents such that the documents in the same cluster are more similar to one another than they are to the documents in other cluster. This technique can be used in information retrieval to automatically categorize large collection of retrieval results by grouping similar type of documents together that helps user's browsing of retrieval results [1].The data objects within one group should provide higher degree of similarity and should be minimized when compared to other clusters.

Documents can be classified into 2 types. 1) Supervised Learning and 2) Unsupervised Learning. In Supervised Learning, the model defines the effect one set of observations called inputs has on other sets of observations, called outputs whereas the observations are assumed to be at the end of casual chain. Clustering is a form of unsupervised learning defined as a process of grouping a set of physical or abstract objects into classes of similar objects. Huge document collection is used to analyze the clusters formed for different distance metrics along with the

execution time. As the number of documents collection increases this bottleneck prevents a more widespread deployment of clustering for information retrieval. To avoid this, we apply different thresholds throughout the cluster generation process and come up with a best suited clustering procedure that can be applied to our document collection [2].

1.1 Thesis Overview

The research involves clustering documents into categories using K-Means clustering algorithm. We choose different distance metrics for K-means clustering algorithm to form clusters apart from the generic ones like Euclidean and Cosine distance measures. The cluster numbers can be modified to see how different clusters are formed. Also we run clustering algorithm to find the time complexities for K-means using different distance metric. Initially we start with data matrix obtained from the text documents after preprocessing steps. This data matrix is represented with each row as a document vector and each column as weight of a significant term. This data matrix is provided as an input to K-Means for clustering documents. The results obtained from above process are used to evaluate and compare different distance metrics and also their time complexities.

The thesis is organized into different chapters starting from introduction followed by the brief explanation about clustering and types

of clusters. Then different clustering methods are given concentrating more toward K-Means clustering algorithm. We discuss preprocessing steps involved in obtaining the weighted matrix that is applied to K-Means. This is continued with the implementation of K-Means. The results obtained from K-Means are analyzed and compared for different distance functions used to form clusters and time complexities of different metrics used. We conclude our thesis with brief overview of future work.

CHAPTER 2

BACKGROUND

Data Mining can be defined as the type of database analysis that attempts to extract useful patterns or relationships in a group of data. This analysis is in used statistical methods, such as cluster analysis and sometimes employs artificial intelligence or neural network techniques .A major goal of data mining is to extract previously unknown useful relationships among different data. There are different data mining techniques among which Clustering or unsupervised learning is the one used in the thesis.

Document Clustering is defined as unsupervised document organization, automatic topic extraction and fast information retrieval. For Example, in web search huge numbers of pages are returned when user enters a query making it difficult for user to browse or extract needed information where as clustering produces results automatically grouped into list of meaningful categories [4]. Document clustering has been investigated for use in different areas of text mining and information retrieval. Initially document clustering was used mainly for finding precision and recall in information retrieval. Recently, it has been

a major technique for use in browsing a collection of documents and for organizing the results obtained from a search engine in response to user query [5].

2.1 Introduction to Clustering

Clustering is defined as grouping a set of physical or abstract objects into classes of similar objects. Every data Object within a cluster is similar to one another and are dissimilar to the objects in other clusters. Early in Childhood, we learn to differentiate between cats and dogs or between animals and plants by continuously improving subconscious clustering schemes. Cluster analysis has a wider range of applications including pattern recognition, data analysis, market research and image processing. It is also used to classify documents on the web for information discovery [6].

Few typical requirements of clustering in data mining include scalability, ability to deal with different attributes, ability to deal with noise data, High dimensionality, constraint-based clustering, Interoperability and usability.

Depending on different requirements we discuss different types of data and different clustering methods [6]. The greater the similarity between objects in a cluster and greater the dissimilarity between objects of other clusters, the more tight are the clusters [4].

2.2 Types of Data in Cluster Analysis

Cluster Analysis groups objects based on the found in data describing the objects or their relationships. The greater the similarity between the objects within the cluster and greater the dissimilarity between data objects of other cluster constitutes a better clustering.

As mentioned in [7], the idea of cluster is imprecise, and the best definition depends on the type of data and the desired results. The following diagrams illustrate this statement.

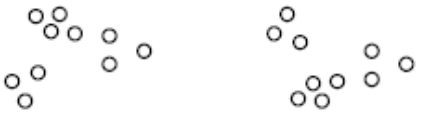

Figure 1a: Initial points.

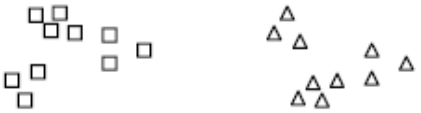

Figure 1b: Two clusters.

Figure 1c: Six clusters

Figure 1d: Four clusters.

Fig 1: Different clusters of the same set of points.

As we can see here same set of points are clustered in four different ways which leads to the ambiguity of definition of clustering. If we allow clusters to be nested, then the most reasonable interpretation about the structure formed here with these points is that there are two

clusters, each of which has three sub clusters. Finally, it may not be reasonable to call that these points form four clusters.

There are different types of data that often needs to be be preprocessed in cluster analysis. Main-Memory based clustering algorithms typically operate on either of the following two data structures.

1. Data Matrix (Object-by-variable structure): This matrix is represented by n objects such as persons and with p variables (also called as attributes), such as weight, age, gender, height and so on. The structure of matrix is in the forms of a relational table, or n-by-p matrix as shown in figure below.

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

Fig 2: Data Matrix

2. Dissimilarity Matrix (or object-by-object structure): This stores a collection of proximities that are available for all pairs of n objects. It is often represented by n-by-n table as shown in figure below.

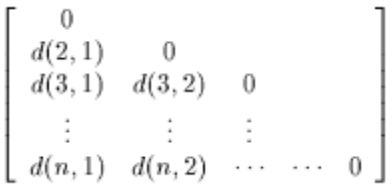

Fig 3: Dissimilarity Matrix

Where d (i, j) is the measured difference or dissimilarity between objects i and j. Since d (i, j) =d (j, i) and d (i, i) = 0, we have matrix in Fig 3.

The rows and columns of the data matrix represent different entities, while those of the dissimilarity matrix represent the same entity. If the data are represented in the form of a data matrix, it can first be transformed into a dissimilarity matrix before applying such clustering algorithms [6].

2.3 Different Clustering Methods

Many different clustering techniques have been proposed of which few are described that produce different clusters. They are classified into following categories [6].

Hierarchical versus partitioning methods (nested and unnested):

Hierarchical techniques produce a nested sequence of partitions, with a single, all inclusive cluster at the top and singleton clusters of individual points at the bottom. It produces a hierarchical tree structure with the leaves of tree are individual clusters of all object inputs and the cluster related to a particular node in the tree is the union of all clusters

related to the child nodes of that particular node. Following figures indicate the hierarchical clustering process.

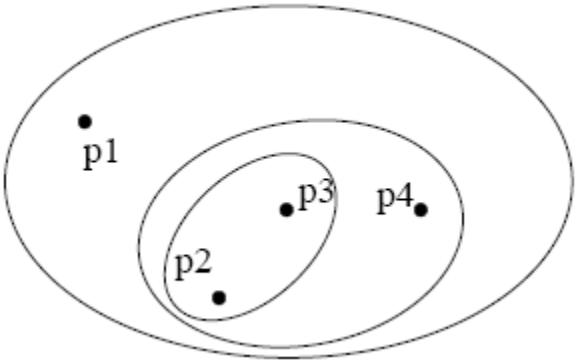

Fig 4a: Traditional nested set

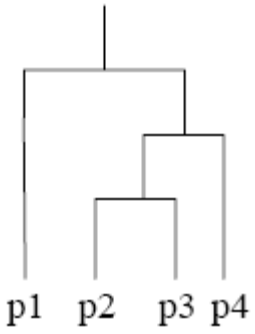

Fig 4b: Traditional dendogram

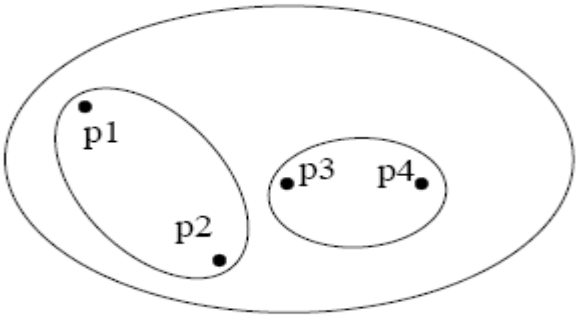

Fig 4c: Non-traditional nested set

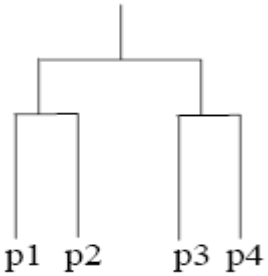

Fig 4d: Non-traditional dendogram

Figures 4a and 4b represent more traditional way of viewing hierarchical clustering as a process of merging two clusters or splitting one cluster into two.

Figure 4a gives the nested set representation and Fig 4b gives a tree structure representation or dendogram. Figure 4c and 4d show a different, hierarchical clustering; one in which p1 and p2 are grouped

together and other group has point 3 and point 4 in the same step. The agglomerative approach, also called as bottom-up approach starts with object forming a group. The divisive approach, also called as top-down approach, starts with all the objects in the same clusters.

Partition Techniques create un-nested clusters where data belongs to only one subset of clusters. If *K* is the number of clusters, then partitional approach typically finds all *K* clusters at once. Partitioning method includes K-Mean algorithm where each cluster is represented by mean value of the object in the cluster and K-Mediods algorithm where each cluster is represented by one of the objects located near the center of the cluster are popular heuristic methods.

Density-based methods:

The idea behind this technique is to continue growing the given cluster as long as the density (number of objects or data points) in the "neighborhood" exceeds some threshold. DBSCAN and its extension, OPTICS are typical density-based methods that grow clusters according to a density-based connectivity analysis. DENCLUE is a method that clusters objects based on the analysis of the value distributions of a density functions.

Grid-based methods:

Grid-based method first covers the problem space domain with a uniform grid mesh. Statistical attributes are collected for all the data objects located in each individual mesh cell and clustering, is then

performed on the grid, instead of data object themselves. The main advantage with this approach is faster processing time. STING is a typical example of grid-based method [8].

Model-based methods:

Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. EM is a Model-based algorithm that consists of two alternating steps: the Expectation (E) step and the Maximization (M) step based on statistic modeling. COBWEB is a conceptual learning algorithm that performs probability analysis and takes concepts as a model for clusters [6].

2.3.1. Different Types of Clusters

In [7], different types of clusters are described, of which few types are discussed below.

1. Well-separated: Here the points that form a cluster are close to one another than any other point that is not a part of the cluster. To find the maximum distance between the points in the cluster a threshold can be set.

2. Prototype based: Clusters formed by points more close or similar to a prototype. This prototype can be any object like the centroid or median representing that particular cluster and so are also called center based clusters.

3. Graph based: Clusters which have objects or points that are close to one or more points/objects in that particular cluster than any other

point not belonging to that cluster is called graph based or contiguity based clusters.

4. Density based: These clusters are a set of high density sections in a pool of low density sections. The only difference between density and graph based clusters is if the noise is added to the later one the bridge connecting the round clusters and the curves would no longer be considered because of their low density.

To form different cluster discussed above, different clustering algorithms have to be applied to the data objects. Different clustering algorithms include exclusive, hierarchical, fuzzy, probabilistic and so on. We apply these algorithms to our data sets to form clusters. To form final clusters from the document set the following points must be adapted:

1) The clustering must be exclusive which means no document should be a member of more than one cluster.

2) The clustering should be complete i.e., every document should be placed or should be a part of some cluster.

3) The clustering should be Partitional i.e. document belongs to just one subset of clusters and there are no overlapping of subsets of clusters.

Thus K-Means clustering provides a complete package of above discussed requirements and in the thesis we use K-Means algorithm for document set. A clear description of K-Means will be discussed in the next chapter along with the distance metrics used.

CHAPTER 3

K-MEANS CLUSTERING

K-means is one of the simplest unsupervised learning algorithms to group similar data objects. It was developed by J.MacQueen (1967) and then by J.A.Hartigan and M.A.Wong around 1975 [11].K-means forms clusters for n objects based on the attributes into k partitions where $k<n$. The algorithm starts by partitioning the input points into k initial sets, either at random or using heuristic data. It then calculates the mean point or centroid of each set. It constructs a new partition by associating each point with the closest centroid. Then the centroids are recalculated for new clusters, and the algorithm is repeated by alternate application of these two steps until convergence, which is obtained when the points no longer switch clusters. The centroids should be placed in a cunning way as different centroid location provides different results [13].

A very popular and efficient heuristic for K-means clustering is Lloyd's algorithm which is discussed in detail in [9].

3.1 K-Means Clustering Algorithm

The main goal using K-means algorithm is to minimize the objective function, shown below

$$J = \sum_{j=1}^{K} \sum_{n \in S_j} |x_n - \mu_j|^2,$$

where $||x_i^{(j)} - c_j||^2$ is a distance measure between a datapoint $x_i^{(j)}$ and cluster center c_j , showing the distance between n data points to their respective cluster centroids[12].

This above equation clearly specifies that clusters are formed by minimizing the distance between the centroid and the data point. The algorithm begins with assigning k centroids choosen randomly in a plane. All the points in the data set are assigned to a centroid that is nearest to it forming clusters. Once this initial arrangement is done, the next step will be to recalculate the centroid in each cluster by finding the center of the cluster from first step. This centroid is the point that is equidistant from all the points in that cluster. The next step is to again assign each point in the data set to the centroid in each cluster by finding the minimum distance between each point and every cluster and choosing the one with the minimum distance. Once again new centroid for every cluster is calculated. This looping is repeated until k centroids do not change their location. The diagrammatic representation of K-means algorithm is shown below followed by the steps in the process.

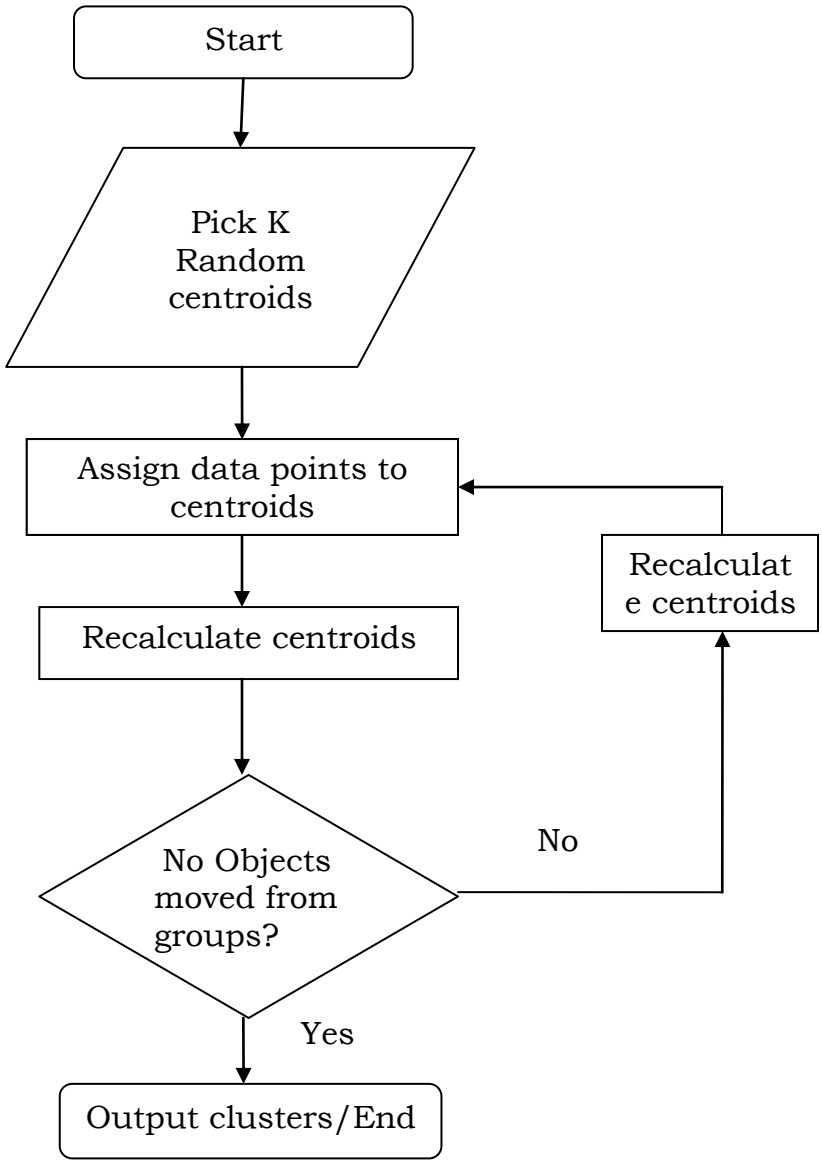

Figure 5. K-Means Algorithm

This algorithm involves following steps:

1) Pick K points at random represented as initial centroids for k clusters into the space.

2) Assign each point to the centroid from which it has minimum distance using distance metrics.

3) Recalculate the centroids after assigning all the points in the clusters.

4) Check for the centroids if they have moved their positions in the clusters. If they have changed their location from previous iteration go to step 2, else if the locations are not changed then the clusters formed separates the objects into different groups based on the distance metrics used.

K-means is a simple greedy algorithm for partitioning *n* objects into *k* clusters by iteratively moving the centroid locations in order to finally get optimal positions which is explained in [12]. The results for forming clusters greatly depends on choosing the number of clusters i.e., k value. A simple approach is to compare results from multiple runs changing k value and choosing the best one according to the given criteria, but needed to be careful as increasing k results in not only smaller error function values by definition, but also an increasing risk of over fitting.

K-means algorithm results largely depend on 3 factors:

1) The value of k (number of clusters)

2) Choosing the centroids(either randomly or using some function)

3) The distance metric used for calculating distance between the data object and the centroid which is concentrated more in the thesis in later chapters.

For the method discussed above, a following pictorial representation taken from [7] shows how the final clusters change with the choice of initial clusters.

In figure 6 below, the algorithm stops after 5 iterations as the clusters does not change though the results produced are not effective.

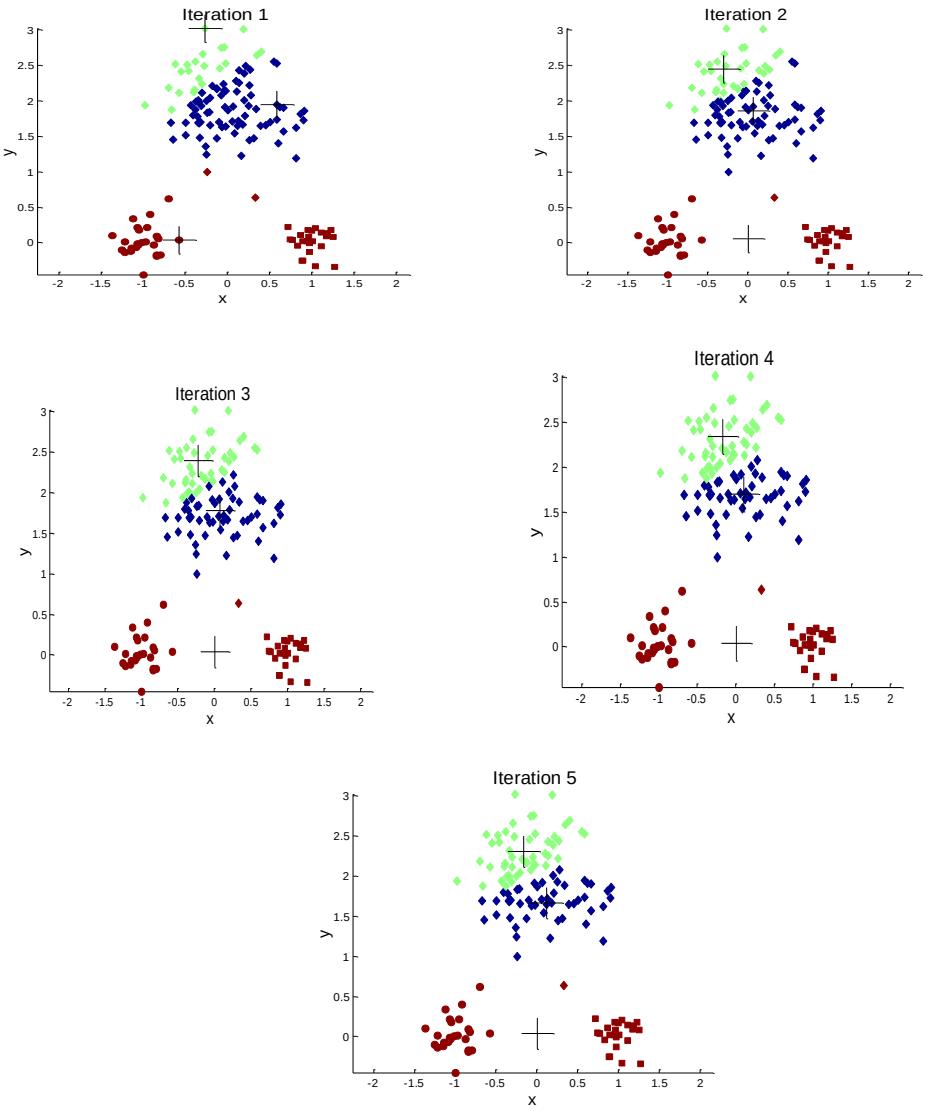

Fig 6: Initial centroids leading to poor clusters

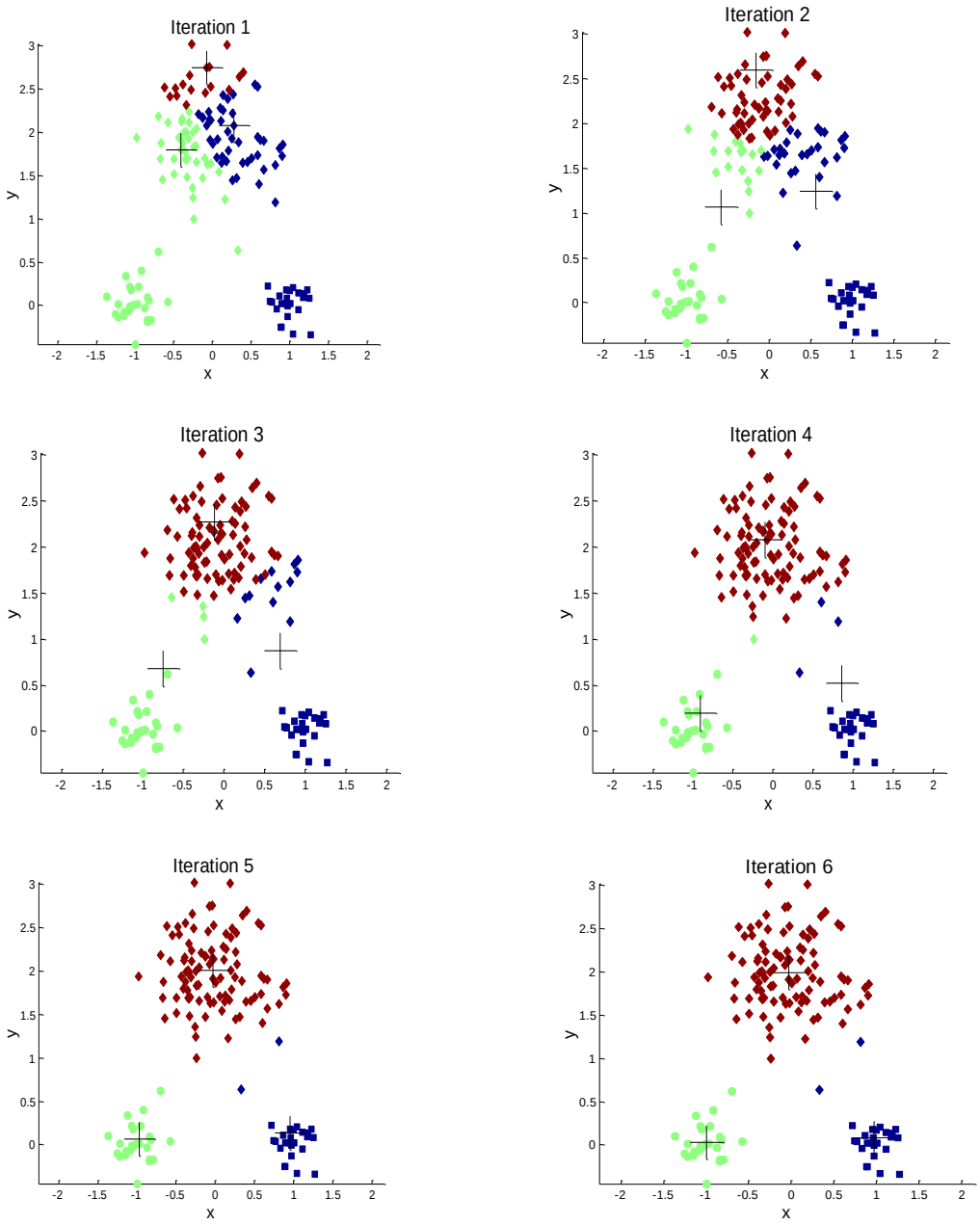

Fig 7: Initial centroids leading to better clusters

In the above figure 7, the initial centroids are changed which produce different results and as we can see here the clusters formed are better and acceptable.

3.2 A Numerical Example of K-Means Algorithm

This example below is taken from Kardi's tutorial [11] explains manual calculations showing how K-Means clustering algorithm works. Here we have four different objects which in this case are medicines and we need to group them into 2 clusters (k=2). There are 2 attributes here in our example weighted index and pH value as shown in the table below.

Object	Weight index	pH value
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

Table 1: Medicine objects with both the attributes

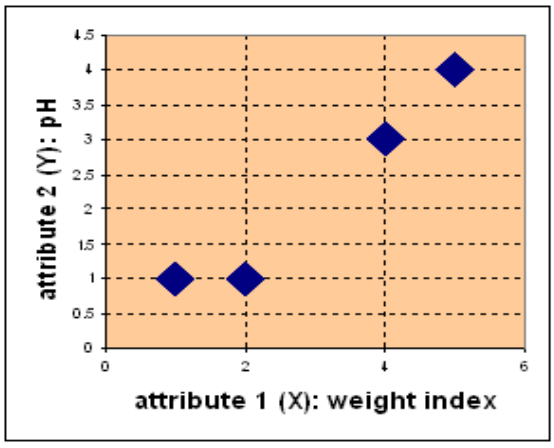

Figure 8: Points plotted representing medicine objects

The above figure shows how objects are plotted graphically in a space with two attributes represented on x and y axis.

After representing we start with assuming initial centroids namely c1 and c2. Here let first two points as centroid. Therefore c_1 = (1, 1) and c_2 = (2, 1) are initial centroids here and are plotted graphically as shown below:

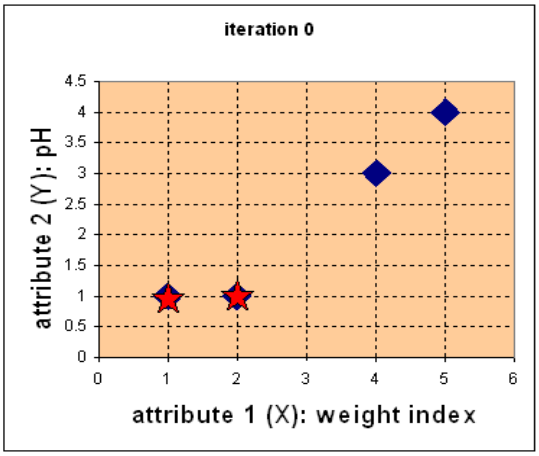

Figure 9: Plotting initial centroids for medicine objects (represented as red stars).

We then start finding distance between each data object and centroid. Here we use Euclidean distance for our example. After first iteration we form a distance matrix with following distances.

Distances	C1	C2
A	0	1
B	1	0
C	3.61	2.83
D	5	4.24

Table 2: Showing Distances between Data points and Centroids C1 & C2

The distance between medicine A and c_1 is 0 [Sqrt $((1-1)^2 + (1-1)^2)$].

The distance between medicine B and c_1 is 1 [Sqrt $((2-1)^2 + (1-1)^2)$].

The distance between medicine C and c_1 is 3.61 [Sqrt $((4-1)^2 + (3-1)^2)$].

The distance between medicine D and c_1 is 5 [Sqrt $((5-1)^2 + (4-1)^2)$].

Similarly distances between data points and centroid c2 are calculated to form a distance matrix represented below

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

We then form a group matrix G^0 showing how objects move into particular cluster by choosing the object with the minimum distance from centroid. In this case, object A moves to group-1 and other 3 objects move to group-2.

$$G^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix}$$
$$A \quad B \quad C \quad D$$

Recalculate the centroids for each cluster again by finding the average of the coordinate objects in that cluster like c_1 remains the same (1,1) being only point in that cluster and c_2 will be ((2+4+5)/3,(1+3+4)/3) = (11/3,8/3). This is plotted as shown in figure below.

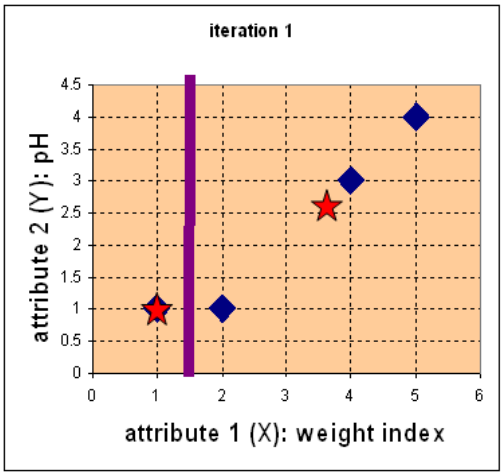

Figure 10: Recalculating centroids after first iteration

Calculate the distance of the objects to the new centroids to form a new distance matrix

$$\mathbf{D}^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

As we can see from above distance matrix after a group matrix A and B belong to group-1 and C and D belong to group-2

$$\mathbf{G}^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Recalculate centroids based on the new cluster objects formed as

$$\mathbf{c}_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\tfrac{1}{2}, 1)$$

$$\mathbf{c}_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\tfrac{1}{2}, 3\tfrac{1}{2})$$

The points along with new centroids are plotted as shown below

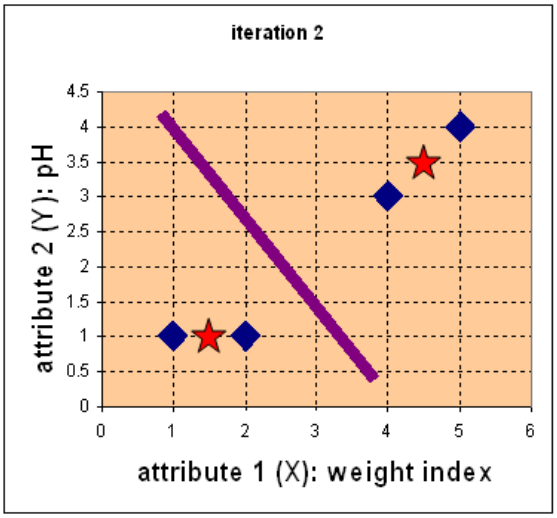

Figure 11: New centroids and cluster points after Iteration 2

Distance matrix is again calculated with new centroids formed to the data points

$$\mathbf{D}^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1\frac{1}{2}, 1) & group-1 \\ \mathbf{c}_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

Again we form a group matrix showing that Medicine A and B fall into group 1 and Medicine C and D fall into group 2.

$$\mathbf{G}^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Now here, the group matrix formed has $\mathbf{G}^2 = \mathbf{G}^1$ the algorithm stops resulting in the final clusters with data points as shown in table below:

Object	Weight index	pH value	Cluster No.
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

Table 3: Final clusters formed with data points

3.3 Distance Metrics

Distance function or Metrics is defined as the distance between elements in a space. The performance of many learning and data mining algorithms depend on choosing a good metric over input data. A distance metrics as said in [14], d (X, Y) is a function or algorithm for calculating a distance between two things, X and Y having following properties:

1. It is always positive or zero. $d_{ij} \geq 0$
2. The distance from a document to itself is zero. $D(x, y) = 0$ iff $x = y$
3. It obeys inequality property of a triangle. For any three points X, Y, and Z,

   $D(x, y) + D(y, z) \geq D(x, z)$ for any Y.
4. Similarity axiom $D(x, y) = D(y, x)$

Anything that obeys these 3 properties is a distance metric. Most commonly used distance metric is Euclidean distance given by

$$d_{Euclid}(A, B) = \left((A_x - B_x)^2 + (A_y - B_y)^2\right)^{1/2},$$

Distance measures dissimilarity. Similarity is quantity that reflects the strength of relationship between two objects whereas dissimilarity measures the discrepancy or disorderness between two objects.

3.3.1`Different Distance Functions Used

K-means uses an iterative algorithm that minimizes the sum of distances from each object to its cluster centroid. For the thesis, we have used six different distance functions. These distance functions were chosen from different references available in the link [13] and [16]. As the size of the data set increases with number of attributes it becomes more difficult for the vector matrix to provide better results using K-means.

Here is a brief overview of distance metrics used in the thesis along with some explanation about other distance metrics:

1) Bray-Curtis distance: Braycurtis(u,v) distance between two vectors u and v , is defined as

d (u, v) = $\sum |u_i - v_i| / \sum |u_i + v_i|.$

Where u and v are n-dimensional vectors.

Bray-Curtis distance between two vectors:

BrayCurtisDistance [{a, b, c}, {x, y, z}] then distance is given by (Abs [a-x] +Abs [b-y] +Abs[c-z]) / (Abs [a+x] +Abs [b+y] +Abs[c+z])

BrayCurtisDistance [{1, 2, 3}, {2, 4, 6}] is 1/3

2) Canberra distance: Canberra(u,v) distance between two vectors u and v , is defined as

$$d(u, v) = \frac{\sum_i |u_i - v_i|}{\sum_i |u_i| + |v_i|}.$$

Where u and v are n-dimensional vectors.

Canberra distance between two vectors:

CanberraDistance [{a, b, c},{x,y,z}] then distance is given by (Abs[a-x]/(Abs[a]+Abs[x])) + (Abs[b-y]/(Abs[b] + Abs[y])) + (Abs[c-z]/(Abs[c] + Abs[z]))

Canberra distance between 2 numeric vectors [{1, 2, 3}, {2, 4, 6}] is 1.

3) Euclidean distance: Euclidean distance examines the root of square differences between the coordinates of a pair of objects. For vectors i and j distance d (i, j) is given by

$$d_{ij} = \sqrt{\sum_{k=1}^{n} \left(x_{ik} - x_{jk}\right)^2}$$

Where i and j are n-dimensional vectors.

Euclidean Distance between vectors [{a, b, c}, {x, y, z}] is given by Sqrt($Abs[a-x]^2 + Abs[b-y]^2 + Abs[c-z]^2$)

Euclidean distance between [{1, 2, 3}, {2, 4, 6}] is Sqrt (14).

4) Cosine distance: The most popular distance metrics for text clustering which normalizes the features of a covariance matrix.

The cosine of the angle is calculated using the formula shown below.

$$\cos \Theta = \frac{\sum_{i=1}^{N} x_i y_i}{\sqrt{\sum_{i=1}^{N} x_i^2} * \sqrt{\sum_{i=1}^{N} y_i^2}}$$

where θ refers to the angle between the point and the centroid

x refers to the point

y refers to the centroid

N refers to the dimension of the vector

Cosine distance between two vectors:

Cosine Distance [{a, b, c}, {x, y, z}] is given by 1 – ((ax + by + cz)/((Sqrt(Abs[a]^2 + Abs[b]^2 + Abs[c]^2)* (Sqrt(Abs[x]^2 + Abs[y]^2 + Abs[z]^2)))

Cosine Distance [{1, 2, 3},{3, 5, 7}] is 1- (17 * Sqrt(2/581))

This ratio defines the cosine angle between the vectors, with values between 0 and 1. The expressions cosine similarity, Sim(A, B), or COSIM are commonly used.

$$\text{Sim(A, B)} = \text{cosine } \theta = \frac{A \bullet B}{|A||B|} = \frac{x1^*x2 + y1^*y2}{(x1^2 + y1^2)^{1/2} \ (x2^2 + y2^2)^{1/2}}$$

As the angle between vectors lessens the Cosine angle approaches to1 i.e when angle becomes 0 it will be 1.

This way we can sort the document vectors by ranking by measuring the closeness of vectors to a query vector.

To do this we use the concept of finding term frequency and Document frequency for a given collection of documents that has to be

queried. These terms are extracted from the collection of documents to be queried.

The point coordinates of term weights are given by term frequencies.

$$\mathrm{Sim}(Q, D_i) = \frac{\sum_i w_{Q,j} w_{i,j}}{\sqrt{\sum_j w_{Q,j}^2} \sqrt{\sum_i w_{i,j}^2}}$$

where Q is a query, D is a document relevant to Q and w are weights.

If max is maximum term frequency in a document, N as number of documents in a collection and n as number of documents containing a query term, we can redefine term weights defined by Dr.Garcia in [18] as,

1. w = tf/tfmax
2. w = IDF = log(N/n)
3. w = tf*IDF = tf*log(N/n)
4. w = tf*IDF = tf*log((N - n)/n)

5) Variational distance: The variational distance metric is a measure used to quantify the difference between probability distributions given by

$$V(P||Q) = \sum_{i=1}^{n} |p_i - q_i|.$$

6) Chi-Square distance: The distance between Q and V for Chi-Square distance is given by

$$\chi^2(Q, V) = \sum_i \frac{(q_i - v_i)^2}{q_i + v_i}$$

7) Trigonometric distance: The distance between vectors A and B for trigonometric distance is given by

Dist (A,B) = Sqrt (2 * (S-A)(S-B))

where S is the average of vectors A and B

Different similarity functions produce different clusters for K-means depending on the size of the vectors and the data used. In the thesis, we modify K-means algorithm with above discussed metrics and form clusters and also calculate which metrics works faster. As the data set gets large, the metrics with a dot product in the function does not give good results because of clashes for 0's in finding term weights. In information retrieval applications sometimes, the ratio is calculated to normalize the length of documents since long documents tend to have large term frequencies [4]. We compare different distance functions mentioned above. Similarity function play a major role in information processing tasks to rank items in the data base based according to their similarity to some query. The Quality of the similarity directly determines the quality of clusters formed.

3.4 Time Complexity of K-means using different Metrics

K- Means clustering is easy to implement with different Metrics. Though the time taken for different distance metrics vary. In general, K-means take moderate amount of time complexity. Let t_{dist} be the time

taken to calculate distance between two data objects. Each Iteration has a time complexity of $O(K*n * t_{dist})$

K = number of clusters (centroids)

n = number of objects

If I is the total number of iterations bound then time complexity is given by $O(I*K*n * t_{dist})$.

For m-dimensional Vectors, time complexity will be O(I * K * n * m) where m is large and centroids are not sparse. In the thesis, we perform tests over different distance metrics for K-means clustering over same data set and compare time required for different metrics used based the number of iterations to run to form clusters. The execution time for different distance metrics varies for K-means which will be discussed in detail in chapter 5.

Space Complexity:

For a given vector model, storing points and centroids the space complexity of K-means is given by O ((n + K) m).

CHAPTER 4

IMPLEMENTATION

4.1. Documents Collection

In the thesis, we concentrate on clustering electronic documents into groups containing similar documents together, based on the clusters formed. We apply K-Means over the documents after preprocessing. The implementation of K-means is written in java for different similarity metrics that gives different clustering results to analyze the metrics used and execution time.

To perform K-means, the document collection we use is obtained from "Reuters-21578, Distribution 1.0 test collection". There are 21578 newswire stories classified into several sets of categories by personnel from Reuters Ltd. and Carnegie Group, Inc in 1987 and were further formatted by David D. Lewis and Peter Shoemaker in 1991. There are total 674 categories in Reuters-21578 collection as shown in Table 4 below [17]

Field	Categories
Topics	135
Organizations	56
Exchanges	39
Places	176
People	269

Table 4: Reuters-21578 collection categories

In the thesis, we concentrate on Topics field set for our research and choose 5 categories out of 135 available in the set. They are

1. Acquisition,
2. Grain,
3. Interest Rate,
4. Jobs and
5. Trade

In this five categories there are total of 504 documents mapped from the collection. These 504 documents are further divided into two sets, Training set consisting of 304 and Test set with 200 documents. For the thesis work, training set collection is used to form clusters of documents into categories. The training set collection with 304 documents divided into 5 categories is shown in table 5 below:

Category	Total number of documents
Acquisition	70
Grain	60
Interest Rate	70
Jobs	34
Trade	70

Table 5: Training set collection

K-means algorithm takes n-dimensional vector points as inputs and returns clusters formed into different categories. Documents used in this Reuters collection are in "Standard Generalized Markup Language" format. In order to pass these documents to K-means, these documents need to be preprocessed and converted into a suitable format that is fed as input to the algorithm. Below is a screenshot of a SGML document from the Reuter 21578 collection of category trade.

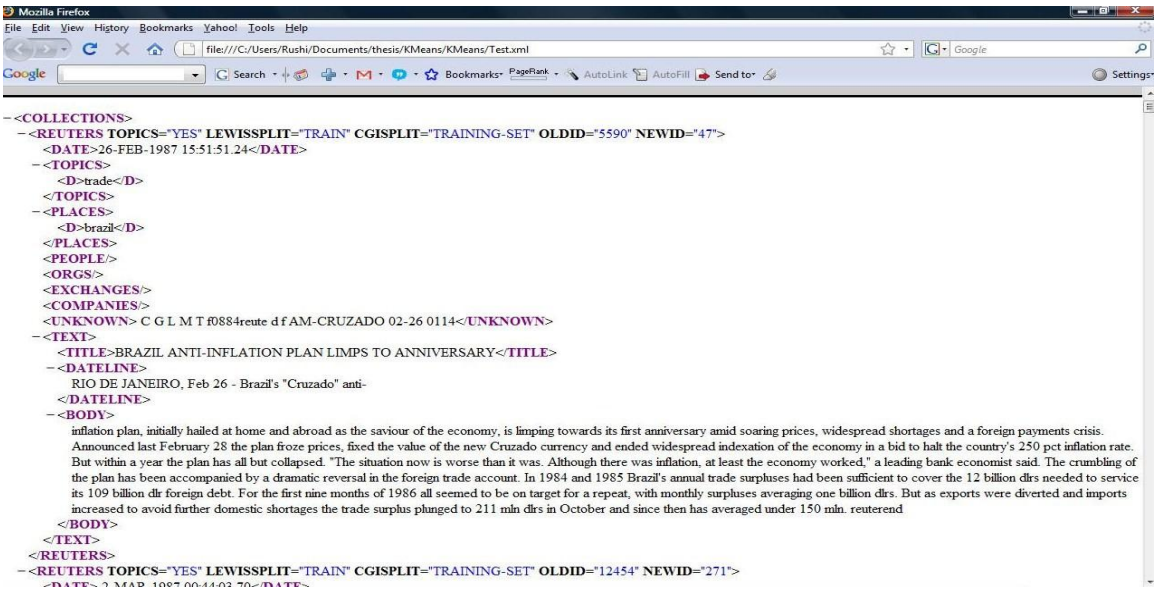

Figure 12: A screenshot of Reuters 21578 collection

A clear description of tags is given in [17]. Every document in the collection starts with a Reuter tag and ends with a Reuters tag. The topics tag indicates the document category manually categorized by experts. Body of the tag contains the whole story or article which ends with a Reuterend statement. This XML document needs to be converted into format to pass as input to K-means which is done by following preprocessing steps.

4.2. Documents Preprocessing

1) Parsing the XML document

All the markup tags are removed to parse the documents using a parser [19] to take the information inside the body tag into a new file.

The XML document after parsing looks as shown below. This is parsed document for the above shown XML document:

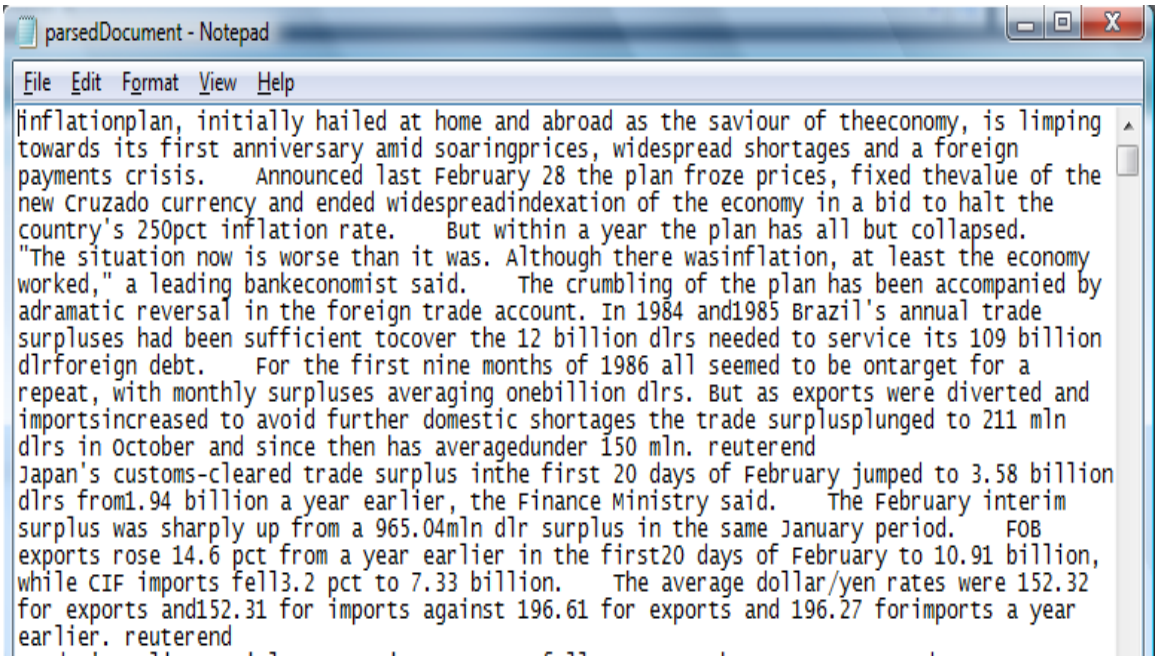

Figure 13: Parsed XML document

2) Tokenization

The text corpus as seen in the screenshot above after parsing is cumbersome and has to be tokenized. Tokenization is the process of breaking parsed document text into chunks, called tokens [20]. This process includes removing the punctuations and the text is lowercased. The above parsed document is tokenized to form a list of tokens as shown in figure below.

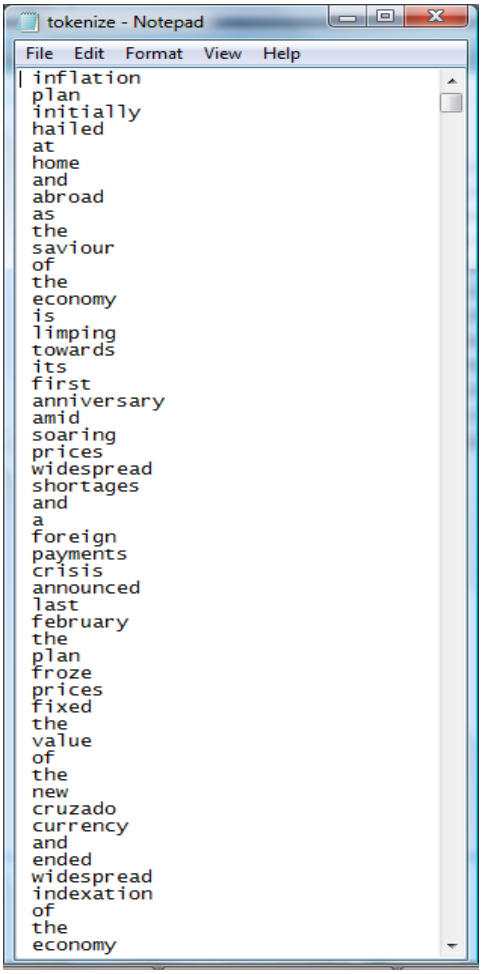

Figure 14: List of tokens after tokenization

3) Stop words Removal

Next after tokenization, it is needed to remove stop words from the list of words. Stop words like is, are, with, the, from, to etc that occur in almost every document are be removed to proceed further which doesn't provide any use to for weighted index being so common. The list of stop words used in the thesis is 416 as shown in the figure below that is a part of the code.

Before removing stop words total number of terms in 304 documents = 52034

```
String[] STOP_WORDS =
{
        "about","able","above","according","accordingly",
    "across","actually", "after","afterwards","again","against", "all",
    "allow","allows","also","almost","alone","along","already","although","always","am","among",
    "amongst", "an", "and",
    "another", "any","anybody","anyhow","anyone","anything","anyways","anywhere",
    "apart","appear","appreciate","appropriate", "are","around", "as","aside","ask","asking",
    "associated", "at","available","away","awfully", "be","became",
    "because", "become","becomes","becoming","been", "before","beforehand","behind", "being", "believe",
    "below","beside","besides","best","better","between","beyond",
    "both", "brief","but", "by", "came", "can","cannot","cant","cause","causes","certain","certainly","changes","clearly", "come",
    "concerning","consequently","consider","considering","contain","containing","contains","corresponding",
    "could","course","currently","definitely","described","despite", "did","different", "do", "does","doing","done",
    "down","downwards","during", "each","edu","eg","eight", "else","elsewhere","enough","entirely","especially","et","etc",
    "even","ever","every","everybody","everyone","everything","everywhere","ex","exactly","example","except","far","few",
    "fifth","first","followed","following","follows","for","former","formely","forth","four", "from",
    "further","furthermore", "get","gets","getting","given","gives","go","goes","going","gone", "got","gotten","greetings",
     "has", "had","happens","having","hello","help","hence","hereafter","hereby","herein","hereupon","hers","herself",
    "he", "have", "her", "here", "him", "himself",
    "his","hither","hopefully", "how","howbeit","however","if","ignored", "in","inasmuch","inc","indeed", "into", "is", "it",
    "indicate","indicated","inner","insofar","instead","inward","its", "just","keep","keeps","kept","know",
    "knows","known","last","lately","later","latter","latterly","least","less","lest","let", "like", "liked",
    "likely","little","look","looking","looks","ltd","mainly","may","maybe","make", "many", "me","mean","meanwhile","merely",
    "might", "more","moreover", "most","mostly", "much", "must", "my","myself","name","namely","nd","near","nearly","necessary",
    "need","needs","neither","never","nevertheless","new","next","nine","no","nobody","non","none", "now","normally","novel",
    "nowhere","nor","not","obviously","often","oh","ok","okay","one","old","otherwise","once","ones","ought","ours","outside","overall
     "of", "on", "only", "or","others","own",
    "other", "our", "out", "over", "particular","particularly","per","perhaps","placed","please","plus","possible",
    "presumably","probably","provides","quite","re","rather","really","regards","respectively","right", "said","saw",
    "same", "see", "should", "since", "so", "some","seeing","self","second","say","saying","sensible","six","should","somebody",
    "something","sometimes","somewhat","somewhere","soon","still","specifying","such","sup","sure",
    "still", "such", "take", "than", "that", "the","tell","thanks","the","thereafter","therefore","towards","too","twice",
    "their", "them", "then", "there", "these",
    "they", "this", "those", "through", "to", "too",
    "under", "up", "use","useful","uses","using","usually","value","via","viz", "very", "want", "was",
    "way", "we", "well", "were", "what", "when","whom","why",
    "where", "which", "while", "who", "will",
    "with", "would", "you", "your","yes","yet",
    "a", "b", "c", "d", "e", "f", "g", "h", "i",
    "j", "k", "l", "m", "n", "o", "p", "q", "r",
    "s", "t", "u", "v", "w", "x", "y", "z","$"
};
```

Figure 15: A Screenshot of List of stop words

Total number of words appear to be of less interest in order to save time and space = 24124.

Finally, after removing stop words left over number of terms = 27910. So we further move to stemming after removing stop words.

4) Stemming

Stemming refers to the process of reducing terms to their stems or root variants. For example agreed-> agree; meetings, meeting -> meet; engineering, engineered, engineer -> engine etc. Stemming reduces the computing time as different form of words is stemmed to form a single word. The most popular stemmer in English is Martin Porter's Stemming Algorithm as shown to be effective in many cases in [19]. For this thesis, we use java as a programming language to implement stemming algorithm.

Figure 16: A Screenshot of output after stemming

5) Building Inverted Index

Indexing is nothing but refinement i.e. a sufficient general description of a document such that it can be retrieved with a query that contains the same subject as the document and vice versa. Indexing is a mechanism to locate a given query term in a document [23]. Inverted file contains an inverted file entry that stores a list of pointers to all occurrences of that term in the main test for every term in the lexicon, where each pointer is, in effect, the number of a document in which the term appears. There are two types of inverted index. A record level inverted index consists of a list of references to documents for each term. An example taken from [23] of how inverted index works is show in table below. Consider the traditional children's nursery rhyme in table

Document	**Text**
1	Peace porridge hot, peace porridge cold,
2	Peace porridge in the pot,
3	Nine days old,
4	Some like it hot, some like it cold
5	Some like it in the pot
6	Nine days old.

Table 6: Example text; each line is one document

The inverted index generated for this text is show in table 7 without stemming or removing stop words.

Number	Term	Documents
1	cold	1,4
2	days	3,6
3	hot	1,4
4	in	2,5
5	it	4,5
6	like	4,5
7	nine	3,6
8	old	3,6
9	peace	1,2
10	porridge	1,2
11	pot	2,5
12	some	4,5
13	the	2,5

Table 7: Inverted file for text in table 6

The removal of stop words and stemming results in reduction of terms for indexing favoring query processing to run faster. In the thesis, as shown above in figure 16 we apply inverted index to obtain inverted

index table with the terms and the document number. But for further processing we need significant terms that are obtained from dimensionality reduction. This is a major difficulty in text categorization of feature space i.e. total number of terms considered. Even a moderate size collection consists of thousands of unique terms [24]. So we need to reduce the number of terms in the collection which is done by dimensionality reduction. Out of many methods known, for the thesis we perform document frequency thresholding. This is the simplest technique used for reducing vocabulary in the collection. Predefined threshold value is assigned such that only those terms from the collection that are in the given range are used. As it also depends on the vector formed in the next stage to find term and document frequency. So for the thesis we have defined the document frequency range to be greater than 25 and less than 65. Range less than 25 results in the vectors which doesn't produce efficient clusters and above 65 results in words that are too common for all documents.

Out of 3608 terms after stemming, for the given range for inverted index we get just 123 terms. Once significant terms are obtained, the next step is to find the term frequency and document frequency in order to form vectors for processing K- means algorithm.

4.3. TF * IDF Calculation

Term Frequency and Inverse Document Frequency is a weight often used in text mining and information retrieval. It is a measure of how important a word is to a document in a collection [25]. Term Frequency is defined as the total count of word that is repeated in a document. Inverse Document Frequency is defined as the total number of times the word occurs in the entire documents i.e. number of documents containing the significant word. Thus the term frequency is given by

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

Where $n_{i,\ j}$ is the number of times the significant term t_i occurs in document d_j and the denominator is the sum number of times all the terms occur in document d_j

The inverse document frequency is obtained by dividing the number of documents by the number of documents containing the term, and then the logarithm of that quotient given by

$$\mathrm{idf}_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

Here, |D| is the total number of documents in the corpus

$|\{d : t_i \in d\}|$ is the number of documents where the term t_i appears (that is $n_{i,j}$ is not equal to 0. If the term is not in the corpus, this will lead

to a division by zero. Therefore it is common to use $1+|\{d:t_i \in d\}|$. Then we define TF-IDF given by

$$(\text{tf-idf})_{i,j} = \text{tf}_{i,j} \times \text{idf}_i$$

In the thesis, we have considered 123 significant terms after dimensionality reduction to find term frequency which is shown in screenshot below

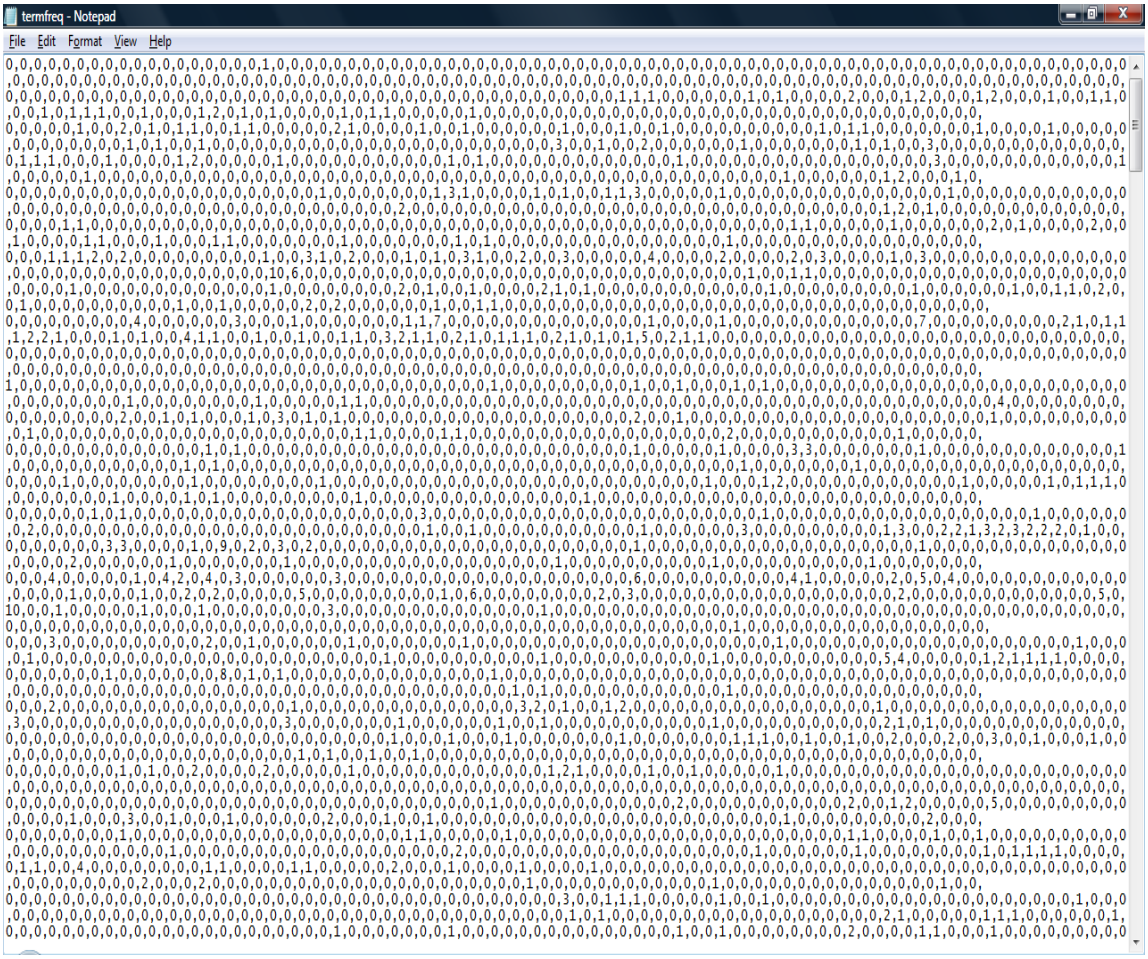

Figure 17: A Screenshot of Term frequency matrix

Next step is to find Document frequency for 123 terms in 304 documents using the equation shown above, followed by TF * IDF which is shown in screenshot below a matrix of size 304 X 123 representing a vector space model formed by 304 documents that given as input to K-means where each row represents vector or document and 123 columns show the dimensions of that vector. The screenshot below shows the matrix formed from TF * IDF calculation.

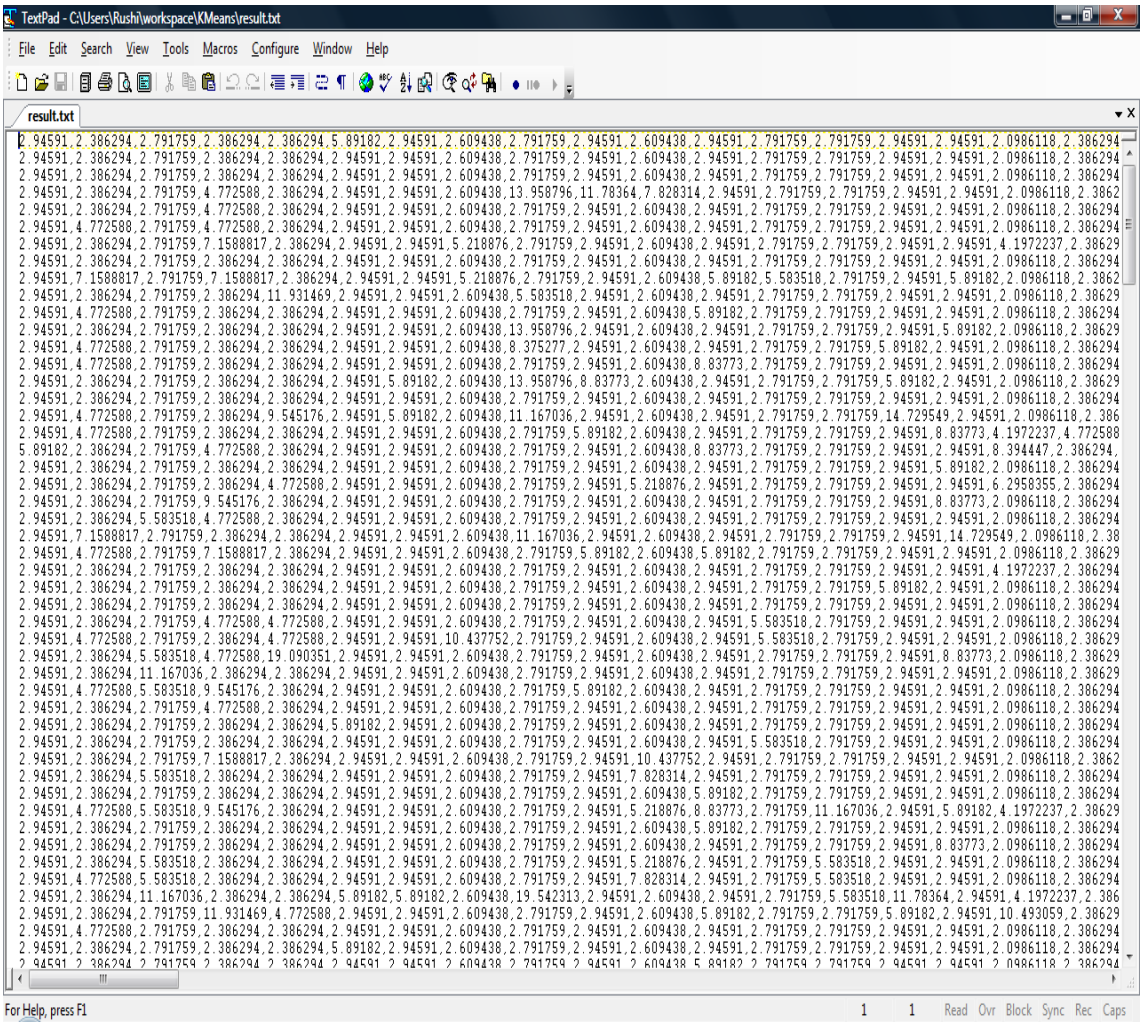

Figure 18: A Screenshot of TF * IDF matrix

4.4. Algorithm Implementation

The vector space model showed above results in n-dimensional vectors with each row representing a vector/document and each column representing a term in the corpus. . This vector is fed as input to K-means algorithm to form clusters. K-means is further modified with different distance metric to form clusters and to find execution time for metrics used. As discussed in chapter 2, K-means algorithm is implemented over the result matrix where each document has a weighted term value. These values can affect algorithm to give worse result if the significant terms produce similar weights. The factors as mentioned earlier that could be varied while implementing the algorithm to produce clusters as desired are:

1. Number of clusters
2. Number of iterations (not required when we compare group matrix so that it is same as previous iteration, only needed to find execution time).
3. The distance metric used for finding distance between the point and the cluster centroid (most important part of the thesis).

Based on above factors the numbers of clusters are varied to get desired results, in the thesis clusters were varied from 5 to 10 along with the distance metrics, here six different distance functions were implemented to find how clusters are formed and which metrics give best results also how the execution time varies for different metrics used. We

form clusters for 304 documents represented as vectors. Though K-means produced few similar results, the documents belonging to the same cluster did not belong to the same category. Some results produced clusters with many documents in the same category.

Algorithm was implemented in Java and hash tables as a data structure to store vectors representing documents, initializing random centroids stored as vectors. The centroid also were modified by using a equation to set initially which produced almost same results as the algorithm runs till the distance metrics is same as in previous iteration.

Below is a part of code to set the number of clusters by randomly selecting centroids

```
  for(int i = 0; i<clustNumber; i++)
{
int randomIndex = random.nextInt(rows);
        for(int j=0; j<cols; j++)
{
        centroid[i][j] = doc[randomIndex][j];
  }
}
```

Then calculate distance (Euclidean is shown below) between centroid and document as shown with a part of code below to produce a distance matrix

```
for(int k=0;k<clustNumber;k++)
{
        for(int i=0; i<rows; i++)
        {
                dotProduct=0;
                for(int j=0; j<cols; j++)
                {
            dotProduct+=Math.pow(centroid[k][j]-doc[i][j],2);
```

```
                }
                distance[k][i] = Math.sqrt(dotProduct);
            }
        }
```

Once distance matrix is obtained, we form a group matrix by based on the least distance of document from the centroid followed by clusters formed based on minimum distance.

```
for(int i=0; i<rows; i++)
    {
        flag=0;
        count2=0;
        minValue=distance[0][i];
        //group[0][i]=1;
        for(int k=0;k<clustNumber;k++)
        {
            if(distance[k][i]<minValue)
            {
                minValue = distance[k][i];
                flag=k;
            }
            count2++;
        }
        if(flag==0 && count2==clustNumber-1)
        {
            group[0][i]=1;
        }
        else
        {
            group[flag][i]=1;
        }
    }
```

Compare group matrix with the group matrix formed in the previous iteration so that algorithm stops if it is same. If the group matrix is not same, recalculate centroids with the code as shown below

```
for(int k=0;k<clustNumber;k++)
    {
        for(int j=0; j<cols; j++)
        {
            count=0;
```

```
                temp=0;
                for(int i=0; i<rows; i++)
                {
                        if(group[k][i]==1)
                        {
                                count++;
                                temp+=doc[i][j];

                        }
                }
                if(count!=0){
                        centroid[k][j]=temp/count;
                }
        }
}
```

Thus, all these steps are implemented in order to form clusters. The results were very similar as in previous case with most of documents into the same cluster and most of other clusters contained documents completely not related to the categories. K-means is implemented again modifying other distance metrics and finding the execution time the algorithm takes as discussed in the next section.

4.4.1. Experiments over different Metrics

In the thesis, K-means clustering is implemented over different distance metric. Apart from commonly used Euclidean and Cosine distance functions we used some new distance functions like Chi-Square, Canberra, Variational etc here. The result of using different functions varies in forming clusters of different sizes i.e. with different documents in cluster. Many of the documents of one category may sometimes move to other cluster due to different metrics. Here we have 304 documents with 123 columns. Few distance metrics like Jaccard

similarity, Hellinger's distance and Harmonic mean when tried to implement over the document set produced worst results because of the weight of the term in the document becoming 0. This causes many of the rows containing 0's in most of the columns. This could be reason why most of the documents are grouped into one cluster though for other distance metrics the results were quite good when the number of significant terms were changed by dimensionality reduction. The different distance functions used are discussed below:

1) Euclidean Distance

As this is the most commonly used distance measures the implementation of Euclidean was quite simple as shown with a small block of distance code below:

```
dotProduct+=Math.pow(centroid[k][j]-doc[i][j],2);
distance[k][i] = Math.sqrt(dotProduct);
```

The number of cluster is chosen initially is defined as clustNumber and then the distance is calculated as mean square root of difference between the centroid to each document in the corpus.

2) Variational Distance

It is the absolute difference between the between the centroid and the documents. It is simple to implement and results obtained are quite better for even large number of significant terms after indexing. The block of code that calculates variational distance in java is shown below:

```
dotProduct+=Math.abs(centroid[k][j]-doc[i][j]);
distance[k][i] = dotProduct;
```

3) Canberra Distance

It is calculated as the division of difference between absolute centroid value and each document's absolute value to the sum of absolute centroid value and each document's absolute value in the set. The block of code for calculating Canberra's distance is shown below:

```
numerator +=(Math.abs(centroid[k][j]))- (Math.abs(doc[i][j]));
denominator += (Math.abs(centroid[k][j])) + (Math.abs(doc[i][j]));
temp3 = numerator/ denominator;
distance[k][i] = temp3;
```

4) Bray-Curtis Distance

It is calculated as the division of absolute difference between centroid and each document and absolute sum of centroid and each document in the set. The block of code for Bray - Curtis is shown below:

```
numerator+=Math.abs(centroid[k][j]-doc[i][j]);
denominator+=Math.abs(centroid[k][j]+doc[i][j]);
temp3 = numerator/ denominator;
distance[k][i] = temp3;
```

5) Chi-Square Distance

To find distance from centroid to each document in the corpus where the numerator is calculated same as Euclidean distance divided

by the absolute sum of centroid and document vector. This distance produced the better results compared to others for forming clusters with more execution time. The block of code in java to calculate Chi-Square distance is shown below.

```
numerator +=Math.pow(centroid[k][j]-doc[i][j],2);
denominator += (Math.abs(centroid[k][j])) + (Math.abs(doc[i][j]));
temp3 = numerator/denominator;
distance[k][i] = temp3;
```

6) Trigonometric measure

This method is commonly used to find distance between sides of a triangle while finding distance between two objects in trigonometry. First average of centroid and document vector is calculated and then square root of absolute distance of product of average, difference of average and centroid and difference of average and document vector is calculated. This calculation is a bit complicated and is implemented as shown below.

```
avg=(centroid[k][j]+doc[i][j])/2;
temp2+=Math.sqrt(Math.abs(2*((avg*(avg-centroid[k][j]))-
                                    (avg*(avg- doc[i][j])))));
distance[k][i] = temp2;
```

All the above mentioned distance functions form clusters with different documents in different categories from one another. The results

obtained after clustering and analysis over the results for different similarity metrics used will be discussed in chapter 6 in the thesis. Execution time as discussed in the implementation above is also calculated for all the distance functions. Though the clusters are obtained from K-means, there are certain limitations for the algorithm that will also be discussed in the next chapter.

CHAPTER 5

RESULTS EVALUATION

K-means clustering is though simple to implement but results are strongly affected by weighting and document length. Cluster size and documents in the cluster varies with number of iterations, cluster centroids and distance metrics used. As discussed in previous chapter, K–means implementations produce different results for different distance functions which we concentrate more on. This chapter is divided in two sections. Initial discussion is based on clusters and comparison of clusters formed for different distance functions and in the other part we discuss time complexities and execution time taken for different functions.

5.1. Comparison Based on Clusters

Clusters formed from K-means are discussed based on the distance functions. As we have 304 training set documents categorized in 5 different categories we perform clustering specifying the number of

clusters. During implementation, as discussed above there is no specific number of iterations provided and so the algorithm stops when there isno change in the clusters set in the current iteration and the previous one. Thus number of clusters is varied to get effective results as shown for different metrics below.

1) Canberra Distance

The table below provides the results obtained for Canberra distance for 7 clusters are shown below. In cluster 0 there are 71 of which most of documents belong equally to trade, interest and jobs category where as cluster 1 has 59 documents of which 80 % belong to trade category.

Cluster Number	**No of Docs**
Cluster 0	71
Cluster 1	59
Cluster 2	58
Cluster 3	43
Cluster 4	29
Cluster 5	38
Cluster 6	6

Table 8: No of documents in each cluster for Canberra's distance

Cluster 2 has 58 documents and 70 % belong to grain category. In cluster 3, there are 43 documents of which 20 % are from trade and 80% from interest category. Cluster 4 contains 29 of which 80 % belong interest and cluster 5 has 38 documents of which half of them belong to acquisition whereas other half belong to jobs and grain category. In the final cluster 6 just 6 documents are grouped belong to jobs category. The results obtained here are to some extent better and as it has bit division in the distance functions, the results get affected.

2) Bray-Curtis Distance:

Bray- Curtis Distance has most of calculation like Canberra and has similar implementation code but as we find absolute values later the division effects lot in the results. These results here show that cluster 0 has 43 documents out of which 70% belong to grain.

Cluster Number	**No of Docs**
Cluster 0	43
Cluster 1	39
Cluster 2	13
Cluster 3	134
Cluster 4	60
Cluster 5	14

Table 9: Documents in each cluster for Bray-Curtis distance

In cluster 1, out of 39 documents 60 % belong to jobs category and few documents from trade and grain category got moved in it. In cluster 2, there are only few 13 documents from different categories whereas in cluster 3 out of 134 documents equal number of documents from trade, interest and jobs category. Cluster 4 contains 60 documents out of which almost all around 80% below to acquisition and few from job category whereas cluster 5 has very few documents from different categories.

3) Variational Distance

Variational distance is simplest metric for implementation but the results produced are not effective in this case for the document set used in the thesis.

Cluster Number	**No of Docs**
Cluster 0	82
Cluster 1	37
Cluster 2	17
Cluster 3	33
Cluster 4	01
Cluster 5	135
Cluster 6	39

Table 10: Documents in each cluster for Variational

Here in cluster 0 out of 82 documents, most of them are mixed from different categories where are cluster 4 has just 1 document and cluster 5 also has documents from all categories which is not a good result. But coming to cluster 1 most of documents are form trade whereas cluster 2 has more from grain. In cluster 3, 33 documents are grouped from jobs category where as cluster 6 has 39 documents numbering most between 200-234 showing they belong to acquisition in the document collection.

4) Chi-Square Distance

Chi-Square distance is said to be a combination of Euclidean in the numerator and absolute difference in the denominator as discussed in chapter 5 in implementation producing effective results. Here there are 5 clusters of which 62 documents belong to cluster 1 of which 80%

Cluster Number	No of Docs
Cluster 0	62
Cluster 1	47
Cluster 2	44
Cluster 3	54
Cluster 4	97

Table 11: Documents in each cluster for Chi-Square

belong to trade and 15 % documents from jobs category whereas cluster 1 has 15 % from trade and 80 % documents form jobs category. In cluster 2, out of 44 documents 90 % of them belong to grain category. In cluster 3 there are 54 documents of which 70 % belong to acquisitions and 30 % of jobs category. Cluster 4 has 97 % of which 70 % from interest and few from jobs and acquisition group.

The results here form cluster effectively which most of documents going into appropriate category as needed. Thus these results are better compared to all other distance metrics cluster outputs.

5) Trigonometric Distance

Trigonometric distance is complicated to implement and has few multiplications involved which results in getting many 0 in the outputs making it unable to move documents in different clusters. The table below shows the results obtained from this metrics for 9 clusters.

Cluster Number	No of Docs
Cluster 0	3
Cluster 1	2
Cluster 2	20
Cluster 3	13
Cluster 4	2
Cluster 5	1
Cluster 6	3
Cluster 7	1
Cluster 8	259

Table 12: Documents in each cluster for Trigonometric

Most of clusters have very few documents and as seen above many documents move in cluster 8 around 259 which completely is worst. Even after changing number of clusters during implementation the clusters formed were never effective using this metrics.

6) Euclidean Distance

Euclidean distance is most commonly used metric for k-means clustering. In this thesis for the document collection used the following table shows 9 clusters formed.

Cluster Number	No of Docs
Cluster 0	33
Cluster 1	45
Cluster 2	29
Cluster 3	25
Cluster 4	42
Cluster 5	41
Cluster 6	22
Cluster 7	32
Cluster 8	35

Table 13: No of documents in each cluster for Euclidean

In cluster 0 there are 33 documents of which many documents numbering 70-130 almost 80 % belonging to category grain are present, cluster 1 has 45 documents with 80 % related to jobs category and few from trade got mixed up. Cluster 2 has 29 documents mixed from all categories and cluster 3 has 25 documents half of which are from trade and few from grain and interest. Cluster 4 has 42 documents of which almost all belonging to trade whereas cluster 5 has 41 and cluster 6 has 22 documents both consisting of 90 % from interest category. Cluster 7 has 32 documents of which 70 % are from acquisitions and cluster 8 has 35 documents of which 70 % are from jobs category.

As seen from cluster results for Euclidean distance, it is equally efficient metric for K-means as Chi-Square and Canberra distance. From the above distance metrics results to form clusters the following analysis and comparison between different metrics are made:

1) Chi-Square distance produced the best results followed by Canberra and Euclidean distance metrics.

2) The results largely got affected with the weighted term matrix and the calculation involved in the distance metrics.

3) Trigonometric and variational distances didn't provide good results at all due to weighted matrix values containing large number of 0's.

4) Bray-Curtis gave fine results but not as good as Canberra, Euclidean or Chi-Square.

5) Number of centroids or clusters defined at each implementation for different metrics changed the results showing that deciding the number of clusters in the beginning plays a key role for algorithm implementation.

Apart from above results, execution time also needs to be considered for efficient implementation which will be discussed for different metrics in the next section.

5.2. Comparison based on Execution time

As seen in the previous section, the comparison for distance metrics based on Clusters formed, Time complexity also plays an important role based on the number of clusters and number of iterations as shown in table below:

Distance Functions / Factors	**Trigonometric**	**Chi-Square**	**Euclidean**	**Canberra**	**Bray-Curtis**	**Variational**
No of iterations	25	25	18	25	25	25
Execution time(in ms)	1863	2803	1608	1300	1168	659
No of Clusters	8	8	8	8	8	8
No of iterations	25	29	25	25	25	25
Execution time	1189	2321	1743	1114	1000	1534
No of Clusters	7	7	7	7	7	7
No of iterations	9	17	25	25	25	25
Execution time	358	1681	1539	1002	829	490
No of Clusters	6	6	6	6	6	6

(ms= milliseconds)

Table 14: Showing execution time for distances

K- Means when used over Cosine for data collections used takes longer time when compared to the distance metrics used for the thesis here. Here, Chi-Square though being the most effective in producing good cluster results takes longer time than other metrics even when there is change in number of clusters. Euclidean metrics work faster in some cases even when the clusters are increased. Canberra work better than Bray-Curtis and also forms better clusters. Variational works faster compared to any other distance metrics as the function used is simple to implement and takes less execution time.

5.3 Limitations of K-Means Algorithm

Though K- means is simple to implement and provides results, the clusters formed failed for few distance metrics. It fails when the documents size is too large and takes lot of time to run for few metrics. The algorithm does not achieve global minimum for the distance over the assignments. It uses discrete assignment rather than set of continuous parameters, therefore the minimum it reaches using the metric cannot be called local minimum [29]. As discussed in [10], the appropriate choice of k i.e. number of cluster or centroids can affect the result.

Number of Iterations also needs to be decided at start as sometimes due to lots of 0's in the weighted matrix the metric used can cause the algorithm to run continuously without stopping for long time as K- means is straightforward. Vector dimension has to be fixed based

on significant terms in order to get proper clusters. Also if appropriate metrics are not chosen, K-means gives incorrect results as for the documents collection set used in the thesis. With the use of Jaccard, Hellinger's distance and Harmonic mean it was difficult to form proper clusters.

CHAPTER 6

CONCLUSION AND FUTURE WORK

The rationale behind the thesis is to find the effects of different distance functions on document clustering using K-means algorithm. Several experiments were conducted on Reuters 21578 collection set using K-means clustering implementation as discussed in chapter 5. Based on the Analysis in chapter 6, we come to the conclusion that Chi-Square works best for the document collection with efficiency around 80 % followed by Canberra and Euclidean distances with 70 %. The results also indicate that the distance metrics like Bray-Curtis, Variational and Trigonometric function didn't produce good results.

As the number clusters were changed, it resulted in variation of cluster size and execution time which was longer for Chi-Square and shorter for Variational distance. Though this implementation provided good results in clustering documents, it doesn't work efficiently when vector size is increased. Though different methods were proposed in this line, a better approach in this direction will be to come up with an efficient distance metric that gives good clustering results and runs faster.

This thesis focused on using limited document set from Reuter's collection but can be expanded to huge document collection in future research work. Other distance metrics can also be used apart from the few discussed in this thesis for clustering documents.

BIBLIOGRAPHY

1. M.A.Heart and J.O.Pedersen. *Reexamining the cluster hypothesis. In Proceedings of SIGIR'96, pp.76-84, 1996.*

2. Valpola, H., *Bayesian Ensemble Learning for Nonlinear Factor Analysis. Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 108, Espoo 2000, 54 pp*. Published by the Finish Academies of Technology.

3. *An Architecture for Efficient Document Clustering and Retrieval on a Dynamic Collection of Newspaper Texts,* Alan F.Smeaton, Mark Burnett, Francis Crimmins and Gerard Quinn; Irsg98-clustering.pdf

4. Wikipedia, the free Encyclopedia, Data Clustering, 2008. http://en.wikipedia.org/wiki/Data_clustering

5. Douglas R. Cutting, David R. Karger, Jan O. Pederson, John W. Turkey, Scatter/Gather: *A Cluster-based Approach in Browsing Large Documents Collection, SIGIR '92, pages 318-329, 1992.*

6. *Data Mining: Concepts and Techniques* by Jiawei Han and Micheline Kamber.Second Edition. Page 383 – 460

7. Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, '*Introduction to Data Mining', Chapters 1, 5, 8 & 9, Addison Wesley, 2006*

8. *A Grid-based Clustering Algorithm using Adaptive Mesh Refinement*, Wei-keng Liao, Ying Liu, Alok Choudhary, 2004

9. T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, An Efficient k-Means Clustering Algorithm: Analysis and Implementation, *IEEE Trans. PAMI*, 24 (2002), 881-892.

10. K. Alsabti, S. Ranka and V. Singh, '*An Efficient K-means Clustering Algorithm', Proc. First Workshop High Performance Data Mining*, March 1998.

11. Teknomo, Kardi. K-Means Clustering Tutorials, http://people.revoledu.com/kardi/ tutorial/kMean/

12. A Tutorial on Clustering Alorithms, www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/

13. Distance Metrics , References
http://docs.scipy.org/doc/scipy/reference/spatial.distance.html

14. Distance Metrics, Greg Konchaski, http://konchanski.org/gpk

15. *Distance Metric learning, with application to clustering with side-Information* Eric P. Xing, Andrew Y. Ng, Michael I. Jordan and Stuart Russell, University of California, Berkeley

16. Sam's string Metrics
http://www.dcs.shef.ac.uk/~sam/stringmetrics.html

17. David D. Lewis , '*Reuters 21578, Distribution 1.0 Test collection'* (n.d.)
www.daviddlewis.com/resources/testcollections/reuters21578/

18. Dr. E. Garcia, '*Document Indexing Tutorial'*, 2006.
http://www.miislita.com/information-retrieval
tutorial/indexing.html

19. Kiran Pai, 'A simple way to read an XML file in Java', 2002.
http://www.developerfusion.com/code/2064/a-simple-way-to-read-an-xml file-in-java/

20. HappyCoders,'TokenizingJavasourcecode'(n.d.)
http://www.java.happycodings.com/Core_Java/code84.html

21. Martin Porter, 'The Porter Stemming Algorithm', Jan 2006.
http://tartarus.org/~martin/PorterStemmer/.

22. Wikipedia, the free Encyclopedia, Inverted Index, 2008.
http://en.wikipedia.org/wiki/Inverted_index

23. Managing Gigabytes - *Compressing and Indexing Documents and Images* by Ian H. Witten, Alistair Moffat, Timothy C. Bell

24. Yiming Yang, Jan O. Pederson, '*A comparative study on feature selection in text categorization', Proceedings of the fourteenth international Conference on Machine Learning*, pages: 412-420, 1997

25. TF-IDF, http://en.wikipedia.org/wiki/Term_frequency

26. David M Mount, '*KMlocal: A Testbed for k-means Clustering Algorithms'*, Version 1.7, University of Maryland, August 2005

27. V. Estivill-Castro, '*Why so many Clustering Algorithms'*, University of Newcastle, Australia

28. Bishop, C. M. *Neural Networks for Pattern Recognition.* Oxford, England: Oxford University Press, 1995.

29. T. Kanungo, D.M. Mount, N.S. Netanyahu, C. Piatko, R. Silverman, and A.Y. Wu, "*Computing Nearest Neighbors for Moving Points and Applications to Clustering," Proc. 10th Ann.* ACM-SIAM symp. *Discrete Algorithms, pp. S931-S932, Jan. 1999*

VITA

Graduate College
University of Nevada, Las Vegas

Rushikesh Veni

Home Address:
10175 Spring Mountain Rd,
Unit # 1053
Las Vegas, Nevada 89117, USA

Degrees:
Bachelor of Engineering, Computer Science, 2007
Osmania University

Master of Science, Computer Science, 2009
University of Nevada, Las Vegas

Thesis Title: Effects of Similarity Metrics on Document Clustering

Thesis Examination Committee:
Chairperson, Dr. Kazem Taghva, Ph.D.
Committee Member, Dr. Ajoy K. Datta, Ph.D.
Committee Member, Dr. Laxmi P. Gewali, Ph.D
Graduate College Representative, Dr. Muthukumar Venkatesan, Ph.D