BRAC UNIVERSITY

INTEGRATION OF DATA CLUSTERING TO INTELLIGENT EMAIL CLASSIFIER

Supervisor: MOIN MOSTAKIM

Prepared By:

Shahriar Ahmed Zisan (14241010)

Signature of supervisor:

ABSTRACT

In recent years there are more than billion of email users .People are dependent on electronic mails. It has become an urgent and crucial component of communication .One of the cheapest and fastest communication mean. Emails can grow at a large scale. Sorting the emails according to their content can be a useful and time saving. Cluster analysis can come in action while classifying the email documents .Cluster analysis is a sub-field in machine learning a subfield of artificial intelligence, that refers to a group of algorithms that try to find a natural grouping of objects based on some objective metric. In general this problem is hard because a good grouping might be subjective. The k-means algorithm is one of the simplest and predominantly used algorithms for unsupervised learning procedure clustering. Existing email clusters are supervised clusters.

In this thesis, we inherit the simplest machine learning strategies like k-means, k nearest neighbors and represent an intelligent email classifier combined with both supervised and unsupervised algorithm that categorizes email documents based on the body contents.

ACKNOWLEDGEMENT

I would like to thank Mr .Moin Mostakim Brac University for agreeing to supervise me with my thesis. His patience and confidence in me has been a source of encouragement and this thesis would not have been possible without the great support, inspiration and influence of him. I believe his dedication to this paper deserves to be reciprocated with great gratitude. I would also like to thank Anisul Amin – 10301001 for his guidance in the process of writing the thesis. Writing this thesis has been hard but in the phase of writing I feel I have learned a lot about clustering.

A special thanks to the Thesis committee for taking the time to review and evaluate my thesis as part of our undergraduate program.

Table Of Contents

1 Introduction

In the current digital era more than 90% of the people are connected to the internet. The same percentage are of email users . Considering this the amount of email messages sent and received has grown significantly and more than before we are seriously facing a message overload problem. To make managing and finding messages easier it is reasonable to classify messages based on user needs. The specific way for classifying emails can be developed by every person just the way it is reasonable for the specific user. New innovative features in our email systems can make our life more smoother .Clustering of email documents can be very much productive and time saving .It can make communication much more faster .Current clustering approaches are of type supervised . The process first gets trained with test documents and develop patterns or similar structures to recognize new incoming emails .It is also sorted with the aid of timestamp of the emails .This thesis accomplishes clustering in a separate way which will be discussed in few minutes .Few current clustering involves the header informations of the emails to clusterize mail documents .Some mail classifier also detects spam emails. Existing email clusters are of these type. Our work only considers all messages from the inbox folder only.It deals with raw text from the email bodies .Any emails containing attachments or html mails are discarded . Clustering techniques are from a subpart of machine learning or information retrieval processes which are indeed sub branches of artificial intelligence. The most challenging part of classification includes the determinations of initial seeds or centroids . The accuracy of the classification totally depends on the centroids.Bad selection of centroid can lead a poor result .As clustering is a part of information retrieval schemes , a vast area of artificial intelligence, it also includes basic natural language processing and linguistics in the preprocessing phase.

As email documents will be shorter than any other documents, we will get some advantages in the preprocessing phase and might get some disadvantages in the output as the emails will be not uniform and therefore will lack key informations.

In this thesis our goal is to cluster incoming inbox emails from personal account to PRE-DEFINED labels given by the user. Cluster based on the raw content of the messages .Messages with in the cluster must be of the same topic .In this paper we take a total sample of 550 messages retrieved from the inbox folder. Emails that are very much far from the given user labels will be stored in an extra different folder named " OTHERS ".Once the documents are classified a supervised 'lazy' technique will be applied on the folder OTHERS, to extract documents which are supposed to be assigned in predefined labels but failed due to the noise of the sample input data.

1.1 Related Works

Our goal was to categorize incoming mails according predefined labels. Several works regarding data mining have been carried out in the last few decades .Data mining techniques are used to accomplish the tasks. So we review the representative papers and research works on these different topics.

Existing works on web text mining and clustering are mainly focused on the different levels like: Web text clustering, Data text mining, Web page information extraction etc. Web data clustering researchers Bouras and Tsogkas (2010) proposed an enhanced model based on the standard k-means algorithm using the external information extracted from WordNet hypernyms in a twofold manner: enriching the "bag of words" used prior to the clustering process and assisting the label generation procedure following it. Murali Krishna and Durga Bhavani (2010) proposed the use of a renowned method, called Apriori algorithm, for mining the frequent item sets and devised an efficient approach for text clustering based on the frequent item sets. Maheshwari and Agrawal (2010) proposed centroid-based text clustering for preprocessed data, which is a supervised approach to classify a text into a set of predefined classes with relatively low computation and better accuracy. Qiujun (2010) proposed a new approach to news content extraction using similarity measure based on edit distance to separate the news content from noisy information. Jaiswal (2007) performed the comparison of different clustering methods like K-Means, Vector Space Model (VSM), Latent Semantic Indexing (LSI), and Fuzzy C-Means (FCM) and selected FCM for web clustering.

1.2 Contributions

The contributions of us in thesis is summarized below.

1.Extract raw email content from user given credentials.Emails with pictures,attachments are avoided

2.Removal of stopwords and the creation of vector space model.

3.Assigning smart weights like term frequency and inverse document frequency.

4.Pre defined labels are computed by the means of the above weights .

5.Applying traditional K means algorithm to sort the emails.An unsupervised technique to train the documents.

6.Applying lazy k neighbours algorithm to the contents that were unable to assign by kmeans algorithm in the first place.

2 Literature Review

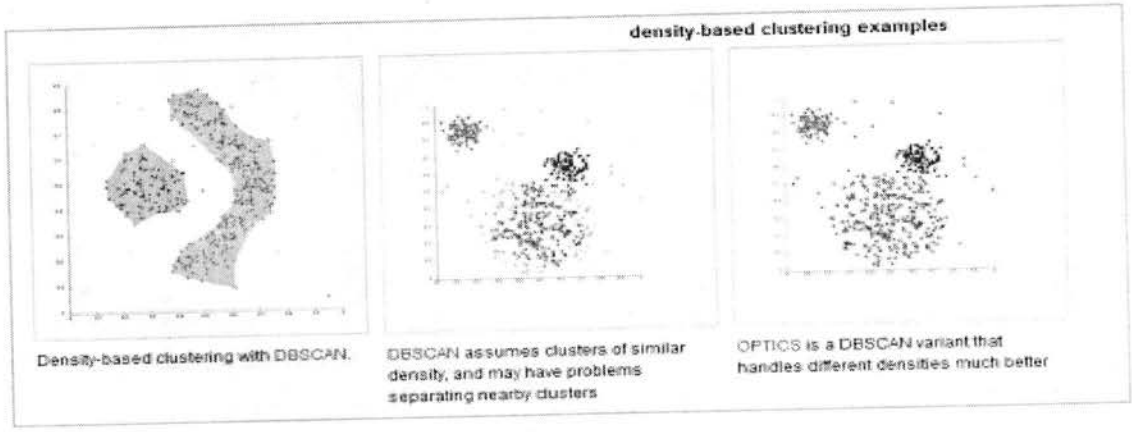
We studied various clustering algorithms i.e. K-Means, K-Medoid, DBSCAN, CURE, BIRCH, CLARAN, CHAMELEON, and their variants which use expectation minimization techniques.

Typical clustering models: Many different clustering techniques that have been proposed over the years these techniques can be described using the following criteria or models.

Centroid models Partition clustering techniques create a one-level partitioning of the data points. K-means, Kmedoid are the two examples of this model. Both these techniques are based on the idea that a centre point can represent a cluster. For K-means we use the notion of a centroid, which is the mean or median point of a group of points. For K-medoid we use the notion of a medoid, which is the most representative (central) point of a group of points. By its definition a medoid is required to be an actual data point. CLARANS is a more efficient version of the basic K-medoid and is used in spatial data mining problems.

Centroid models Partition clustering techniques create a one-level partitioning of the data points. K-means, Kmedoid are the two examples of this model. Both these techniques are based on the idea that a centre point can represent a cluster. For K-means we use the notion of a centroid, which is the mean or median point of a group of points. For K-medoid we use the notion of a medoid, which is the most representative (central) point of a group of points. By its definition a medoid is required to be an actual data point. CLARANS is a more efficient version of the basic K-medoid and is used in spatial data mining problems.

Fig :1



Hierarchical Clustering In hierarchical clustering the goal is to produce a hierarchical series of nested clusters, ranging from clusters of individual points at the bottom to an all-inclusive cluster at the top. 5 A diagram called a dendogram graphically represents this hierarchy and is an inverted tree that describes the order in which points are merged (bottom-up view) or clusters are split (top-down view). Hierarchical clustering builds models based on distance connectivity.

Connectivity based clustering, also known as hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away. As such, these algorithms connect "objects" to form "clusters" based on their distance. A cluster can be described largely by the maximum distance needed to connect parts of the cluster. At different distances, different clusters will form, which can be represented using a dendrogram, which explains where the common name "hierarchical clustering" comes from.

Distribution models: The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution. A nice property of this approach is that this closely resembles the way artificial data sets are generated: by sampling random objects from a distribution. Clusters are modelled using statistic distributions, such as multivariate normal distributions used by the Expectationmaximization algorithm.

Graph-Based Clustering: The hierarchical clustering algorithms can be viewed as operating on a proximity graph. However, they are most commonly viewed in terms of merging or splitting clusters, and often there is no mention of graph related concepts. There are some clustering techniques, however, that are explicitly cast in terms of a graph or a hyper graph. Many of these algorithms are based on the idea of looking at the nearest neighbours of a point.

2.2 Related Work There are many clustering algorithms based on the models listed above, we studied some selected algorithms from these clustering models to understand concepts behind each algorithm. By this study we tried to find out the limitations of these algorithms.

K-Mean Clustering [8] The K-means clustering technique is very simple. K-means uses the notion of a centroid, which is the mean or median point of a group of points. a centroid almost never corresponds to an actual data point. Basic Algorithm for finding K clusters. Given k, the k-means algorithm is implemented in 4 steps:

1. Partition objects into k nonempty subsets.

2. Compute seed points as the centroids of the clusters of the current partition. The centroid is the centre (mean point) of the cluster.

3. Assign each object to the cluster with the nearest seed point.

4. Go back to Step 2, stop when no more new assignment.

K-Medoids Clustering [9] The objective of K-medoid clustering is to find a non-overlapping set of clusters such that each cluster has a most representative point, i.e., a point that is most centrally located with respect to some measure, e.g., distance. These representative points are called medoids. Basic K-medoid Algorithm for finding K clusters:

1. Select K initial points. These points are the candidate medoids and are intended to be the most central points of their clusters.

2. Consider the effect of replacing one of the selected objects (medioids) with one of the non-selected objects. The distance of each non-selected point from the closest candidate medoid is calculated, and this distance is summed over all points. This distance represents the "cost" of the current configuration. All possible swaps of a non-selected point for a selected one are considered, and the cost of each configuration is calculated.

3. Select the configuration with the lowest cost. If this is a new configuration, then repeat step 2.

4. Otherwise, associate each non-selected point with its closest selected point (medoid) and stop.

CLARANS itself grew out of two clustering algorithms, PAM and CLARA, it was developed specifically for use in spatial data mining. PAM (Partitioning Around Medoids) is a "K-medoid" based clustering algorithm that attempts to cluster a set of m points into K clusters. CLARA (Clustering LARge Applications) is an adaptation of PAM, for handling larger data sets. It works by repeatedly sampling a set of data points, calculating the medoids of the sample, and evaluating the cost of the configuration that consists of these "sample-derived" medoids and the entire data set. The set of medoids that minimizes the cost is selected. The algorithm can be stated as follows:

1. Randomly pick K candidate medoids.

2. Randomly consider a swap of one of the selected points for a non-selected point.

3. If the new configuration is better, i.e., has lower cost, then repeat step 2 with the new configuration. 4. Otherwise, repeat step 2 with the current configuration unless a parameterized limit has been exceeded. (For example: limit is set to max (250, K *(m - K)).

5. Compare the current solution with any previous solutions and keep track of the best.

6. Return to step 1 unless a parameterized limit has been exceeded.

DBSCAN is a density based clustering algorithm that works with a number of different distance metrics. When DBSCAN has processed a set of data points, a point will either be in 7 a cluster or will be classified as noise. DBSCAN is based on the concepts of a point being "density reachable" and "density connected". Conceptually, data points fall into three classes

: 1. Core points. These are points that are at the interior of a cluster. A point is an interior point if there are enough points in its neighbourhood, i.e., if the number of points within a given

neighbourhood around the point exceeds a certain threshold, as determined by the distance function and a supplied distance parameter. If two core points belong to each other□s neighbourhoods, then the core points belong to the same cluster.

2. Border points. A border point is a point that is not a core point, i.e., there are not enough points in its neighbourhood, but it falls within the neighbourhood of a core point.

3. Noise points. A noise point is any point that is not a core point or a border point.

Thus, for DBSCAN, a cluster is the set of all core points whose neighbourhoods transitively connect them together, along with some border points.

CURE (Clustering Using Representatives) is a clustering algorithm that uses a variety of different techniques to create an approach which can handle large data sets, outliers, and clusters with non-spherical shapes and non-uniform sizes. We summarize our description of CURE by explicitly listing the different steps:

1. Draw a random sample from the data set.

2. Partition the sample into p equal sized partitions.

3. Cluster the points in each cluster using the hierarchical clustering algorithm to obtain m/pq clusters in each partition and a total of m/q clusters. Some outlier elimination occurs during this process.

4. Eliminate outliers. This is the second phase of outlier elimination.

5. Assign all data to the nearest cluster to obtain a complete clustering.

Chameleon is a clustering algorithm that combines an initial partitioning of the data using an efficient graph partitioning algorithm with a novel hierarchical clustering scheme that dynamically models clusters. The key idea is that two clusters will be merged only if the resulting cluster is similar to the original clusters, i.e. self-similarity is preserved. In short, the steps of the Chameleon algorithms are:

1. Build a k-nearest neighbour graph.

2. Partition the graph into partitions using a multilevel graph-partitioning algorithm.

3. Perform a hierarchical clustering starting with the partitions. This hierarchical clustering will merge the clusters which best preserve the cluster self-similarity with respect to relative interconnectivity and relative closeness.

BIRCH (Balanced and Iterative Reducing and Clustering using Hierarchies) is based on the notion of a clustering feature (CF) and a CF tree. The idea is that a cluster of data points (vectors) can be represented by a triple of numbers (N, LS, SS), where N is the number of points in the cluster, LS is the linear sum of the points, and SS is the sum of squares of the points. A CF tree is built as the data is scanned. As each data point is encountered, the CF tree is traversed, starting from the root and choosing the closest node at each level. When the closest "leaf" cluster for the current data point is finally identified, a test is performed to see if adding the data item to the candidate cluster will result in a new cluster with a diameter greater than the given threshold, T. If not, then the data point is "added" to the candidate cluster by updating the CF information. The cluster information for all nodes from the leaf to the root is also updated. BIRCH consists of a number of phases beyond the initial creation of the CF tree. The phases of BIRCH are as follows:

1. Load the data into memory by creating a CF tree that "summarizes" the data.

2. Build a smaller CF tree if it is necessary for phase 3. T is increased, and then the leaf node entries (clusters) are reinserted. Since T has increased, some clusters will be merged.

3. Perform global clustering. Different forms of global clustering (clustering which uses the pair wise distances between all the clusters) can be used.

4. Redistribute the data points using the centroids of clusters discovered in step 3 and thus, discover a new set of clusters. By repeating this phase, multiple times, the process converges to a local minimum. Because of page size constraints and the T parameter, points that should be in one cluster are sometimes split, and points that should be in different clusters are sometimes combined. Also, if the data set contains duplicate points, these points can sometimes be clustered differently, depending on the order in which they are encountered.

3 Technical Review

3.1 OVERVIEW OF KMEANS ALGORITHM

k-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move any more. Finally, this algorithm aims at minimizing an objective function know as squared error function given by:

$$J(V) = \sum_{i=1}^{c} \sum_{j=1}^{c_i} \left(\left\| x_i - v_j \right\| \right)^2$$

where,

'$||x_i - v_j||$' is the Euclidean distance between x_i and v_j.

'c_i' is the number of data points in i^{th} cluster.

'c' is the number of cluster centers.

Algorithmic steps for k-means clustering

Let $X = \{x_1, x_2, x_3, \ldots\ldots, x_n\}$ be the set of data points and $V = \{v_1, v_2, \ldots\ldots, v_c\}$ be the set of centers.

1) Randomly select 'c' cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..

4) Recalculate the new cluster center using:

$$J(V) = \sum_{i=1}^{c} \sum_{j=1}^{c_i} \left(\left\| x_i - v_j \right\| \right)^2$$

where, 'c_i' represents the number of data points in i^{th} cluster.

5) Recalculate the distance between each data point and new obtained cluster centers.

6) If no data point was reassigned then stop, otherwise repeat from step 3).

Advantages

1) Fast, robust and easier to understand.

2) Relatively efficient: O(tknd), where n is # objects, k is # clusters, d is # dimension of each object, and t is # iterations. Normally, k, t, d << n.

3) Gives best result when data set are distinct or well separated from each other.

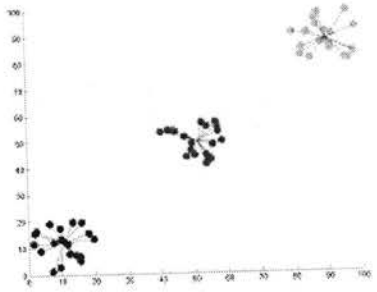


Fig I: Showing the result of k-means for 'N' = 60 and 'c' = 3

Note: For more detailed figure for k-means algorithm please refer to k-means figure sub page.

Disadvantages

1) The learning algorithm requires apriori specification of the number of cluster centers.

2) The use of Exclusive Assignment - If there are two highly overlapping data then k-means will not be able to resolve that there are two clusters.

3) The learning algorithm is not invariant to non-linear transformations i.e. with different representation of data we get

different results (data represented in form of cartesian co-ordinates and polar co-ordinates will give different results).

4) Euclidean distance measures can unequally weight underlying factors.

5) The learning algorithm provides the local optima of the squared error function.

6) Randomly choosing of the cluster center cannot lead us to the fruitful result. Pl. refer Fig.

7) Applicable only when mean is defined i.e. fails for categorical data.

8) Unable to handle noisy data and outliers.

9) Algorithm fails for non-linear data set.

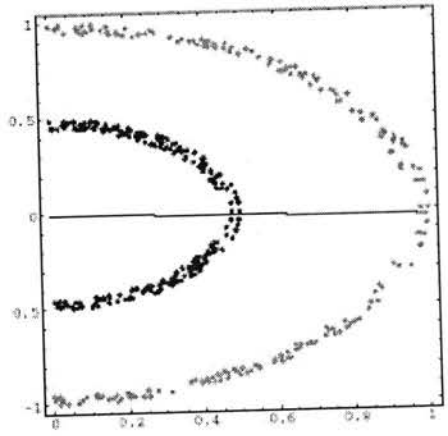


Fig II: Showing the non-linear data set where k-means algorithm fails

3.2 OVERVIEW OF K NEAREST NEIGHBOUR ALGORITHM

The nearest neighbor (NN) rule identifies the category of unknown data point on the basis of its nearest neighbor whose class is already known. This rule is widely used in pattern recognition , text categorization , ranking models , object recognition and event recognition applications. T. M. Cover and P. E. Hart purpose k-nearest neighbor (kNN) in which nearest neighbor is calculated on the basis of value of k, that specifies how many nearest neighbors are to be

considered to define class of a sample data point . T. Bailey and A. K. Jain improve kNN which is based on weights . The training points are assigned weights according to their distances from sample data point. But still, the computational complexity and memory requirements remain the main concern always. To overcome memory limitation, size of data set is reduced. For this, the repeated patterns, which do not add extra information, are eliminated from training samples . To further improve, the data points which do not affect the result are also eliminated from training data set . Besides the time and memory limitation, another point which should be taken care of, is the value of k, on the basis of which category of the unknown sample is determined. Gongde Guo selects the value of k using model based approach . The model proposed automatically selects the value of k. Similarly, many improvements are proposed to improve speed of classical kNN using concept of ranking , false neighbor information , clustering . The NN training data set can be structured using various techniques to improve over memory limitation of kNN. The kNN implementation can be done using ball tree , k-d tree , nearest feature line (NFL) , tunable metric , principal axis search tree and orthogonal search tree . The tree structured training data is divided into nodes, whereas techniques like NFL and tunable metric divide the training data set according to planes. These algorithms increase the speed of basic kNN algorithm.

Nearest neighbor techniques are divided into two categories: 1) Structure less and 2) Structure Based. A. Structure less NN techniques The k-nearest neighbor lies in first category in which whole data is classified into training data and sample data point. Distance is evaluated from all training points to sample point and the point with lowest distance is called nearest neighbor. This technique is very easy to implement but value of k affects the result in some cases. Bailey uses weights with classical kNN and gives algorithm named weighted kNN (WkNN) . WkNN evaluates the distances as per value of k and weights are assigned to each calculated value, and then nearest neighbor is decided and class is assigned to sample data point. The Condensed Nearest Neighbor (CNN) algorithm stores the patterns one by one and eliminates the duplicate ones. Hence, CNN removes the data points which do not add more information and show similarity with other training data set. The Reduced Nearest Neighbor (RNN) is improvement over CNN; it includes one more step that is elimination of the patterns which are not affecting the training data set result. The another technique called Model Based kNN selects similarity measures and create a 'similarity matrix' from given training set. Then, in the same category, largest local neighbor is found that covers large number of neighbors and a data tuple is located with largest global neighborhood. These steps are repeated until all data tuples are grouped. Once data is formed using model, kNN is executed to specify category of unknown sample. Subash C. Bagui and Sikha Bagui improve the kNN by introducing the concept of ranks. The method pools all the observations belonging to different categories and assigns rank to each category data in ascending order. Then observations are counted and on the basis of ranks class is assigned to unknown sample. It is very much useful in case of multi-variants data. In Modified kNN, which is modification of WkNN validity of all data samples in the training data set is computed, accordingly weights are assigned and then validity and weight both together set basis for classifying the class of the sample data point. Yong zeng, Yupu Zeng and Liang Zhou define the

new concept to classify sample data point. The method introduces the pseudo neighbor, which is not the actual nearest neighbor; but a new nearest neighbor is selected on the basis of value of weighted sum of distances of kNN of unclassified patterns in each class. Then Euclidean distance is evaluated and pseudo neighbor with greater weight is found and classified for unknown sample. In the technique purposed by Zhou Yong , Clustering is used to calculate nearest neighbor. The steps include, first of all removing the samples which are lying near to the border, from training set. Cluster each training set by k value clustering and all cluster centers form new training set. Assign weight to each cluster according to number of training samples each cluster have.

B. Structure based NN techniques The second category of nearest neighbor techniques is based on structures of data like Ball Tree, k-d Tree, principal axis Tree (PAT), orthogonal structure Tree (OST), Nearest feature line (NFL), Center Line (CL) etc. Ting Liu introduces the concept of Ball Tree. A ball tree is a binary tree and constructed using top down approach. This technique is improvement over kNN in terms of speed. The leaves of the tree contain relevant information and internal nodes are used to guide efficient search through leaves. The k-dimensional trees divide the training data into two parts, right node and left node. Left or right side of tree is searched according to query records. After reaching the terminal node, records in terminal node are examined to find the closest data node to query record. The concept of NFL given by Stan Z.Li and Chan K.L. divide the training data into plane. A feature line (FL) is used to find nearest neighbor. For this, FL distance between query point and each pair of feature line is calculated for each class. The resultant is set of distances. The evaluated distances are sorted into ascending order and the NFL distance is assigned as rank 1. An improvement made over NFL is Local Nearest Neighbor which evaluates the feature line and feature point in each class, for points only, whose corresponding prototypes are neighbors of query point. Yongli Zhou and Changshui Zhang introduce new metric for evaluating distances for NFL rather than feature line. This new metric is termed as "Tunable Metric". It follows the same procedure as NFL but at first stage it uses tunable metric to calculate distance and then implement steps of NFL. Center Based Nearest Neighbor is improvement over NFL and Tunable Nearest Neighbor. It uses center base line (CL) that connects sample point with known labeled points. First of all CL is calculated, which is straight line passing through training sample and center of class. Then distance is evaluated from query point to CL, and nearest neighbor is evaluated. PAT permits to divide the training data into efficient manner in term of speed for nearest neighbor evaluation. It consists of two phases 1) PAT Construction 2) PAT Search. PAT uses principal component analysis (PCA) and divides the data set into regions containing the same number of points. Once tree is formed kNN is used to search nearest neighbor in PAT. The regions can be determined for given point using binary search. The OST uses orthogonal vector. It is an improvement over PAT for speedup the process. It uses concept of "length (norm)", which is evaluated at first stage. Then orthogonal search tree

is formed by creating a root node and assigning all data points to this node. Then left and right nodes are formed using pop operation.

ADVANTAGES AND DISADVANTAGES OF KNN

Robust to noisy training data(especially if we use inverse square of weighted distance as the "distance")

Effective if the training data is large.

Need to determine the value of parameter K(number of nearest neighbours)

Computation cost is high because we need to calculate of query instance to all training samples.Some indexing like KD Tree may reduce the computational cost.

3.3 EMAIL MESSAGE FORMAT

The email message format is defined in RFC (Request for Comments) 5322 (released in October 2008) and also some additional RFCs from 2045 to 2049. Collectively, these RFCs are called Multipurpose Internet Mail Extensions, or in short MIME.

An email message consists of two major sections:

- Header contains information about the sender, receiver, subject, date, etc
- Body is the message itself as text and is the same as the body of a regular letter

Now, let's take a look at the main fields of an email header:

- Date – time of sending out the email message
- From – usually the author of the email
- To – one or many recipients of email
- Cc – recipients who are not directly related to message but may be interested in the information containing in email
- Bcc – recipients on that filed will remain invisible to other addressees

• Subject – a short summary of the contents of email All these fields contain valuable information to classify an email message. In addition we can also use information in the email body and if an attachment is added, this is also a good piece of information to use. The email body can be written in plain text or in HTML.

Our thesis only work with the body part of the email. So we will be dealing with raw text only.

Several methods and functions was required to carry out the raw text messages and to save as text documents which are fetched for preprocessing system.

3.4)STOP WORD REMOVAL

When working with text mining applications, we often hear of the term "stop words" or "stop word list" or even "stop list". Stop words are basically a set of commonly used words in any language, not just English. The reason why stop words are critical to many applications is that, if we remove the words that are very commonly used in a given language, we can focus on the important words instead. For example, in the context of a search engine, if your search query is "how to develop information retrieval applications", If the search engine tries to find web pages that contained the terms "how", "to" "develop", "information", "retrieval", "applications" the search engine is going to find a lot more pages that contain the terms "how", "to" than pages that contain information about developing information retrieval applications because the terms "how" and "to" are so commonly used in the English language. So, if we disregard these two terms, the search engine can actually focus on retrieving pages that contain the keywords: "develop" "information" "retrieval" "applications" – which would more closely bring up pages that are really of interest. This is just the basic intuition for using stop words.

Stop words are generally thought to be a **"single set of words"**. It really can mean different things to different applications. For example, in some applications removing all stop words right from determiners (e.g. the, a, an) to prepositions (e.g. above, across, before) to some adjectives

(e.g. good, nice) can be an appropriate stop word list. To some applications however, this can be detrimental. For instance, in sentiment analysis removing adjective terms such as 'good' and 'nice' as well as negations such as 'not' can throw algorithms off their tracks. In such cases, one can choose to use a minimal stop list consisting of just determiners or determiners with prepositions or just coordinating conjunctions depending on the needs of the application.

3.5 VECTOR SPACE MODEL

The vector space model procedure can be divided in to three stages. The first stage is the document indexing where content bearing terms are extracted from the document text. The second stage is the weighting of the indexed terms to enhance retrieval of document relevant to the user. The last stage ranks the document with respect to the query according to a similarity measure.

The vector space model has been criticized for being ad hoc

It is obvious that many of the words in a document do not describe the content, words like *the, is.* By using automatic document indexing those non significant words (function words) are removed from the document vector, so the document will only be represented by content bearing words. This indexing can be based on term frequency, where terms that have both high and low frequency within a document are considered to be function words. In practice, term frequency has been difficult to implement in automatic indexing. Instead the use of a stop list which holds common words to remove high frequency words (stop words) which makes the indexing method language dependent. In general, 40-50% of the total number of words in a document are removed with the help of a stop list .

Non linguistic methods for indexing have also been implemented. Probabilistic indexing is based on the assumption that there is some statistical difference in the distribution of content bearing words, and function words .Probabilistic indexing ranks the terms in the collection w.r.t. the term frequency in the whole collection. The function words are modeled by a Poisson distribution over all documents, as content bearing terms cannot be modeled. The use of Poisson model has been expand to Bernoulli model .Recently, an automatic indexing method which uses serial clustering of words in text has been introduced .The value of such clustering is an indicator if the word is content bearing.

3.6 TERM FREQUENCY AND INVERSE DOCUMENT FREQUENCY

In information retrieval, **tf–idf**, short for **term frequency–inverse document frequency**, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. tf–idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification.

One of the simplest ranking functions is computed by summing the tf–idf for each query term; many more sophisticated ranking functions are variants of this simple model.

Suppose we have a set of English text documents and wish to determine which document is most relevant to the query "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its *term frequency*.

The first form of term weighting is due to Hans Peter Luhn (1957) and is based on the Luhn Assumption:

- The weight of a term that occurs in a document is simply proportional to the term frequency.

In the case of the **term frequency** $\mathrm{tf}(t,d)$, the simplest choice is to use the *raw frequency* of a term in a document, i.e. the number of times that term t occurs in document d. If we denote the raw frequency of t by $f_{t,d}$, then the simple tf scheme is $\mathrm{tf}(t,d) = f_{t,d}$. Other possibilities include

- Boolean "frequencies": $\mathrm{tf}(t,d) = 1$ if t occurs in d and 0 otherwise;
- logarithmically scaled frequency: $\mathrm{tf}(t,d) = 1 + \log f_{t,d}$, or zero if $f_{t,d}$ is zero;
- augmented frequency, to prevent a bias towards longer documents, e.g. raw frequency divided by the maximum raw frequency of any term in the document:

$$\mathrm{tf}(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

Variants of TF weight

weighting scheme	TF weight
binary	$0, 1$
raw frequency	$f_{t,d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K)\frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Because the term "the" is so common, term frequency will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less common words "brown" and "cow". Hence an *inverse document frequency* factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.

Karen Spärck Jones (1972) conceived a statistical interpretation of term specificity called IDF, which became a cornerstone of term weighting:

- The specificity of a term can be quantified as an inverse function of the number of documents in which it occurs.

The **inverse document frequency** is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

with

- N: total number of documents in the corpus $N = |D|$
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\mathrm{tf}(t, d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

Then tf–idf is calculated as

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. Since the ratio inside the idf's log function is always greater than or equal to 1, the value of idf (and tf-idf) is greater than or equal to 0. As a term appears in more documents, the ratio inside the logarithm approaches 1, bringing the idf and tf-idf closer to 0.

Recommended TF-IDF weighting schemes

weighting scheme	document term weight	query term weight
1	$f_{t,d} \cdot \log \frac{N}{n_t}$	$\left(0.5 + 0.5 \frac{f_{t,q}}{\max_t f_{t,q}}\right) \cdot \log \frac{N}{n_t}$
2	$1 + \log f_{t,d}$	$\log(1 + \frac{N}{n_t})$
3	$(1 + \log f_{t,d}) \cdot \log \frac{N}{n_t}$	$(1 + \log f_{t,q}) \cdot \log \frac{N}{n_t}$

3.7 THE DOT PRODUCT

Let's begin with the definition of the dot product for two vectors:

$\vec{a} = (a_1, a_2, a_3, \ldots)$ and $\vec{b} = (b_1, b_2, b_3, \ldots)$, where an and bn are the components of the vector (features of the document, or TF-IDF values for each word of the document in our example) and the n is the dimension of the vectors:

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$$

As you can see, the definition of the dot product is a simple multiplication of each component from the both vectors added together. See an example of a dot product for two vectors with 2 dimensions each (2D):

$\vec{a} = (0, 3)$
$\vec{b} = (4, 0)$
$\vec{a} \cdot \vec{b} = 0 * 4 + 3 * 0 = 0$

The first thing you probably noticed is that the result of a dot product between two vectors isn't another vector but a single value, a scalar.

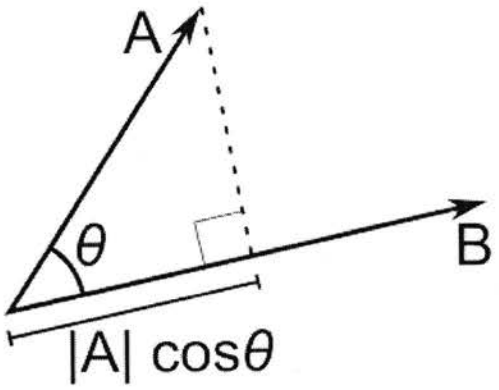
This is all very simple and easy to understand, but what is a dot product ? What is the intuitive idea behind it ? What does it mean to have a dot product of zero ? To understand it, we need to understand what is the geometric definition of the dot product:

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

Rearranging the equation to understand it better using the commutative property, we have:

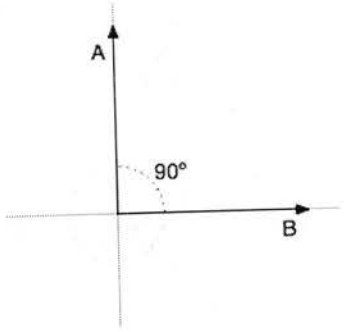
$$\vec{a} \cdot \vec{b} = \|\vec{b}\| \|\vec{a}\| \cos\theta$$

So, what is the term $\|\vec{a}\| \cos\theta$? This term is the projection of the vector $\vec{a}$ into the vector $\vec{b}$ as shown on the image below:



The projection of the vector A into the vector B. By Wikipedia.

Now, what happens when the vector $\vec{a}$ is orthogonal (with an angle of 90 degrees) to the vector $\vec{b}$ like on the image below ?



Two orthogonal vectors (with 90 degrees angle).

There will be no adjacent side on the triangle, it will be equivalent to zero, the term $\|\vec{a}\| \cos\theta$ will be zero and the resulting multiplication with the magnitude of the vector $\vec{b}$ will also be zero. Now you know that, when the dot product between two different vectors is zero, they are orthogonal to each other (they have an angle of 90 degrees), this is a very neat way to check the orthogonality of different vectors. It is also important to note that we are using 2D examples, but the most amazing fact about it is that we can also calculate angles and similarity between vectors in higher dimensional spaces, and that is why math let us see far than the obvious even when we can't visualize or imagine what is the angle between two vectors with twelve dimensions for instance.

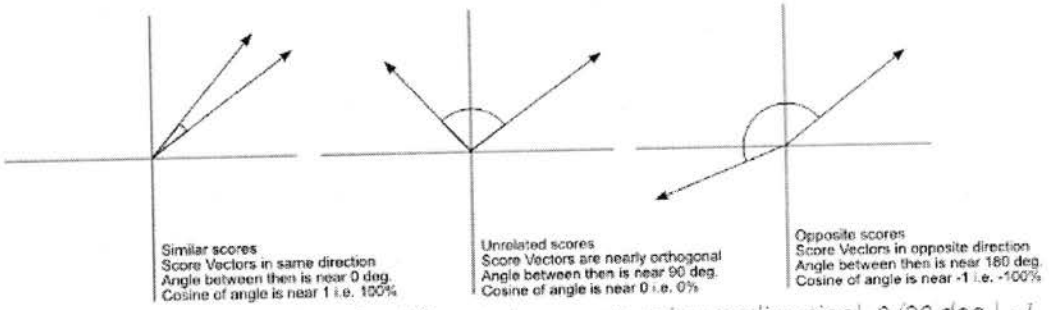
3.8 THE COSINE SIMILARITY

The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle between them. This metric is a measurement of orientation and not magnitude, it can be seen as a comparison between documents on a normalized space because we're not taking into the consideration only the magnitude of each word count (tf-idf) of each document, but the angle between the documents. What we have to do to build the cosine similarity equation is to solve the equation of the dot product for the cosØ:

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

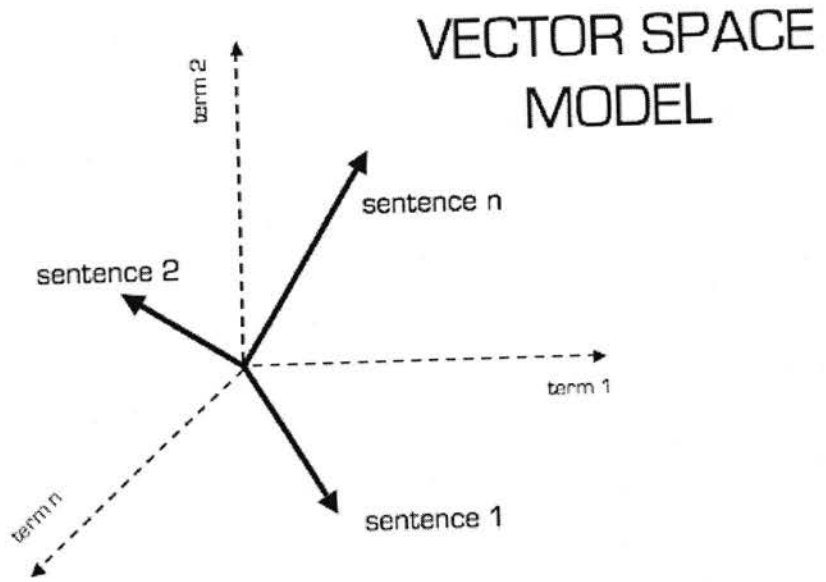
And that is it, this is the cosine similarity formula. Cosine Similarity will generate a metric that says how related are two documents by looking at the angle instead of magnitude, like in the examples below:



The Cosine Similarity values for different documents, 1 (same direction), 0 (90 deg.), -1 (opposite directions).

Note that even if we had a vector pointing to a point far from another vector, they still could have an small angle and that is the central point on the use of Cosine Similarity, the measurement tends to ignore the higher term count on documents. Suppose we have a document with the word "sky" appearing 200 times and another document with the word "sky" appearing 50, the Euclidean distance between them will be higher but the angle will still be small because they are pointing to the same direction, which is what matters when we are comparing documents.

Now that we have a Vector Space Model of documents (like on the image below) modeled as vectors (with TF-IDF counts) and also have a formula to calculate the similarity between different documents in this space.



Vector Space Model

3.9 CENTROID SELECTION

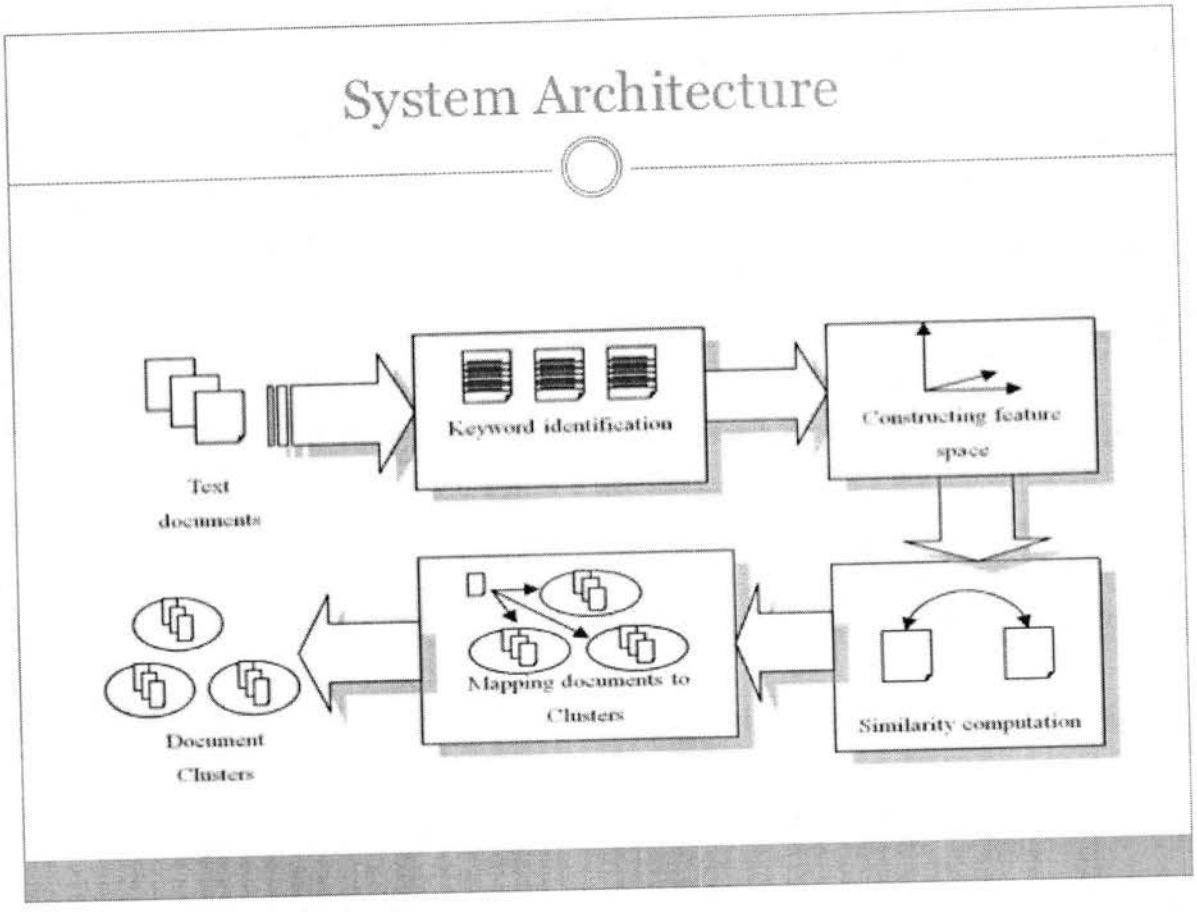
Centroid selection is the most trickiest part of the entire process.Here centroid is meant by a document which represents all the other documents in that cluster,that is all the documents of that cluster is very much relevant or similar to the centroid document.One can randomly assign a centroid and calculate rest of the centroids by the aid of the distant measurement like cosine similarities.That is the next centroid will be the one which has the maximum differences from the randomly chosen one, and the process continues until there are k centroids.The methodology does not work well all the time.It has some limitations.The random selection might not pick the correct document vector for the initial centroids and thus the rest of the centroids might end up with ambiguous centroids.In this thesis we predefined the labels for initial centroids.The user will know what and how many categories there might be in their inboxes.So it is upto the user who takes the responsibilities of choosing the predefined label.And therefore the clusters centroids are picked as the labels created by the user.

4 System Design ,Tools ,Algorithm and Methodology

For our experiment, we used pure JAVA and the eclipse luna IDE the editor.

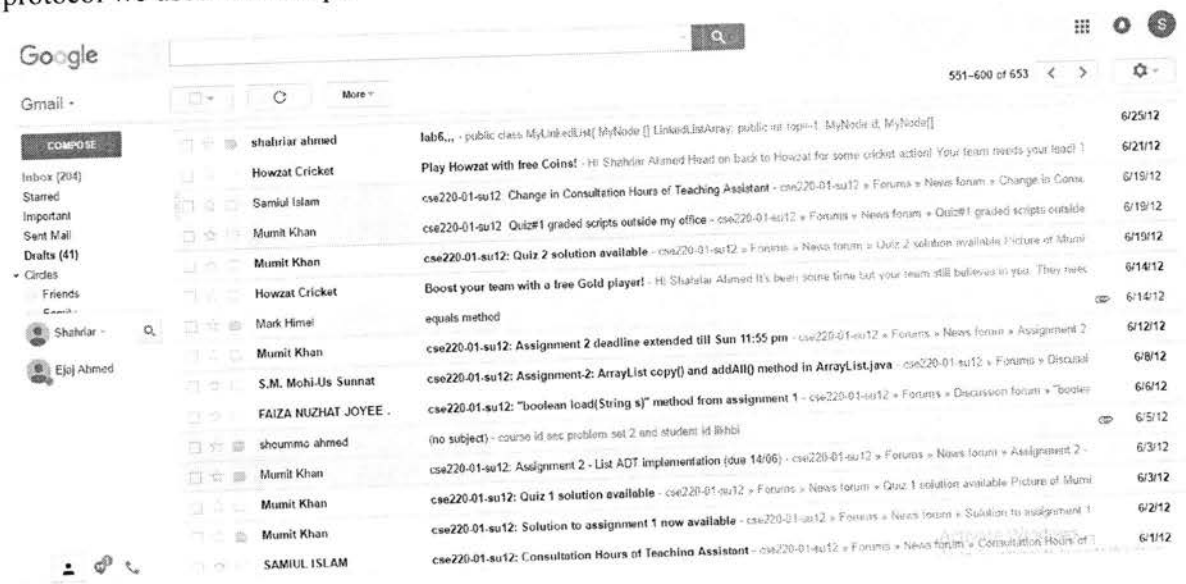
Algorithm we used is Kmeans and K nearest neighbor algorithm.

Methodologies we followed are Morphologically Open Image, Subtracting Back ground from Email body extraction , Tokenizing, Stopword removal ,constructing VSM ,Normalizing the vectors ,Assigning TFIDF, and hence measuring the closeness with cosine similarities etc



4.1 DATA COLLECTION

For the purpose of the thesis our data was from my personal email messages at gmail.com.A total of 550 messages represent the entire corpus. It consist mails from different individuals and from different social sites.Mails with null body html documents are discarded while extracting it.We used pure java programming using the mail and mail.internet libraries of javamail API.The protocol we used was imaps.



4.2 EXTRACTING RAW TEXT MAILS IN FOLDER

We used imaps protocol to achieve the task.The javamail API comes handy while accomplishing the task.Steps required to extract the mails are as follows:

Step 1 - Define Protocol
Step 2 - Get a session instance to read email
Step 3 - Access emails through store class
Step 4 - Read Inbox

Step 1 : Define Protocol
props.setProperty("mail.store.protocol", "imaps")
First we need to define the protocol for processing emails.
SMTP - is the protocol to send email
POP3 - is the protocol to receive emails
IMAP- IMAP is an acronym for Internet Message Access Protocol. Its an advanced protocol for receiving messages.

This property takes two parameters (key, Value) key is "mail.store.protocol" and its value is "imaps" since we are going to read email protocol is defined as "imaps"

Step 2 : Get a session instance to read email

This property is used to get a session instance for reading email and its done as shown below in the code.

```
Session session = Session.getInstance(props, null);
```

Step 3 : Access emails through store class

Store - An abstract class that models a message store and its access protocol, for storing and retrieving messages. Store provides many common methods for naming stores, connecting to stores, and listening to connection events.

We will be making use of connect(String host,String user,String password) method to connect to specified host and get access to Inbox.

```
Store store = session.getStore();
store.connect("imap.gmail.com", "yourEmailId@gmail.com", "password");
Folder inbox = store.getFolder("INBOX");
```

Then we need to open the required folder in Read mode.

```
inbox.open(Folder.READ_ONLY);
```

Summary: *So far we have created a session and connected to gmail host with our username and password and got read access to Inbox.*

Step 4 : Read Inbox

We are almost done, now get access to your email using 'Message' class as shown below and typecast the content of the mail to Multipart to read the body of the email.

```
Message msg = inbox.getMessage(1);
```

Here 1 indicates the first email received in your inbox and getMessageCount() will give you the number of emails in your inbox. So read the latest email use,

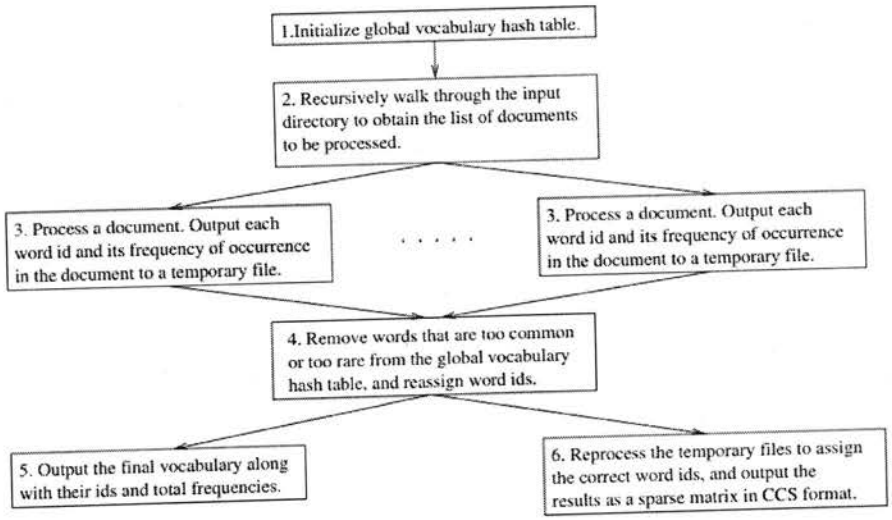
```
Message msg = inbox.getMessage(inbox.getMessageCount());
```

Once the mails are saved as raw text documents,the preprocessing technique starts.

4.3 PREPROCESSING TASK

The preprocessing task includes , tokenizing, stop word removal, identify key words, assign tfidf scheme,normalization,creating the vector space model.

The input to the preprocessing step is a data directory that contains all the documents to be processed. The documents may also be contained within subdirectories of the input directory. The output is the vector space model described in the above section, which can be represented as



a highly sparse word-by-document matrix. We store this sparse matrix by using the Compressed Column Storage(CCS) format. In this format, we record the value of each non-zero element, along with its row and column index. The column indices represent the input documents, the row indices represent ids of distinct words present in the document collection, and the non-zero entries in the matrix represent the frequencies of words in documents. Figure gives an outline of the preprocessing algorithm. The algorithm first initializes a global hash table. To resolve whether a word has been encountered previously, a local and a global hash table are used. Both these hash tables use words as keys and store the corresponding row indices and frequencies as values. As the names suggest, the global hash table keeps track of words and their occurrences in the entire document collection while the local hash table does so for just one document. After initializing the global hash table, the algorithm recursively walks through the input directory to obtain the list of documents to be processed. The preprocessing algorithm then creates several threads of computation. The purpose of each thread is to process a set of documents independently and output its results into temporary files. Details of this processing are given in Figure which we discuss in the next paragraph. After all the threads have finished, the global hash table is examined, and words that are too common or too rare are removed from the global hash table. Unique word ids are assigned to the words that still remain in the global hash table. The temporary files are then reloaded, the word ids are resolved and then the final vocabulary and word-by-document matrix are output.

Figure describes the various steps performed by each preprocessing thread. Two decisions warrant further explanation — the use of temporary files for storing the partial vector space model and the way in which the local and global hash tables are accessed. As mentioned in the last section, storing the partial vector space model in main memory would require a few gigabytes of main memory and is thus prohibitive for modern workstations. Hence to reduce main memory consumption we store the contents of the local hash table onto temporary files.

Since this only leads to local disk access, the resulting overhead is not substantial. The global hash table is accessed and modified by all processing threads and hence is a shared resource. In order to achieve maximum parallelism, we need to minimize the number of times the global hash table is locked and modified by each processing thread. We achieve this 8 by using a local hash table to process the entire document first, and then making a block access to the global hash table. This access involves resolving the word ids and possible modification of the global hash table, at which time this data structure needs to be locked. See Figure for details, especially step 5.

The basic idea is to represent each document as a vector of certain weighted word frequencies. In order to do so, the following parsing and extraction steps are needed. 1 Ignoring case, extract all unique words from the entire set of documents. 2 Eliminate non-content-bearing "stopwords" such as "a", "and", "the", etc. For sample lists of stopwords. 3 For each document, count the number of occurrences of each word.

fficient Clustering of Very Large Document Collections 5 4 Using heuristic or information-theoretic criteria, eliminate noncontent-bearing "high-frequency" and "low-frequency" words [SM83]. 5 After the above elimination, suppose w unique words remain. Assign a unique identifier between 1 and w to each remaining word, and a unique identifier between 1 and d to each document. The above steps outline a simple preprocessing scheme. In addition, one may extract word phrases such as "New York," and one may reduce each word to its "root" or "stem", thus eliminating plurals, tenses, prefixes, and suffixes .The above preprocessing yields the number of occurrences of word j in document i, say, fji , and the number of documents which contain the word j, say, dj . Using these counts, we can represent the i-th document as a w-dimensional vector xi as follows. For $1 \leq j \leq w$, set the j-th component of xi , to be the product of three terms $xji = tji \cdot gj \cdot si$, where tji is the term weighting component and depends only on fji , while gj is the global weighting component and depends on dj , and si is the normalization component for xi . Intuitively, tji captures the relative importance of a word in a document, while gj captures the overall importance of a word in the entire set of documents. The objective of such weighting schemes is to enhance discrimination between various document vectors for better retrieval effectiveness . There are many schemes for selecting the term, global, and normalization components, see for various possibilities. In this paper we use the popular tfn scheme known as normalized term frequency inverse document frequency. This scheme uses tji = fji , gj = log(d/dj) and si = ^{3}Pw j=1(tfidf) 2 ′−1/2 . Note that this normalization implies that kxik = 1, i.e., each document vector lies on the surface of the unit sphere in R||w||. Intuitively, the effect of normalization is to retain only the proportion of words occurring in a document. This ensures that documents dealing with the same subject matter (that is, using similar words), but differing in length lead to similar document vectors.

4.4 APPLYING KMEANS ALGORITHM

Given the vector space model, the document vectors may be represented by x1, x2, . . . , xd, where each xi ∈ Rw. Recall that w stands for the number of unique words in the vector space model and d is the total number of documents. A clustering of the document collection is its partitioning into the disjoint subsets π1, π2, ..., πk, i.e., [k j=1 πj = {x1, x2, ..., xd} and πj ∩ πl = φ, j 6= l.

The most important and challenging characterisitics of the vector space models that arise from text data are high dimensionality and sparsity. Typically, w is in the thousands and a sparsity of 99% is common. For purposes of efficiency, it is important that the clustering algorithm exploit the sparsity of the data while giving meaningful results at the same time. The spherical k-means algorithm satisfies both these properties and hence is our algorithm of choice. We now briefly formalize this algorithm highlighting its salient features. More details may be found in [DM01]. Any text clustering algorithm needs an objective notion of similarity between documents. A widely used measure of similarity is the cosine of the angle between two document vectors [FBY92, SM83]. Cosine similarity is easy to interpret and simple to compute for sparse vectors and has been used in other information retrieval applications, such as query retrieval. Based on cosine similarity, we can define the "goodness" or "coherence" of cluster πj

$$\sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i^T \mathbf{c}_j, \tag{4.1}$$

where each $\mathbf{x}_i$ is assumed to be normalized such that $\|\mathbf{x}_i\| = 1$ and $\mathbf{c}_j$ is the normalized centroid of cluster π_j,

$$\mathbf{c}_j = \frac{\sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i}{\| \sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i \|}.$$

By the Cauchy-Schwarz inequality,

$$\sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i^T \mathbf{z} \leq \sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i^T \mathbf{c}_j, \quad \forall \mathbf{z} \in R^w,$$

and thus the normalized centroid is the vector that is closest in cosine similarity (in an average sense) to all the document vectors in the cluster π_j. As a result, we also call the vector $\mathbf{c}_j$'s as *concept vectors*.

Aggregating (4.1) over all clusters, we can measure the goodness of any given partitioning $\{\pi_j\}_{j=1}^k$ using the following *objective function*:

$$Q(\{\pi_j\}_{j=1}^k) = \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i^T \mathbf{c}_j. \tag{4.2}$$

Intuitively, the objective function measures the combined coherence of all the k clusters. Having posed the objective function, we now present an algorithm that attempts to maximize its value.

1 Initialize clustering. Start with some initial partitioning of the document vectors, namely $\{\pi_j^{(0)}\}_{j=1}^k$. Let $\{\mathbf{c}_j^{(0)}\}_{j=1}^k$ be the concept vectors of the associated partitioning. Set the iteration count t to 0.

2 Re-assign document vectors. For each document vector $\mathbf{x}_i$, $1 \leq i \leq d$, do the following:

a. Compute $\mathbf{x}_i^T \mathbf{c}_l^{(t)}$ for all $l = 1, 2, \ldots, k$.

b. From among all $\mathbf{x}_i^T \mathbf{c}_l^{(t)}$ computed above, find $j = \arg\max_l \mathbf{x}_i^T \mathbf{c}_l^{(t)}$ (break ties arbitrarily if $\mathbf{x}_i$ has largest cosine similarity with more than one concept vector).

This induces the new partitioning

$$\pi_j^{(t+1)} = \{\mathbf{x}_i : j = \arg\max_l \mathbf{x}_i^T \mathbf{c}_l^{(t)}\}, \quad 1 \leq j \leq k.$$

3 Update concept vectors. Compute the concept vectors corresponding to the new partitioning:

$$\mathbf{s}_j = \sum_{\mathbf{x}_i \in \pi_j} \mathbf{x}_i, \qquad \mathbf{c}_j^{(t+1)} = \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}, \quad 1 \leq j \leq k.$$

4 Check the stopping criterion. If the stopping criterion is satisfied, then exit. Otherwise increase t by 1 and go to step 2 above.

After applying the kmeans algorithm the documents are clustered as the predefined labels .Documents which are not assigned to any of the labels are resided at OTHERS folders.This situation typically takes place due to the changes of the centroids after each assignments.The classified documents can be called as trained documents and hence we apply the K nearest neighbor algorithm for futher clustering of the unassigned documents.

4.5 APPLYING K NEAREST NEIGHBOUR ALGORITHM

Until now we have classified email documents to the predefined labels.The unassigned documents are resided at an extra label called OTHERS.So the algorithm works on the OTHERS

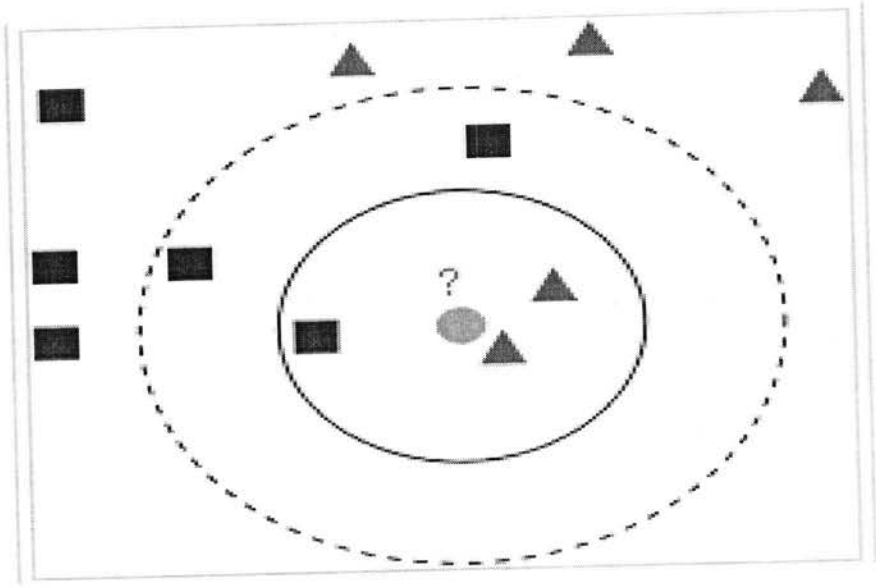
document vectors once again to assign to the predefined labels.It has seen that the folder looses thirty to fourty of the documents that is this amount of documents are assigned to the predefined labels.The following steps shows how the algorithm works:

1)Go through the unassigned documents from the OTHERS folder one by one and compute the distance between all the other documents from the training set that is all the documents that are resided with in the clusters itself.

2)List top K documents which has the least distance metric between them

3)Take the most least pair from the above process and assign the unlabeled document to the cluster of the paired one.

After KNN finishes the folder named OTHERS will still consist of some vectors which has nothing to do with the predefined labels.The documents are totally different from the given predefined labels.



5 RESULT:

Our dataset was my personal email messages from gmail.A total of 550 emails are prepared as text documents for the task.There might different frequent categories in the emails depending on the users.Emails include mails from individuals as well as from social sites.The user himself can only know how many categories of mails ends up in his inbox.In my case for the time being I have the following categories

1)facebook

2)glassdoor

3)howzat_cricket

4)sadika

5)banglalion

6)shamsul

7)Mumit

8)void

9)piazza

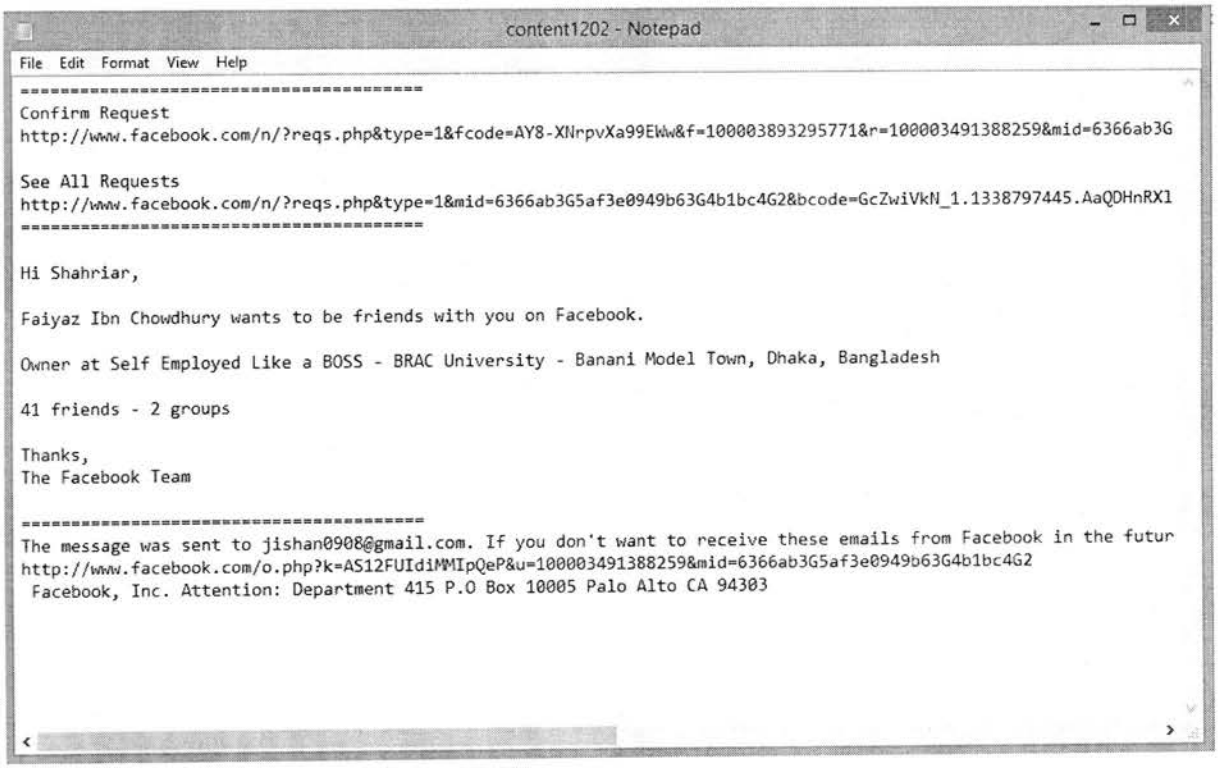
10)everjobs

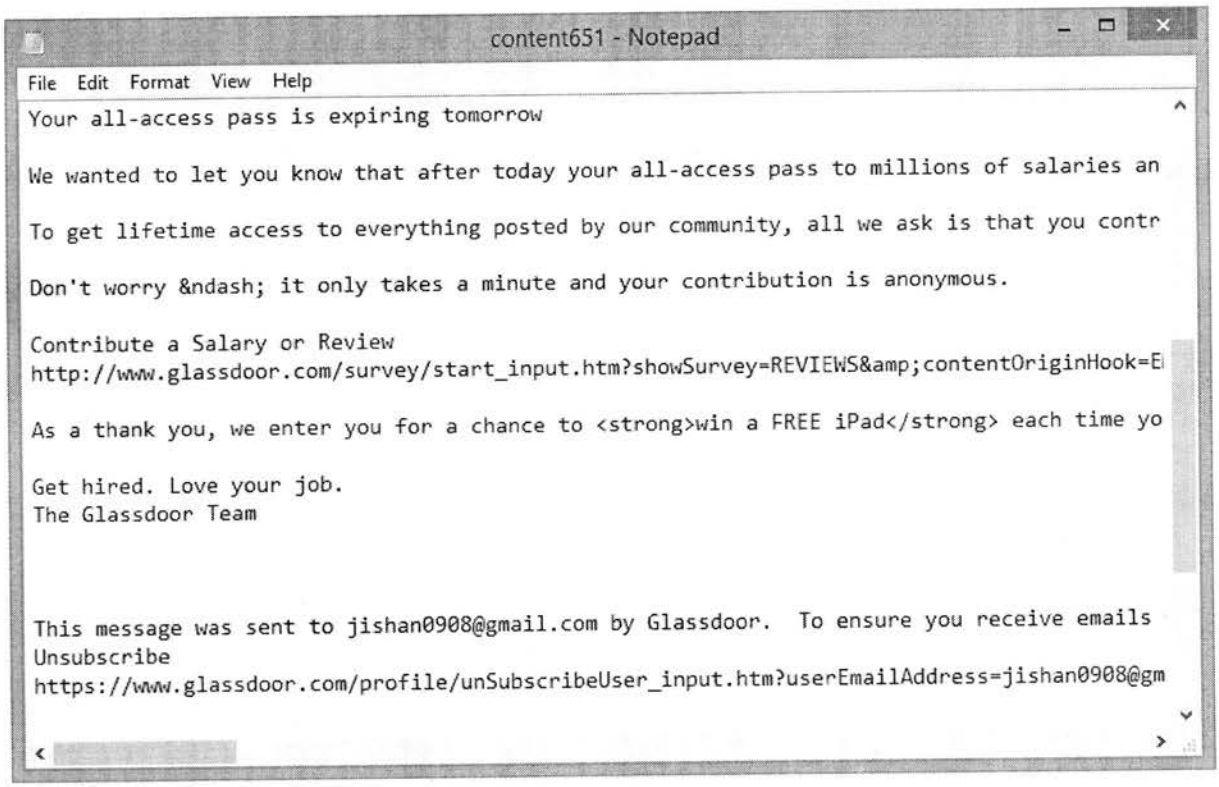
Therefore the very first task was to choose initial centroids on the basis of the predefined labels as given above.So centroid 1 will be of facebook,centroid 2 will be of glassdoor and so on

Initial result:

```
ENTERED FinalCluster class
initially    content1202.txt
initially    content651.txt
initially    content368.txt
initially    content114.txt
initially    content0.txt
initially    content131.txt
initially    content138.txt
initially    content11.txt
initially    content346.txt
initially    content716.txt
```

As shown above content1202 represent a document regarding facebook ,content654 is for glassdoor and so on.

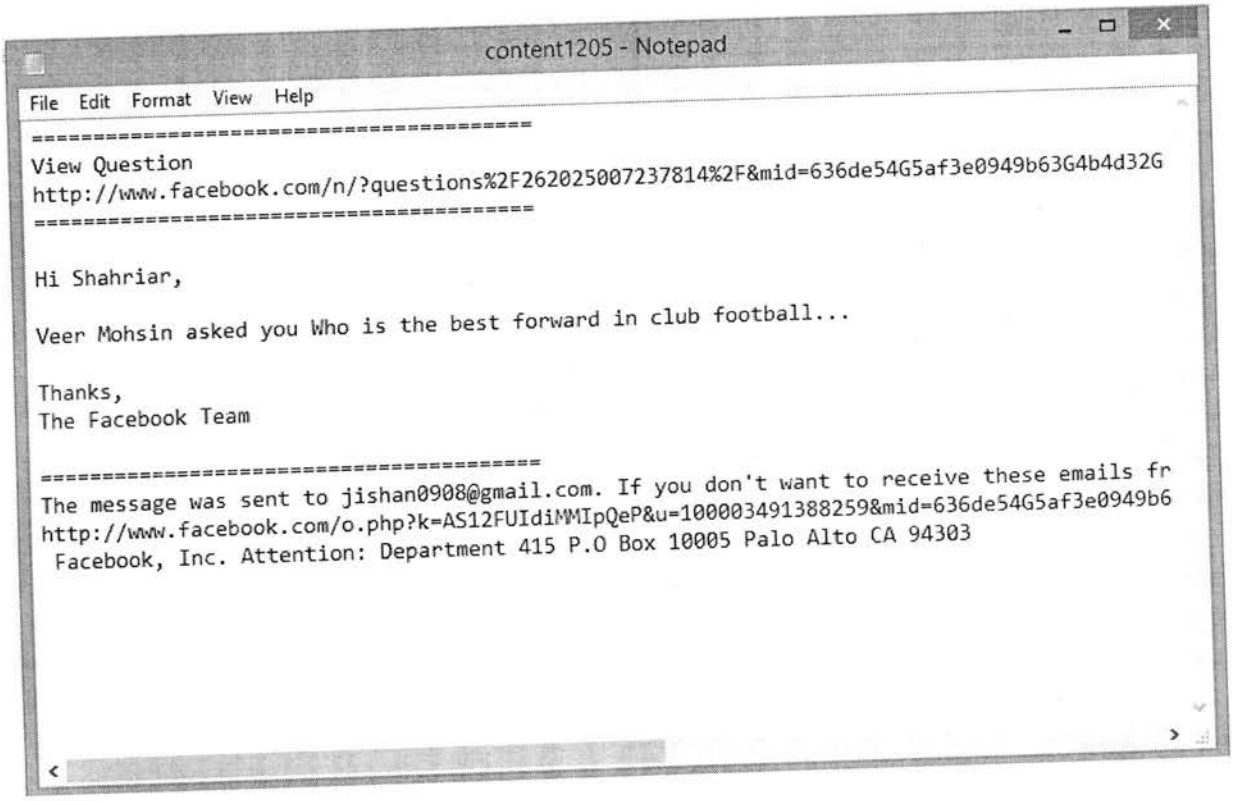

content1202 - Notepad
File Edit Format View Help
==========
Confirm Request
http://www.facebook.com/n/?reqs.php&type=1&fcode=AY8-XNrpvXa99EWw&f=100003893295771&r=100003491388259&mid=6366ab3G
See All Requests
http://www.facebook.com/n/?reqs.php&type=1&mid=6366ab3G5af3e0949b63G4b1bc4G2&bcode=GcZwiVkN_1.1338797445.AaQDHnRX1
==========
Hi Shahriar,
Faiyaz Ibn Chowdhury wants to be friends with you on Facebook.
Owner at Self Employed Like a BOSS - BRAC University - Banani Model Town, Dhaka, Bangladesh
41 friends - 2 groups
Thanks,
The Facebook Team
==========
The message was sent to jishan0908@gmail.com. If you don't want to receive these emails from Facebook in the futur
http://www.facebook.com/o.php?k=AS12FUIdiMMIpQeP&u=100003491388259&mid=6366ab3G5af3e0949b63G4b1bc4G2
Facebook, Inc. Attention: Department 415 P.O Box 10005 Palo Alto CA 94303

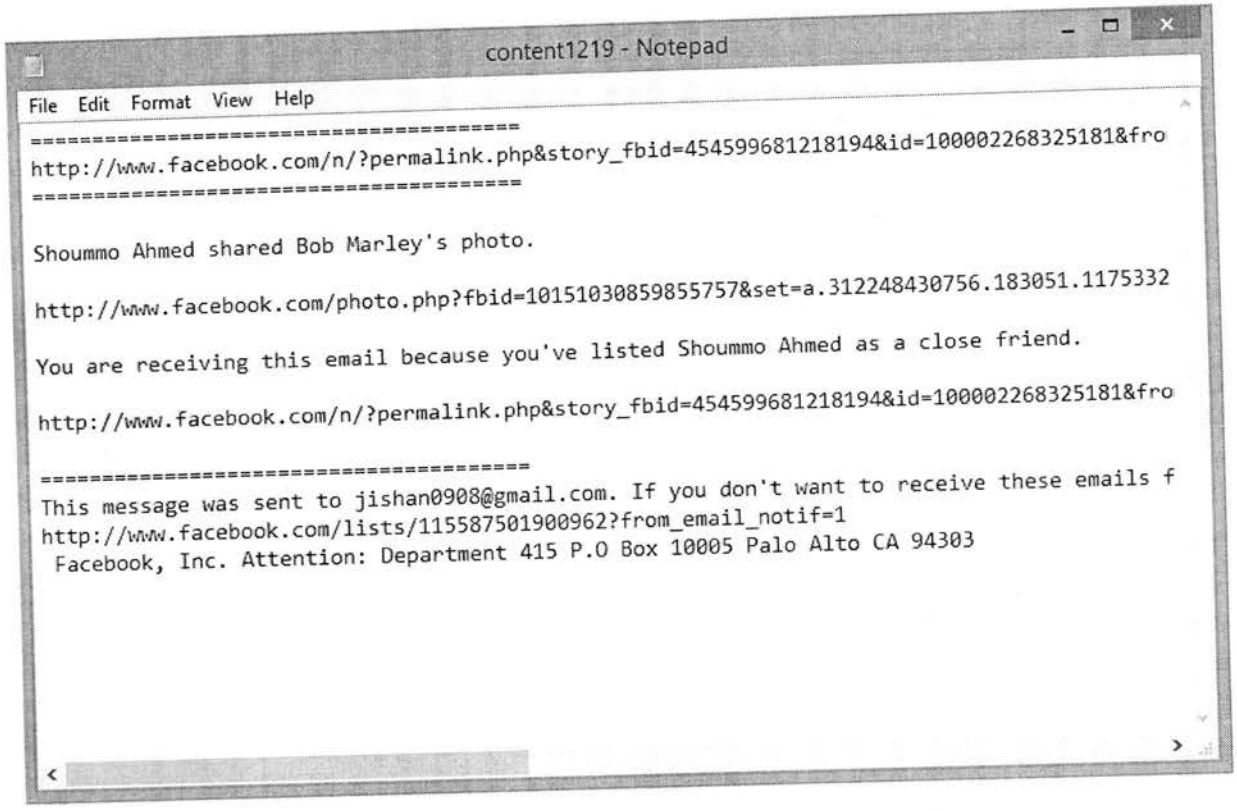

content651 - Notepad
File Edit Format View Help
Your all-access pass is expiring tomorrow
We wanted to let you know that after today your all-access pass to millions of salaries an
To get lifetime access to everything posted by our community, all we ask is that you contr
Don't worry – it only takes a minute and your contribution is anonymous.
Contribute a Salary or Review
http://www.glassdoor.com/survey/start_input.htm?showSurvey=REVIEWS&contentOriginHook=E
As a thank you, we enter you for a chance to win a FREE iPad each time yo
Get hired. Love your job.
The Glassdoor Team
This message was sent to jishan0908@gmail.com by Glassdoor. To ensure you receive emails
Unsubscribe
https://www.glassdoor.com/profile/unSubscribeUser_input.htm?userEmailAddress=jishan0908@gm

Once centroids are chosen then Kmeans and the K nearest neighbor algorithm is started.Below are some screenshots of the result after the entire process ends

Clustering result of facebook.41 assignments out of 58 mails.Rest mails are resided in the OTHERS cluster.content1205 and 1219 showing the relevance with the PREDEFINED labels

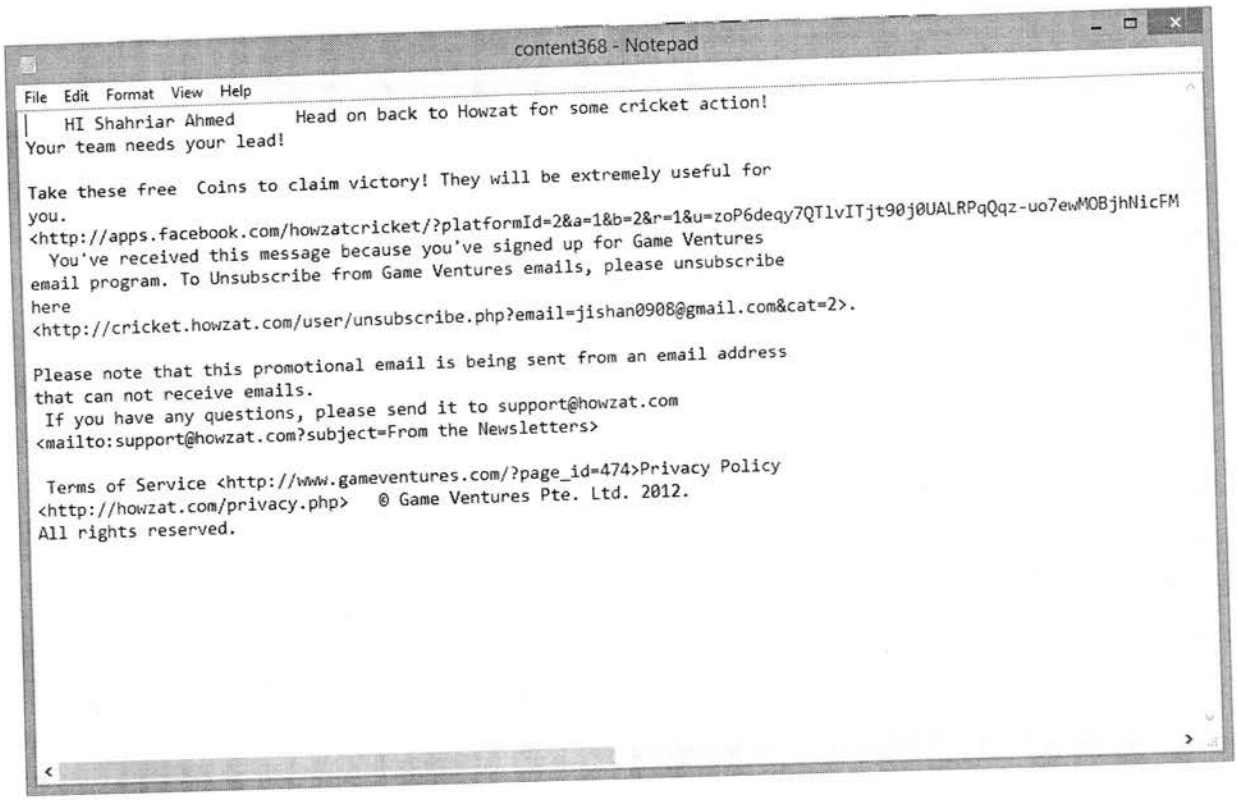
```
tester class PRINTSTATUS mehtod()
cluster number  0     facebook   41
content1202.txt
content1204.txt
content1206.txt
content1205.txt
content1207.txt
content1212.txt
content1211.txt
content1216.txt
content1219.txt
content1218.txt
content1223.txt
content1225.txt
content1233.txt
content1238.txt
content1237.txt
content1247.txt
content1258.txt
content1257.txt
content1255.txt
content1254.txt
content1261.txt
content1260.txt
content1263.txt
content1262.txt
content869.txt
content865.txt
content868.txt
content867.txt
content1203.txt
content1210.txt
```

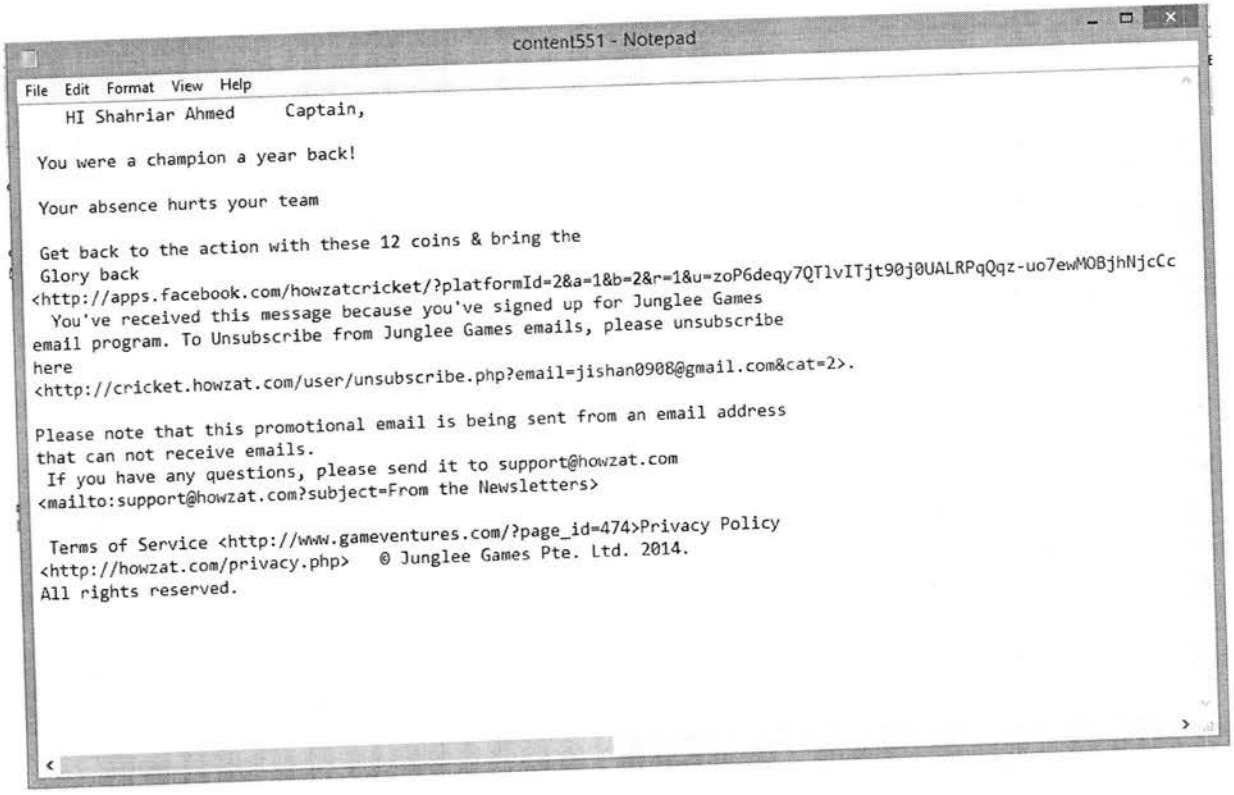

content1205 - Notepad
File Edit Format View Help
==========
View Question
http://www.facebook.com/n/?questions%2F262025007237814%2F&mid=636de54G5af3e0949b63G4b4d32G
==========
Hi Shahriar,
Veer Mohsin asked you Who is the best forward in club football...
Thanks,
The Facebook Team
==========
The message was sent to jishan0908@gmail.com. If you don't want to receive these emails fr
http://www.facebook.com/o.php?k=AS12FUIdiMMIpQeP&u=100003491388259&mid=636de54G5af3e0949b6
Facebook, Inc. Attention: Department 415 P.O Box 10005 Palo Alto CA 94303


content1219 - Notepad
File Edit Format View Help
==========
http://www.facebook.com/n/?permalink.php&story_fbid=4545996812181948id=100002268325181&fro
==========
Shoummo Ahmed shared Bob Marley's photo.
http://www.facebook.com/photo.php?fbid=10151030859855757&set=a.312248430756.183051.1175332
You are receiving this email because you've listed Shoummo Ahmed as a close friend.
http://www.facebook.com/n/?permalink.php&story_fbid=4545996812181948id=100002268325181&fro
==========
This message was sent to jishan0908@gmail.com. If you don't want to receive these emails f
http://www.facebook.com/lists/115587501900962?from_email_notif=1
Facebook, Inc. Attention: Department 415 P.O Box 10005 Palo Alto CA 94303

Clustering result of howzat.11assignments out of 19 mails.Rest mails are resided in the OTHERS cluster.content 368 and 551 showing the relevance with the PREDEFINED labels

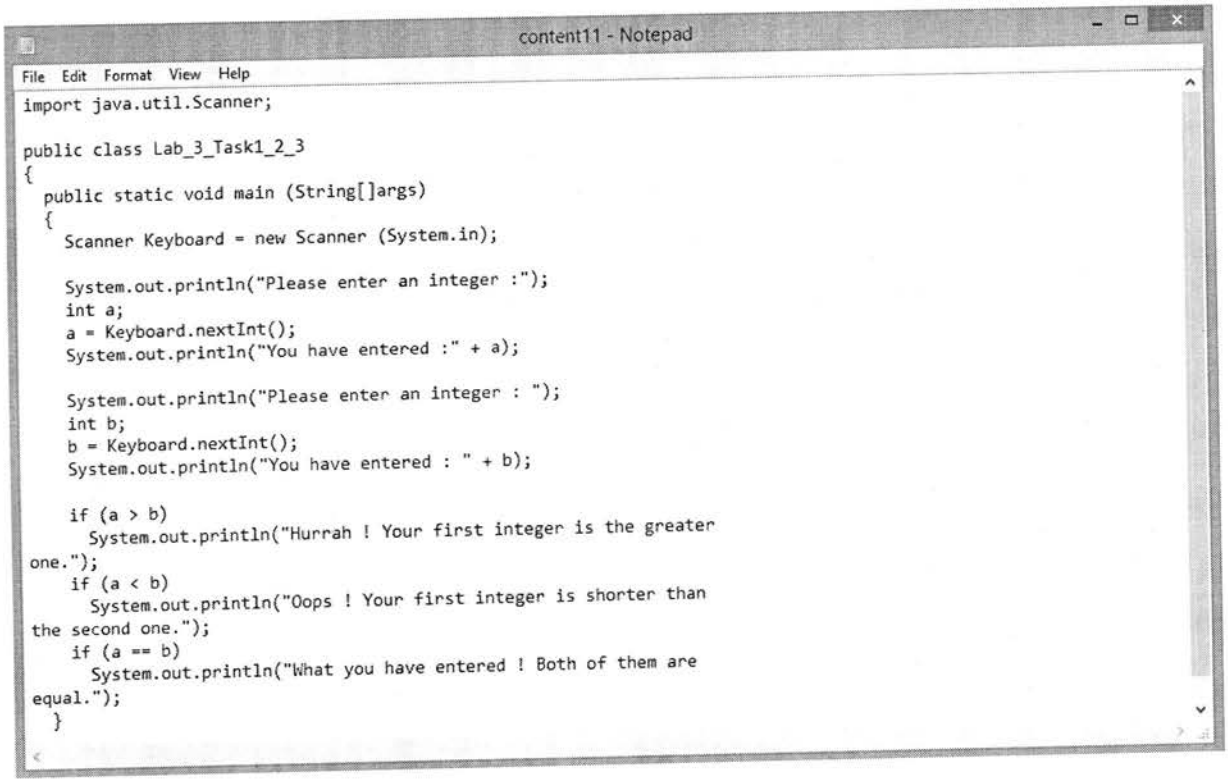
```
cluster number  2     howzat   11
content368.txt
content388.txt
content391.txt
content402.txt
content408.txt
content396.txt
content468.txt
content448.txt
content452.txt
content520.txt
content551.txt
```

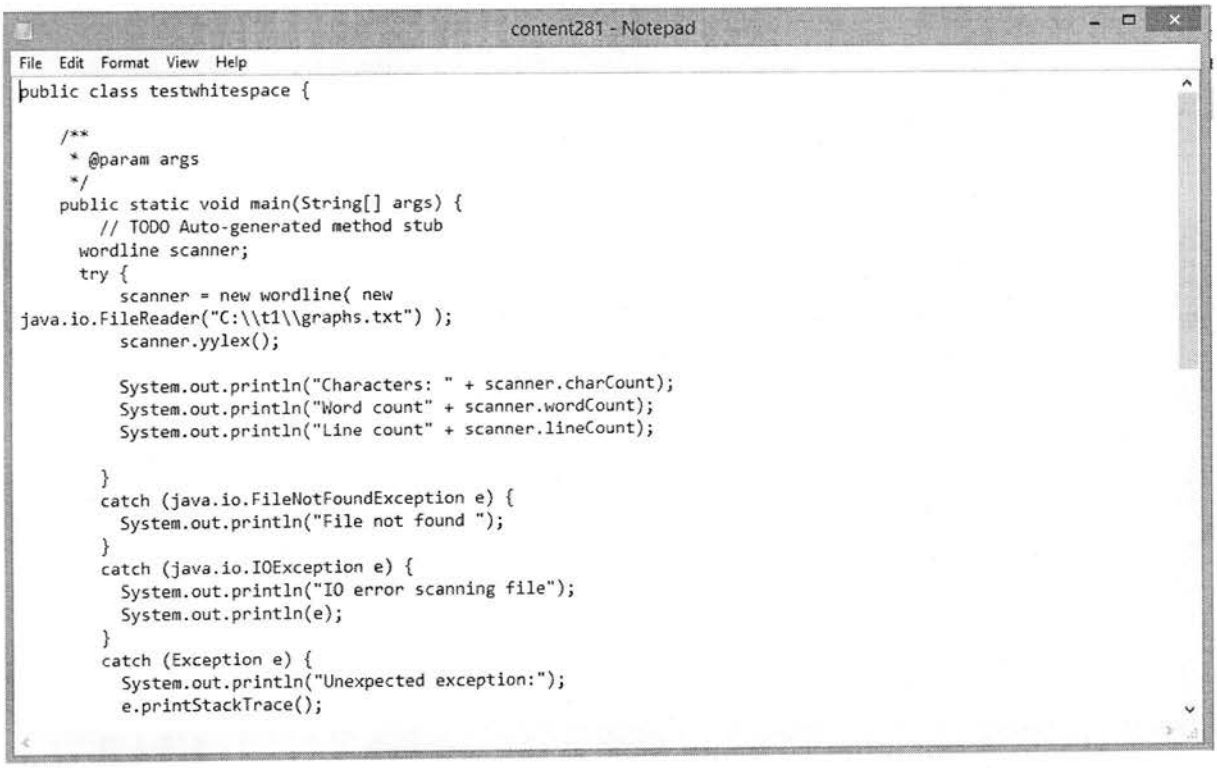




Clustering result of void.Programming related mails .11assignments out of 23mails.Rest mails are resided in the OTHERS cluster.content 11 and 281showing the relevance with the PREDEFINED labels

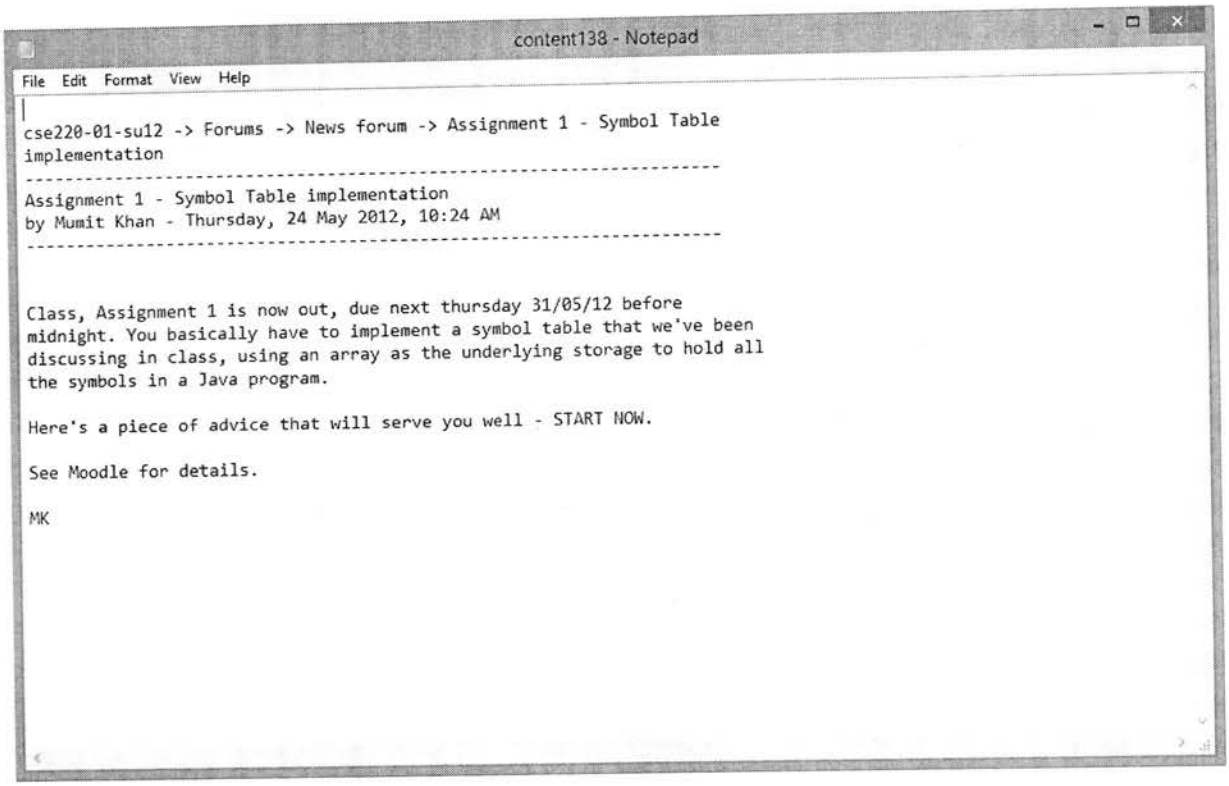
```
cluster number  7     void   11
content11.txt
content12.txt
content20.txt
content281.txt
content32.txt
content332.txt
content37.txt
content68.txt
content69.txt
content75.txt
content279.txt
```

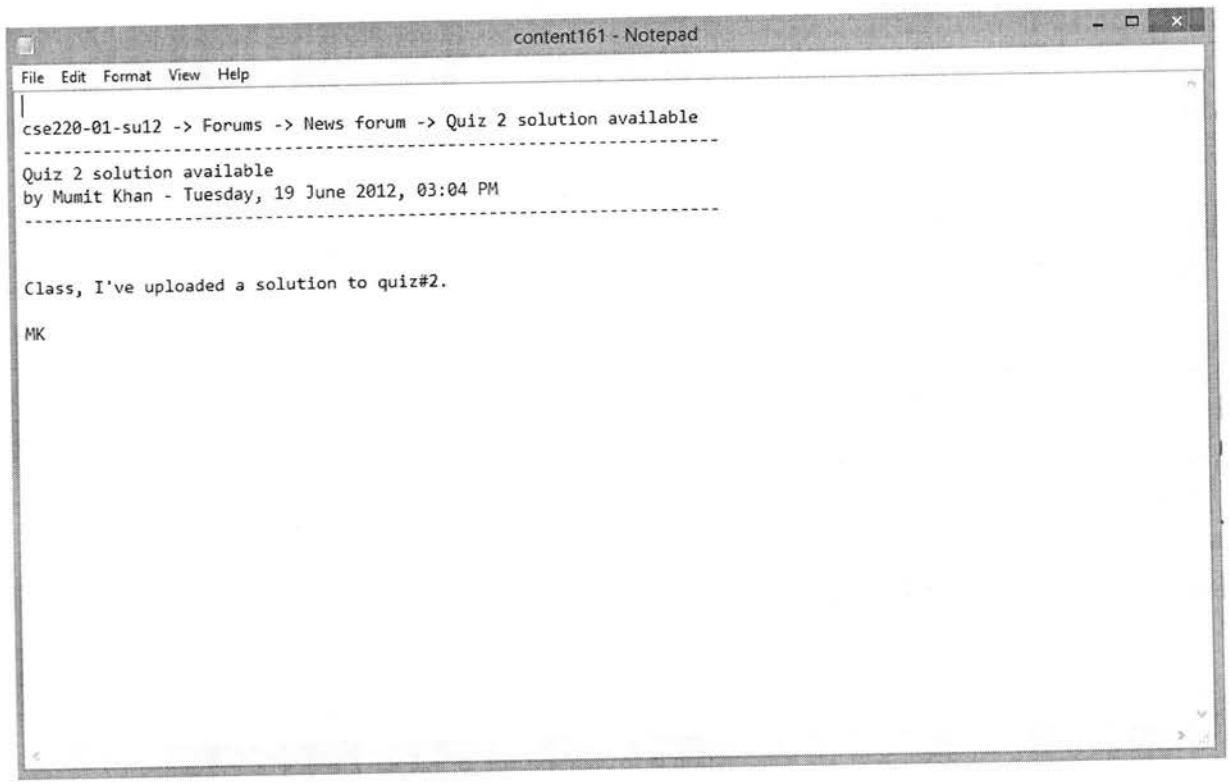

content11 - Notepad
File Edit Format View Help
import java.util.Scanner;
public class Lab_3_Task1_2_3
{
public static void main (String[]args)
{
Scanner Keyboard = new Scanner (System.in);
System.out.println("Please enter an integer :");
int a;
a = Keyboard.nextInt();
System.out.println("You have entered :" + a);
System.out.println("Please enter an integer : ");
int b;
b = Keyboard.nextInt();
System.out.println("You have entered : " + b);
if (a > b)
System.out.println("Hurrah ! Your first integer is the greater
one.");
if (a < b)
System.out.println("Oops ! Your first integer is shorter than
the second one.");
if (a == b)
System.out.println("What you have entered ! Both of them are
equal.");
}



Clustering result of Mumit.Mails from an individual . 22assignments out of 30mails.Rest mails are resided in the OTHERS cluster.content 138 and 161 showing the relevance with the PREDEFINED labels

```
cluster number  6     mumit   22
content138.txt
content140.txt
content145.txt
content146.txt
content147.txt
content148.txt
content141.txt
content142.txt
content143.txt
content144.txt
content153.txt
content152.txt
content156.txt
content154.txt
content149.txt
content151.txt
content150.txt
content163.txt
content158.txt
content157.txt
content162.txt
content161.txt
```




content161 - Notepad
File Edit Format View Help
cse220-01-su12 -> Forums -> News forum -> Quiz 2 solution available
Quiz 2 solution available
by Mumit Khan - Tuesday, 19 June 2012, 03:04 PM
Class, I've uploaded a solution to quiz#2.
MK

6)LIMITATIONS

In our thesis work we had few limitations .Both in the preprocessing task as well as in the post processing part. Many of the implementations regarding these sort of work also include a part of natural language processing.It works in the preprocessing phase .It includes stemming,lemmatization etc. We did not implemented that part, because it itself introduces a whole new topic and it's a group task. Vectors might get a little shorter if we used the NLP subpart.

Our second limitation is that we used user given PREDEFINED labels for choosing the initial centroids. Work load is reduced due to the labels. We did not implement any algorithm for choosing the initial centroids. Random selection ends up with bad results.

7 References

[BGG+98] D. Boley, M. Gini, R. Gross, E.-H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Document categorization and query generation on the World Wide Web using WebACE. AI Review, 1998.

[Cal99] Brent Callaghan. NFS Illustrated. Addison-Wesley, 1999.

[CKPT92] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In ACM SIGIR, 1992.

[DGL89] I. Duff, R. Grimes, and J. Lewis. Sparse matrix test problems. ACM Trans Math Soft, pages 1–14, 1989.

[DH73] R. O. Duda and P. E. Hart. Pattern Classification and Scene Analysis. Wiley, 1973.

[DM01] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. Machine Learning, 42(1):143–175, January 2001. Also appears as IBM Research Report RJ 10147, July 1999.

[FBY92] W. B. Frakes and R. Baeza-Yates. Information Retrieval: Data Structures and Algorithms. Prentice Hall, Englewood Cliffs, New Jersey, 1992.

[GJW82] M. R. Garey, D. S. Johnson, and H. S. Witsenhausen. The complexity of the generalized Lloyd-Max problem. IEEE Trans. Inform. Theory, 28(2):255–256, 1982. REFERENCES 25

[Hea78] J. Heaps. Information Retrieval - Computational and Theoretical Aspects. Academic Press, 1978. [Kol97] T. G. Kolda. Limited-Memory Matrix Methods with Applications. PhD thesis, The Applied Mathematics Program, University of Maryland, College Park, Mayland, 1997.

[KPR98] Jon Kleinberg, C. H. Papadimitriou, and P. Raghavan. A microeconomic view of data mining. Data Mining and Knowledge Discovery, 2(4):311–324, December 1998.

[Mit97] Tom M. Mitchell. Machine Learning. McGraw-Hill, 1997.

[MS96] D. Musser and A. Saini. STL Tutorial and Reference Guide. Addison-Wesley, 1996.

[NBF96] Bradford Nichols, Bick Buttlar, and Jackie Proulx Farrell. Pthreads Programming. O'Reilly & Associates, Inc., 981 Chestnut Street, Newton, MA 02164, USA, 1996.

[Pax96] Vern Paxson. Flex user manual, November 1996.

[Ras92] E. Rasmussen. Clustering algorithms. In William B. Frakes and Ricardo Baeza-Yates, editors, Information Retrieval: Data Structures and Algorithms, pages 419–442. Prentice Hall, Englewood Cliffs, New Jersey, 1992.

[SB88] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. Information Processing & Management, 4(5):513–523, 1988.

[SGM00] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In Proceedings of the AAAI2000 Workshop on Artificial Intelligence for Web Search, pages 58–64, Austin, Texas, July 2000. AAAI/MIT Press.

[SM83] G. Salton and M. J. McGill. Introduction to Modern Retrieval. McGraw-Hill Book Company, 1983. [SS97] H. Schutze " and C. Silverstein. Projections for efficient document clustering. In ACM SIGIR, 1997. [Wil88] P. Willet. Recent trends in hierarchic document clustering: a critical review. Information Processing & Management, 24(5):577–597, 1988.

[ZE98] O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In ACM SIGIR, 1998.

[Zip49] G. K. Zipf. Human Behavior and the Principle of Least Effort. Addison Wesley, Reading, MA, 1949.