Indian Statistical Institute
Kolkata

M.Tech. (Computer Science) Dissertation

Hierarchical Approach to Document Classification of 20 Newsgroup Dataset

A dissertation submitted in partial fulfillment of the requirements
for the award of Master of Technology
in
Computer Science

Author:
Kanishka Dhamija
Roll No: CS1402

Supervisor:
Prof. Swapan Kumar Parui
Computer Vision and Pattern
Recognition Unit

CERTIFICATE

M.TECH(CS) DISSERTATION COMPLETION CERTIFICATE
Student : Kanishka Dhamija (CS 1402)
Topic : Hierarchical Approach to Document Classification of 20 Newsgroup Dataset
Supervisor : Prof. Swapan Kumar Parui

This is to certify that the dissertation titled **Hierarchical Approach to Document Classification of 20 Newsgroup Dataset** submitted by **Kanishka Dhamija** in partial fulfillment for the award of the degree of Master of Technology is a bonafide record of work carried out by him under our supervision. The dissertation has fulfilled all the requirements as per the regulations of this Institute and, in our opinion, has reached the standard needed for submission. The results embodied in this dissertation have not been submitted to any other university for the award of any degree or diploma.

Prof. Swapan Kumar Parui
Computer Vision and Pattern Recognition Unit
Indian Statistical Institute

Abstract

The aim of the dissertation is to come up with a good algorithm that will help classify the documents of the 20 newsgroup data set to it's proper classes. Different methods are applied using the vector representation of documents (number of times a uni-gram occurs in a document) to come up with a method that gives best accuracy after classification. Hierarchical structure for classification was followed and different methods were experimented with to see which one gives the best accuracy. Different ways to detect outliers in the training set were also applied and these outliers were removed from the training set to improve accuracy.

Acknowledgment

I would like to express my sincere gratitude to my guide, Prof. Swapan Kumar Parui, for his continuous support, understanding, patience, motivation, and immense knowledge. His guidance helped me at all time during the research and writing of this dissertation. I can not imagine this being possible without his guidance.

I would also like to thank my parents and my sister for their support and keeping me motivated whenever I felt low and for believing in me.

Lastly, I would like to thank my friends who have helped me in every way possible whenever I needed support and for morally boasting me.

Contents

List of Algorithms

List of Tables

Chapter 1

Introduction

1.1 Document Classification

The objective of document classification is to group the documents that are similar to each other into groups called classes. This would help to reduce the detail and diversity of data. Document categorization or document classification may be viewed as assigning documents or parts of documents in a predefined set of categories [1]. The classes can also be referred to as labels and the document which has a class assigned to it is called a labelled document. Generally, for data classification, there is a training set that is a set labelled documents whose label remains unchanged over time. Document classification simply means to assign a label (class) to a document whose label (class) is not known. The categories, classes or labels are already fixed in case of document classification. In the domain of text mining document categorization also involves the preliminary process of automatically learning categorization patterns so that the categorization of new (uncategorized) documents is straightforward [1].

The main goal of Document Classification is to derive methods for the classification of natural language text. The objective is to automatically derive methods that, given a set of training documents $D = d_1, d_2, \ldots, d_r$ with known categories $C = c_1, c_2, \ldots, c_q$ and a new document q, which is usually called the query, will predict the querys category, that is, will associate with q one or more of the categories in C [2].

The methods that are used in Document Classification are generally the same that are used in the more general area of Information Retrieval, where the goal is to find documents or passages within documents that are relevant, or related to, a particular query. By considering the document to classify as the query and the classes of the documents that are retrieved as the possible classes for the query, a method developed for Information Retrieval can be used for Document Classification tasks. Document Classification techniques are necessary to find relevant information in many different tasks that deal with large quantities of information in text form. Some of the most common tasks where these techniques are applied are finding answers to similar questions that have been answered before, classifying news by subject or newsgroup, sorting spam from legitimate e-mail messages, finding Internet pages on a given subject, among others. In each case, the goal is to assign the appropriate category or label to each document that needs to be classified [2].

1.2 Document Clustering

Clustering is the grouping of similar objects. In contrast to classification, clustering is an unsupervised learning procedure that means the labels of the documents are not known in advance and the number of possible labels are also not known. Cluster analyses are targeted on exploring similarities in the contents of the documents and arranging them in groups according to these properties. They are not based on a predefined structure of knowledge: Neither classes are predefined nor examples are given that show what types of relationships are expected between the objects [2]. Therefore, there is a need for method that can tell the similarities or dissimilarities between the data and an extend or a threshold that can indicate whether the similarity is sufficient or not to group the document under the same cluster. The objective is to divide the given dataset (that is the collection of documents) into different groups or clusters such that the similarities of all pair of documents belonging to the same cluster are high and the dissimilarities of all pair of documents belonging to different clusters are high.

1.3 Confusion Matrix

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm. Each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class (or vice-versa).

1.4 Overview

The objective of the dissertation is to come up with a good algorithm that will help classify the documents of the 20 newsgroup data set to it's proper classes. The structure of the report is given in this section.

Chapter 2 of this dissertation describes the 20 newsgroup dataset that has been used in the dissertation. It gives the overview of the dataset and also provides some details about the format of the dataset that has been used for implementation.

Chapter 3 tells about the different notation that have been used in the dissertation and the different similarity/distance measures used and their implementations.

Chapter 4 gives a brief description of k-NN classifier, it's algorithm and some results of applying k-NN on the 20 newsgroup dataset.

Chapter 5 gives the description of k-Means clustering and also explains different methods used after k-Means clustering to classify the documents. The different algorithms used are given in this chapter and their results are also displayed in the chapter.

Chapter 6 gives a description of hierarchical agglomerative clusters and the different types of hierarchical agglomerative clustering. The hierachical agglomerative clustering had been applied on different classes individually and the results are displayed in the chapter.

Chapter 7 explains the concept of Naive Bayes classifier. It also talks about different modifications done on Naive Bayes classifier to get better accuracy that is the use of 1-NN after application of Naive Bayes classifier. All algorithms used and their results are given in the chapter.

Chapter 2

20 Newsgroup Dataset

2.1 Introduction

The 20 Newsgroups dataset is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups.The data is organized into 20 different newsgroups, each corresponding to a different topic. Some of the newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian) [7]. The list of the 20 newsgroups, partitioned (more or less) according to subject matter is given in table 2.1.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Table 2.1: 20 Newsgroup Dataset Categories

2.2 The Dataset Used

2.2.1 Classes

The dataset that is used consists of 18846 documents (11314 training and 7532 testing) from 20 different classes. The classes are assigned a class number for this dissertation and will be referred to by it's number. The class labels corresponding to different class numbers are given in the table 2.2.

Class Number	Class Label
Class 1	alt.atheism
Class 2	comp.graphics
Class 3	comp.os.ms-windows.misc
Class 4	comp.sys.ibm.pc.hardware
Class 5	comp.sys.mac.hardware
Class 6	comp.windows.x
Class 7	misc.forsale
Class 8	rec.autos
Class 9	rec.motorcycles
Class 10	rec.sport.baseball
Class 11	rec.sport.hockey
Class 12	sci.crypt
Class 13	sci.electronics
Class 14	sci.med
Class 15	sci.space
Class 16	soc.religion.christian
Class 17	talk.politics.guns
Class 18	talk.politics.mideast
Class 19	talk.politics.misc
Class 20	talk.religion.misc

Table 2.2: Class Labels for Different Class Numbers

2.2.2 Documents

As mentioned earlier there are 18846 documents. 11314 documents belong to the training set and the rest 7532 belong to the testing set. The distribution of documents over different classes is given in table 2.3.

Class Number	Number of Train Documents	Number of Test Documents	Total Number of Documents
Class 1	480	319	799
Class 2	584	389	973
Class 3	591	394	985
Class 4	590	392	982
Class 5	578	385	963
Class 6	593	395	988
Class 7	585	390	975
Class 8	594	396	990
Class 9	598	398	996
Class 10	597	397	994
Class 11	600	399	999
Class 12	595	396	991
Class 13	591	393	984
Class 14	594	396	990
Class 15	593	394	987
Class 16	599	398	997
Class 17	546	364	910
Class 18	564	376	940
Class 19	465	310	775
Class 20	377	251	628
Total	11314	7532	18846

Table 2.3: Distribution of Documents over different Classes

For each document, the different words appearing in that document along with their frequencies (the number of times they appeared in the document) was given. The data was already preprocessed (removal of stop words, stemming and so on had already been applied on the data-set). For the experiments, each document was considered as a vector, each dimension representing a different term and value corresponding to number of times that term appears in the document. Although there were 90,288 different words in the vocabulary, each document contained very few words compared to the size of the vocabulary and it was better to store the term-ids and their frequencies for each document rather than storing it as a vector and unnecessarily including a lot of zero values.

2.2.3 Vocabulary

The training data-set consists of 90,288 words. The table 2.4 shows the number of uni-grams per class considering only the documents from the training set.

Class Number	Number of Unique Terms
Class 1	6435
Class 2	8307
Class 3	34522
Class 4	6777
Class 5	6053
Class 6	9464
Class 7	7222
Class 8	6817
Class 9	7043
Class 10	6054
Class 11	7635
Class 12	9224
Class 13	6860
Class 14	9454
Class 15	9088
Class 16	7657
Class 17	10305
Class 18	9537
Class 19	7564
Class 20	6699

Table 2.4: Number of Unigrams per Class

The observation given in table 2.4 can be visualised from the bar chart given in figure 2.1.

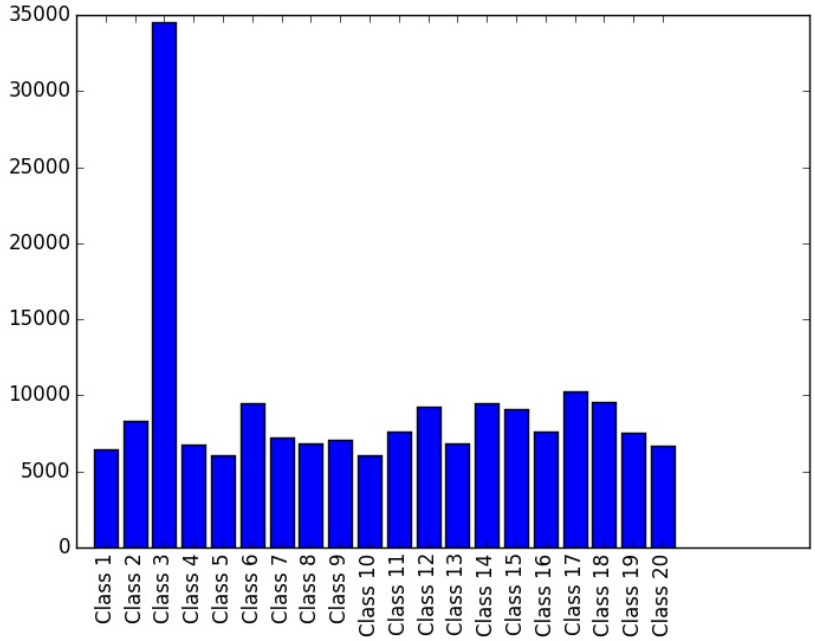

Figure 2.1: Distributions of words across different classes

Each term (unigram) has a unique id which is an integer assigned to it and will be known by that id.

Chapter 3

Notations and Measures

3.1 Some Notations

3.1.1 Class

A class $class_i$ is a list of documents and can be represented by:

$$class_i = \{doc_{i,1}, doc_{i,2}, \ldots, doc_{i,m_i}\}$$

m_i is the number of documents in the i^{th} class $class_i$ and it would be mentioned if only training documents of that class $class_i$ are considered or only test documents of that class $class_i$ is considered or both.

3.1.2 Document

Even though we are logically using a document as a vector of size $N = 90288$, each dimension representing a different unigram (sorted in order according to its id which is an integer value), we will not be representing it as a vector for implementation as that would require a lot of 0 values to be stored and will in turn increase the space that stores the document information and also increase time when performing some actions on the documents.

Hence, a document, doc_i, is represent as:

$$doc_i = \{termid_{i,1}, termid_{i,2}, \ldots, termid_{i,n_i}, freq_{i,1}, freq_{i,2}, \ldots, freq_{i,n_i}\}$$

where n_i is the number of unigrams in the document, $termid_{i,j}$ is the term id of a unigram present in doc_i arranged in accenting order i.e. $termid_{i,1} > termid_{i,2} > \ldots > termid_{i,n}$. The $freq_{i,j}$ represents the number of times unigram corresponding to $termid_{i,j}$ occurs in the document doc_i.

3.1.3 Normalised Document

A document doc_i can be normalised by changing the frequency of each unigram by normalised frequency given by:

$$normfreq_{i,j} = \frac{freq_{i,j}}{\sum_{j=1}^{n_i} freq_{i,j}} \mid 1 \leq j \leq n_i$$

The normalised document $normdoc_i$ will satisfy the following property:

$$\sum_{j=1}^{n_i} normfreq_{i,j} = 1$$

The notation for normalised document will be:

$$normdoc_i = \{termid_{i,1}, termid_{i,2}, \ldots, termid_{i,n_i}, normfreq_{i,1}, normfreq_{i,2}, \ldots, normfreq_{i,n_i}\}$$

3.2 Distance/Similarity Measures

A similarity/distance measure reflects the degree of closeness or separation of the target documents and should correspond to the characteristics that are believed to distinguish the documents embedded in the data. Choosing an appropriate similarity measure is crucial for classification or cluster analysis.

Since a document can be looked as a vector in N-dimensional ($N = 90288$) space, a metric on this space can be defined and must satisfy the following four conditions :

Let doc_i and doc_j be any two documents in a set and $dist(doc_i, doc_j)$ be the distance between x and y.

1. The distance between any two pdocemnts must be non-negative, that is, $dist(doc_i, doc_j) \geq 0$.
2. The distance between two documents must be zero if and only if the two objects are considered identical, that is, $dist(doc_i, doc_j) = 0$ if and only if $doc_i = doc_j$.
3. Distance must be symmetric, that is, distance from doc_i to doc_j is the same as the distance from doc_i to doc_j, i.e. $dist(doc_i, doc_j) = dist(doc_i, doc_j)$.
4. . The measure must satisfy the triangle inequality, which is $dist(doc_i, doc_j) \leq dist(doc_i, doc_j) + dist(doc_i, doc_j)$.

The following few sections describe the different similarity and distance measures used in the dissertation.

3.3 Cosine Similarity

3.3.1 Introduction

When documents are represented as term vectors, the similarity of two documents corresponds to the correlation between the vectors. This is quantified as the cosine of the angle between vectors, that is, the so called cosine similarity. Cosine similarity is one of the most popular similarity measure applied to text documents [4]. In other words, cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between

them. The cosine of 0° is 1, and it is less than 1 for any other angle. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors at an angle of 90° have a similarity of 0, independent of their magnitude. Cosine similarity is particularly used in positive hyper-octant, as in case of documents, where the outcome is neatly bounded in [0,1]. Cosine similarity is most commonly used in high-dimensional positive spaces. Cosine similarity gives a useful measure of how similar two documents are likely to be in terms of their subject matter. An important property of the cosine similarity is its independence of document length. For example, combining two identical copies of a document d to get a new pseudo document d', the cosine similarity between any document doc_i and d is same as that between doc_i and d'.

3.3.2 Mathematical Definition

The Cosine Similarity can be defined mathematically between two vectors $\overrightarrow{A} = a_1, a_2, \ldots, a_n$ and $\overrightarrow{B} = b_1, b_2, \ldots, b_n$ as:

$$\begin{aligned} Cosine\ Similarity &= \frac{\overrightarrow{A}.\overrightarrow{B}}{\|\overrightarrow{A}\|\|\overrightarrow{B}\|} \\ &= \frac{\sum_{i=1}^{n}(a_i \times b_i)}{\sqrt{\sum_{i=1}^{n} a_i^2} \times \sqrt{\sum_{i=1}^{n} b_i^2}} \end{aligned} \tag{3.1}$$

3.3.3 Implementation

The cosine similarity between two documents, doc_i and doc_j (following notions as given in section 3.1.2) can be calculated as follows:

$$Cosine\ Distance = \frac{Numerator}{Denominator}$$

$$Numerator = \sum_{p,q} freq_{i,p} \times freq_{j,q}, \quad summation\ is\ over\ all\ p, q\ such\ that$$

$$1 \leq p \leq n_i\ \&\ 1 \leq q \leq n_j\ \&\ termid_{i,p} = termid_{j,q}$$

$$Denominator = \sqrt{\sum_{p=1}^{n_i} freq_{i,p}^2} \times \sqrt{\sum_{p=1}^{n_j} freq_{j,p}^2}$$

3.3.4 Cosine Similarity Based Distance

The cosine similarity based distance can be defined as:

$$Cosine\ Distance = 1 - Cosine\ Similarity \tag{3.2}$$

We will be referring to the cosine similarity based distance as cosine distance in this dissertation.

3.4 Euclidean Distance

3.4.1 Mathematical Definition

Euclidean distance is widely used in clustering problems, including clustering text [4]. Mathematically, euclidean distance between two vectors $\overrightarrow{A} = a_1, a_2, \ldots, a_n$ and $\overrightarrow{B} = b_1, b_2, \ldots, b_n$ can be defined as:

$$Euclidean\ Distance = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{3.3}$$

For using the euclidean distance as a distance measure, all the document vectors have to be normalized.

3.4.2 Implementation

For using the euclidean distance as a distance measure, the documents need to be normalised because, unlike cosine similarity, euclidean distance does have an effect on the length of the vector.

A document can be normalised as given in section 3.1.3 and the same notation is followed below.

The euclidean distance between two normalised documents $normdoc_i$ and $normdoc_j$ can be calculated by:

$$Euclidean\ Distance = \sqrt{x + y + z}$$

$$x = \sum_{p,q}(normfreq_{i,p} - normfreq_{j,q})^2 \quad summation\ is\ over\ all\ p, q\ such\ that$$
$$1 \leq p \leq n_i\ \&\ 1 \leq q \leq n_j\ \&\ termid_{i,p} = termid_{j,q}$$

$$y = \sum_{p}(normfreq_{i,p})^2 \quad summation\ is\ over\ all\ p\ such\ that$$
$$1 \leq p \leq n_i\ \&\ termid_{i,p} \notin normdoc_j$$

$$z = \sum_{p}(normfreq_{j,p})^2 \quad summation\ is\ over\ all\ p\ such\ that$$
$$1 \leq p \leq n_j\ \&\ termid_{j,p} \notin normdoc_i$$

3.5 Jaccard Distance

3.5.1 Introduction

The Jaccard index, also known as the Jaccard similarity coefficient, is a statistic used for comparing the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets. For dissertation, since documents are represented as vectors, Jaccard Similarity will be used in vector sense which has a slightly different definition.

3.5.2 Mathematical Definition

Mathematically, Jaccard similarity between two vectors $\overrightarrow{A} = a_1, a_2, \ldots, a_n$ and $\overrightarrow{B} = b_1, b_2, \ldots, b_n$ is defined as:

$$Jaccard\ Similarity = \frac{\sum_{i=1}^{n} \min(a_i, b_i)}{\sum_{i=1}^{n} \max(a_i, b_i)} \tag{3.4}$$

The Jaccard distance, which measures dissimilarity between two vectors, is complementary to the Jaccard similarity and can be defined as:

$$Jaccard\ Distance = 1 - Jaccard\ Similarity \tag{3.5}$$

3.5.3 Implementation

Same as in case of euclidean distance, Jaccard distance only makes sense when the documents are normalised. A document can be normalised as given in section 3.1.3 and the same notation is followed below.

The Jaccard distance between two normalised documents $normdoc_i$ and $normdoc_j$ can be calculated by:

$$Jaccard\ Similarity = \frac{Numerator}{Denominator}$$

$$Numerator = \sum_{p,q} min(normfreq_{i.p}, normfreq_{j,q}) \quad summation\ is\ over\ all\ p, q\ such\ that$$
$$1 \leq p \leq n_i\ \&\ 1 \leq q \leq n_j\ \&$$
$$termid_{i,p} = termid_{j,q}$$

$$Denominator = x + y + z$$

$$x = \sum_{p,q} max(normfreq_{i,p}, normfreq_{j,q}) \quad summation\ is\ over\ all\ p, q\ such\ that$$
$$1 \leq p \leq n_i\ \&\ 1 \leq q \leq n_j\ \&\ termid_{i,p} = termid_{j,q}$$

$$y = \sum_{p} normfreq_{i,p} \quad summation\ is\ over\ all\ p\ such\ that$$
$$1 \leq p \leq n_i\ \&\ termid_{i,p} \notin normdoc_j$$

$$z = \sum_{p} normfreq_{j,p} \quad summation\ is\ over\ all\ p\ such\ that$$
$$1 \leq p \leq n_j\ \&\ termid_{j,p} \notin normdoc_i$$

$$Jaccard\ Distance = 1 - Jaccard\ Similarity$$

Chapter 4

k Nearest Neighbor Approach

4.1 Introduction

The intuition underlying Nearest Neighbour Classification is quite straightforward, examples are classified based on the class of their nearest neighbours. It is often useful to take more than one neighbour into account so the technique is more commonly referred to as k-Nearest Neighbour (k-NN) Classification where k nearest neighbours are used in determining the class. Since the training examples are needed at run-time, i.e. they need to be in memory at run-time, it is sometimes also called Memory-Based Classification. Because induction is delayed to run time, it is considered a Lazy Learning technique. Because classification is based directly on the training examples it is also called Example-Based Classification or Case-Based Classification. kNN classification has two stages; the first is the determination of the nearest neighbours and the second is the determination of the class using those neighbours [3].

4.2 Algorithm

If we want to classify a new test document doc_i, we first find the k documents from the training set that have the minimum distance from doc_i. We assign doc_i to class to which majority of these k documents belong. In case of ties, we assign doc_i to the class of the document which is most closest to (has minimum distance from) doc_i among the documents involved in the tie.

Algorithm 4.1: k-NN classification

1 For a new document doc_i, find k documents (in training set) that are closest to (have a least distance from) it ;

2 Assign doc_i the class which the majority of these k documents belong to. In case of ties, assign the class of the document which is most similar to (has the least distance from) doc_i among the documents involved in the tie ;

4.3 Experiments and Results

The confusion matrix for k-NN taking the value of k as 1 using cosine distance as distance measure based on algorithm 4.1 is shown in table 4.1.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	220	4	1	3	1	1	2	2	0	2	0	3	1	6	3	31	4	4	6	25
2	6	234	20	16	16	34	6	2	1	3	2	9	17	3	3	7	4	1	2	3
3	2	23	228	26	4	45	24	1	2	3	4	2	7	1	2	3	3	5	4	5
4	0	15	30	236	38	3	27	3	4	4	2	3	17	4	1	1	1	0	1	2
5	0	15	9	44	242	6	31	0	2	8	2	4	13	1	0	4	1	0	1	2
6	7	42	40	6	15	232	11	3	6	0	1	6	6	2	10	1	3	0	1	3
7	1	5	6	35	29	2	241	14	5	2	7	4	20	3	6	6	2	0	2	0
8	0	4	2	2	4	0	14	311	19	6	1	1	18	1	2	1	3	1	4	2
9	0	3	0	7	3	0	2	21	326	4	2	5	7	5	0	2	4	2	1	4
10	2	0	1	0	6	1	18	4	1	302	46	4	3	1	0	0	3	0	2	3
11	0	6	0	3	1	2	4	6	4	19	345	0	2	2	1	2	1	1	0	0
12	0	3	3	2	4	2	2	2	0	2	1	331	3	2	1	4	14	5	14	1
13	6	9	9	38	25	7	15	10	12	3	4	5	227	5	9	3	0	1	5	0
14	13	12	5	6	9	2	4	10	3	4	7	3	11	247	4	17	7	5	22	5
15	2	17	2	1	6	3	4	12	3	6	3	6	3	14	296	1	8	0	3	4
16	24	4	0	2	1	3	1	6	1	1	2	1	0	6	4	295	3	3	3	38
17	11	1	1	1	1	1	3	4	1	3	1	11	0	9	4	4	268	5	20	15
18	34	2	0	5	0	2	0	1	3	3	3	2	1	1	0	8	8	252	17	34
19	6	1	0	1	2	2	0	4	2	4	3	5	3	4	2	6	75	6	168	16
20	33	1	1	1	0	1	0	2	2	6	1	1	1	4	3	32	14	2	6	140

Table 4.1: Confusion Matrix : 1-NN using Cosine Similarity (Accuracy : 68.27%)

The confusion matrix for k-NN taking the value of k as 1 using Jaccard distance as distance measure and following algorithm 4.1 is shown in table 4.2.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	231	0	3	2	1	0	0	3	0	3	0	6	1	7	3	22	1	1	3	32
2	2	235	27	14	14	34	7	5	4	5	2	16	14	5	2	3	0	0	0	0
3	5	29	252	36	11	29	12	1	1	1	1	1	4	1	4	0	1	1	0	4
4	0	15	36	246	27	6	28	5	2	1	2	2	15	3	2	0	0	0	1	1
5	1	8	12	41	247	7	21	5	1	4	3	3	19	1	6	1	1	0	2	2
6	2	34	48	5	12	254	8	4	0	2	0	5	6	3	8	1	1	1	0	1
7	2	5	8	22	16	5	288	8	6	2	2	0	11	3	2	5	0	0	2	3
8	0	4	2	5	8	2	14	317	12	1	0	0	19	3	2	1	1	1	3	1
9	0	2	0	3	2	0	5	19	350	5	2	2	2	4	0	0	2	0	0	0
10	4	2	0	0	3	0	4	5	2	332	24	4	3	3	4	2	1	0	2	2
11	1	3	2	2	1	1	6	4	1	24	346	0	1	1	2	1	1	1	1	0
12	2	2	4	3	5	4	1	2	3	4	1	339	3	4	3	0	10	2	4	0
13	2	10	17	26	15	8	21	6	8	2	1	10	245	5	8	3	1	0	2	3
14	7	9	6	8	12	8	5	14	5	8	6	1	13	241	11	17	7	6	9	3
15	4	14	3	2	5	7	2	4	3	2	1	2	4	8	311	2	10	0	9	1
16	27	3	0	1	0	1	3	1	1	3	0	2	3	1	2	307	2	4	3	34
17	3	2	1	1	1	1	3	0	0	3	1	10	1	1	2	3	301	4	7	19
18	25	3	1	2	3	0	0	2	1	5	2	2	3	1	3	12	4	282	16	9
19	3	2	1	1	4	0	1	5	0	2	1	8	5	2	2	6	67	3	178	19
20	30	1	0	0	2	2	1	3	1	3	0	2	2	3	6	29	9	4	6	147

Table 4.2: Confusion Matrix : 1-NN using Jaccard Distance (Accuracy : 72.34%)

The confusion matrix for k-NN taking the value of k as 3 using cosine distance as distance measure based on algorithm 4.1 is shown in table 4.3.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	211	3	1	3	1	3	2	2	0	2	0	3	1	10	3	31	3	5	7	28
2	6	232	19	17	16	34	8	2	2	3	3	9	18	2	3	5	4	1	3	2
3	3	21	240	23	4	37	23	1	2	3	3	4	6	1	3	3	3	4	4	6
4	0	12	26	250	39	7	20	3	3	4	1	3	16	3	1	1	1	0	1	1
5	0	12	11	64	234	4	19	1	2	8	1	4	15	1	1	4	1	0	1	2
6	7	43	42	4	15	244	10	2	3	0	1	2	7	1	10	1	1	0	1	1
7	3	5	7	35	25	2	243	15	8	3	7	3	18	2	4	5	2	0	2	1
8	0	4	1	2	3	0	12	322	12	5	1	2	20	1	1	1	3	0	4	2
9	0	2	1	3	2	0	4	27	319	5	2	6	9	6	0	2	2	2	3	3
10	3	0	2	0	6	1	14	4	2	300	49	3	3	1	0	0	4	0	2	3
11	0	6	0	3	2	2	3	6	2	16	350	0	1	1	1	1	1	1	3	0
12	0	2	2	3	2	4	3	2	0	2	2	338	4	2	1	3	11	3	11	1
13	4	10	10	38	21	7	12	12	15	3	4	4	230	5	9	2	0	2	5	0
14	13	11	5	7	8	2	5	10	3	3	8	3	11	241	2	22	8	5	23	6
15	3	17	2	2	4	2	3	6	6	5	2	4	6	11	304	3	8	0	3	3
16	29	4	1	1	1	3	1	4	0	1	2	2	0	4	4	296	2	4	3	36
17	13	1	1	0	1	2	4	6	0	2	1	8	0	8	4	3	274	4	19	13
18	46	2	0	5	0	2	0	1	3	3	3	2	1	0	0	5	7	260	16	20
19	7	0	0	0	2	2	1	4	3	5	2	8	1	5	2	4	80	8	164	12
20	43	0	1	1	0	1	1	0	3	5	1	0	1	4	4	34	15	2	6	129

Table 4.3: Confusion Matrix : 3-NN using Cosine Similarity (Accuracy : 68.79%)

The confusion matrix for k-NN taking the value of k as 3 using Jaccard distance as distance measure based on algorithm 4.1 is shown in table 4.4.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	226	1	3	1	1	0	0	2	0	4	0	5	0	8	3	25	1	2	3	34
2	3	243	23	15	13	28	9	3	5	5	1	14	16	5	3	2	0	0	1	0
3	4	29	267	36	9	22	7	1	0	1	1	1	5	0	4	0	1	1	0	5
4	0	12	36	248	30	10	20	3	1	2	2	4	19	3	2	0	0	0	0	0
5	0	7	15	39	253	6	18	4	1	4	2	4	18	1	6	1	1	0	2	3
6	3	38	50	3	10	260	7	1	0	2	0	5	4	2	7	0	1	1	0	1
7	2	5	6	21	18	5	295	5	8	2	2	0	12	1	1	4	0	0	1	2
8	0	2	2	4	4	1	12	326	12	3	0	0	18	3	2	1	2	0	3	1
9	0	2	0	3	2	0	6	19	348	5	2	2	2	4	0	0	2	1	0	0
10	3	3	0	0	3	0	5	5	2	342	21	3	2	0	3	2	1	0	0	2
11	2	3	1	2	1	0	5	4	2	17	355	0	1	1	2	1	1	1	0	0
12	2	2	5	2	6	2	2	0	3	3	1	344	3	6	2	0	10	0	3	0
13	2	11	18	27	17	9	20	8	11	3	1	10	233	4	10	4	1	0	2	2
14	7	7	6	6	11	10	7	12	6	7	7	1	16	246	8	12	9	5	8	5
15	2	16	3	1	2	6	1	2	3	0	1	2	3	8	323	3	11	0	6	1
16	28	4	0	1	0	1	3	2	1	2	0	2	4	2	2	310	2	3	6	25
17	3	3	2	1	1	0	4	2	0	3	1	10	1	4	5	1	301	4	5	13
18	19	3	1	1	2	0	0	2	0	5	2	2	3	1	4	12	5	294	14	6
19	4	3	1	1	3	0	1	3	0	2	1	9	4	2	2	5	74	3	181	11
20	31	1	0	0	3	1	1	3	1	3	0	1	2	3	7	35	12	4	6	137

Table 4.4: Confusion Matrix : 3-NN using Jaccard Distance (Accuracy : 73.45%)

The confusion matrix for k-NN taking the value of k as 5 using cosine distance as distance measure based on algorithm 4.1 is shown in table 4.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	209	2	0	1	1	3	1	3	0	1	1	3	0	11	3	40	2	7	4	27
2	5	233	21	12	19	32	10	2	4	5	3	8	16	3	4	3	2	2	3	2
3	5	21	245	27	2	40	22	1	2	2	2	2	3	0	2	2	2	4	4	6
4	0	9	28	258	40	6	21	2	1	4	0	5	12	1	0	1	0	1	2	1
5	1	7	15	62	237	6	15	2	1	9	0	2	20	0	1	3	1	0	1	2
6	3	42	41	4	12	252	9	3	3	0	1	1	9	0	7	2	1	0	1	4
7	2	4	8	34	29	4	239	17	5	5	9	3	20	3	2	3	1	1	0	1
8	0	5	2	2	4	0	10	324	13	4	1	2	19	0	0	1	2	1	5	1
9	1	2	1	6	2	0	3	27	320	5	3	3	9	5	0	4	1	1	2	3
10	3	0	1	0	3	2	15	3	5	308	44	3	2	1	0	0	3	0	2	2
11	0	4	0	0	1	2	6	9	1	12	356	0	1	0	1	1	2	0	3	0
12	0	0	4	2	5	5	3	1	0	1	3	339	3	2	2	3	9	6	7	1
13	2	11	12	38	24	7	13	12	17	2	4	7	221	3	9	2	0	1	6	2
14	15	8	2	5	13	2	1	9	4	4	12	5	10	248	2	22	3	5	22	4
15	4	20	0	1	2	1	2	3	6	4	3	5	11	8	306	3	7	2	6	0
16	30	5	2	1	1	2	1	3	0	2	1	1	1	7	4	307	1	3	6	20
17	11	3	1	1	3	2	3	2	2	1	1	11	1	6	6	4	272	1	18	15
18	54	3	0	3	0	1	0	1	2	4	1	4	0	0	0	2	12	265	13	11
19	8	2	0	1	1	1	0	2	6	5	3	6	2	6	2	3	82	8	166	6
20	41	2	0	1	0	2	0	3	2	5	1	1	2	4	3	41	15	3	4	121

Table 4.5: Confusion Matrix : 5-NN using Cosine Similarity (Accuracy : 69.38%)

The confusion matrix for k-NN taking the value of k as 5 using Jaccard distance as distance measure based on algorithm 4.1 is shown in table 4.6.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	232	1	0	1	1	0	1	2	0	2	0	5	1	7	2	29	1	4	3	27
2	4	252	25	13	7	27	11	5	5	7	1	12	12	3	2	0	0	0	1	2
3	5	23	279	33	9	21	9	0	0	1	1	2	2	0	4	0	1	0	0	4
4	0	10	38	256	24	5	21	3	2	1	2	4	22	1	0	1	0	0	0	2
5	1	8	15	44	259	4	16	3	0	3	1	2	14	0	5	2	1	0	3	4
6	4	41	63	3	12	247	6	0	1	1	1	3	4	1	5	0	1	1	1	0
7	1	4	4	19	16	3	312	7	4	3	2	0	7	1	0	3	0	0	1	3
8	0	4	3	4	5	0	10	321	13	6	0	1	16	3	2	1	2	1	2	2
9	1	1	0	3	2	1	6	17	351	6	1	1	3	2	0	1	1	1	0	0
10	1	4	1	0	2	0	4	3	4	349	22	1	1	0	2	0	2	0	1	0
11	2	2	1	1	1	0	3	5	1	15	361	0	1	1	1	1	2	1	0	0
12	3	4	6	1	4	2	2	0	1	1	1	350	3	7	2	0	7	0	2	0
13	2	14	22	22	17	9	16	7	17	2	0	7	237	4	9	3	2	0	1	2
14	5	11	4	8	11	5	7	13	5	6	6	1	19	250	6	16	7	4	6	6
15	3	13	2	0	6	4	3	1	3	2	1	2	6	5	329	2	9	0	3	0
16	25	4	2	3	1	1	2	2	1	1	0	1	2	1	1	318	1	4	4	24
17	5	1	1	0	1	0	3	0	1	4	0	11	1	2	4	1	313	3	5	8
18	16	3	0	1	0	2	1	1	1	4	2	2	3	3	5	11	6	302	7	6
19	5	3	1	0	1	0	0	1	2	3	0	5	4	2	5	4	80	5	179	10
20	34	1	0	0	2	1	2	0	1	3	0	0	2	3	7	41	12	5	4	133

Table 4.6: Confusion Matrix : 5-NN using Jaccard Distance (Accuracy : 74.75%)

The confusion matrix for k-NN taking the value of k as 10 using cosine distance as distance measure based on algorithm 4.1 is shown in table 4.7.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	201	1	2	1	2	3	2	1	0	1	1	2	0	11	5	56	3	6	0	21
2	6	252	23	10	14	30	3	2	3	4	1	12	16	0	2	2	1	3	3	2
3	5	17	261	31	6	36	9	3	1	3	3	4	4	0	2	1	0	1	1	6
4	1	6	29	265	39	8	16	1	1	4	2	3	11	0	0	2	0	2	1	1
5	2	8	16	63	236	6	13	3	3	6	0	3	16	0	0	6	1	0	1	2
6	5	42	42	4	15	257	5	1	3	2	1	1	7	1	4	1	0	0	1	3
7	2	4	6	48	25	5	233	18	8	4	5	3	19	0	0	2	3	1	3	1
8	1	2	2	3	5	0	6	324	11	2	2	2	21	0	0	2	3	3	6	1
9	1	2	1	9	3	0	2	22	330	3	4	1	10	2	1	1	1	2	2	1
10	3	0	1	0	4	1	4	3	8	318	47	1	1	0	0	1	2	0	2	1
11	0	5	0	0	1	1	0	10	1	10	363	0	1	0	1	1	2	0	3	0
12	1	1	2	1	6	3	3	0	0	2	2	349	2	1	1	1	9	2	9	1
13	0	13	10	41	23	4	9	12	29	3	7	12	206	5	6	2	2	1	4	4
14	11	10	3	5	12	7	5	7	6	2	15	3	15	244	2	16	5	4	21	3
15	6	18	0	1	4	0	2	3	4	4	3	2	10	8	309	5	6	3	6	0
16	27	3	2	1	1	2	0	1	0	3	1	2	2	4	3	313	1	2	8	22
17	9	2	1	1	2	1	2	1	3	1	0	14	0	3	4	4	294	4	11	7
18	58	2	0	1	1	0	0	0	1	5	3	4	0	2	0	5	9	269	11	5
19	11	2	1	0	1	0	0	0	2	4	5	6	2	5	3	2	84	6	171	5
20	37	0	0	0	2	3	3	2	2	3	0	1	2	6	3	52	14	4	4	113

Table 4.7: Confusion Matrix : 10-NN using Cosine Similarity (Accuracy : 70.47%)

The confusion matrix for k-NN taking the value of k as 10 using Jaccard distance as distance measure based on algorithm 4.1 is shown in table 4.8.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	226	2	0	1	1	0	1	2	0	0	0	5	1	7	4	34	2	5	4	24
2	4	264	22	13	10	29	11	0	0	7	1	9	11	0	4	0	0	0	2	2
3	6	26	287	27	7	17	8	0	2	0	1	2	2	0	4	1	1	0	0	3
4	1	10	37	262	28	8	16	1	2	1	1	1	19	0	2	0	0	0	2	1
5	4	7	19	55	254	2	15	0	0	3	0	3	15	0	2	0	1	0	1	4
6	3	41	60	5	6	253	5	3	2	0	1	3	3	1	3	0	0	2	4	0
7	0	3	5	21	14	3	314	8	3	3	2	0	7	1	1	4	0	0	0	1
8	1	4	2	3	6	0	11	326	10	7	0	1	12	2	0	3	3	0	2	3
9	1	0	1	2	4	1	5	21	351	5	1	0	3	0	0	1	2	0	0	0
10	1	2	1	0	3	0	5	3	3	356	21	1	0	0	0	0	0	0	0	1
11	0	4	0	1	1	0	3	5	1	14	364	0	1	2	0	0	1	0	1	1
12	4	4	8	0	6	2	2	0	1	1	1	353	2	1	1	1	3	3	3	0
13	2	21	14	31	24	8	16	6	15	3	1	12	221	1	6	3	1	2	2	4
14	6	14	6	6	15	5	3	6	5	7	4	1	16	255	3	14	4	6	13	7
15	6	15	2	1	5	4	3	2	3	3	2	2	9	6	320	1	4	2	4	0
16	28	4	1	1	1	3	1	1	1	1	0	0	4	1	1	325	0	1	3	21
17	4	2	2	0	1	0	3	1	0	4	1	11	1	2	3	3	317	4	4	1
18	20	2	1	0	0	0	2	0	3	4	3	2	2	1	0	7	7	311	5	6
19	7	2	0	0	2	0	1	0	2	3	0	5	2	2	5	3	86	3	180	7
20	39	2	1	0	2	1	2	2	0	2	0	1	0	6	6	41	12	3	5	126

Table 4.8: Confusion Matrix : 10-NN using Jaccard Distance (Accuracy : 75.21%)

The confusion matrix for k-NN taking the value of k as 20 using cosine distance as distance measure based on algorithm 4.1 is shown in table 4.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	194	2	2	1	2	2	5	0	0	5	1	3	0	10	6	59	6	5	2	14
2	4	251	25	13	15	31	4	4	2	3	2	11	11	3	4	1	0	2	0	3
3	3	20	263	32	9	34	6	0	1	3	1	5	4	1	3	1	0	1	2	5
4	0	3	39	253	48	7	10	2	1	6	2	1	13	1	1	1	0	2	0	2
5	4	6	22	69	239	3	11	1	3	5	0	2	13	1	1	3	0	0	0	2
6	7	54	53	3	10	241	3	1	1	0	0	2	10	2	5	1	1	0	1	0
7	3	5	8	53	25	3	241	16	4	1	3	3	15	3	0	3	3	0	0	1
8	1	4	1	4	1	1	7	322	13	2	1	2	20	0	0	2	5	2	6	2
9	3	2	0	12	2	1	2	19	327	4	1	0	12	2	0	1	4	1	3	2
10	7	0	1	0	3	0	4	5	2	313	52	0	1	2	1	2	3	0	1	0
11	1	4	1	0	2	0	0	8	0	12	365	0	1	0	0	1	2	0	2	0
12	3	4	9	2	5	1	3	2	0	4	2	347	1	0	1	0	8	1	2	1
13	3	15	16	51	25	4	9	9	24	5	7	16	188	4	4	3	1	0	3	6
14	13	8	3	5	11	7	9	7	2	6	12	4	17	246	4	12	3	6	19	2
15	6	17	0	1	5	2	4	1	3	4	1	2	9	6	312	3	11	2	4	1
16	30	5	2	0	1	2	1	0	0	2	0	1	4	2	2	321	0	1	11	13
17	8	3	1	0	1	1	1	1	2	2	0	10	0	1	4	4	304	4	11	6
18	55	4	0	0	0	0	0	0	1	6	1	7	0	0	0	4	7	275	11	5
19	10	3	0	0	0	0	0	0	1	5	2	8	3	4	1	0	91	7	171	4
20	46	1	1	1	0	1	0	1	2	3	1	0	0	5	4	61	21	3	7	93

Table 4.9: Confusion Matrix : 20-NN using Cosine Similarity (Accuracy : 69.92%)

The confusion matrix for k-NN taking the value of k as 20 using Jaccard distance as distance measure based on algorithm 4.1 is shown in table 4.10.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	210	2	0	0	2	0	1	0	0	2	1	4	0	8	5	46	2	7	3	26
2	2	260	24	9	11	35	9	2	2	6	0	8	10	0	4	2	0	0	2	3
3	6	24	296	26	6	13	4	1	1	1	1	3	1	0	5	2	1	0	0	3
4	0	13	44	267	27	2	14	0	0	2	2	3	14	0	1	1	0	0	0	2
5	2	6	16	55	265	3	12	0	1	3	0	3	11	0	1	0	1	1	1	4
6	4	41	70	4	4	251	3	1	2	2	0	4	2	1	4	1	0	0	1	0
7	0	3	4	23	16	1	315	10	2	1	2	0	5	2	0	4	0	0	1	1
8	0	3	3	3	8	1	11	327	9	7	1	1	12	0	1	2	2	1	1	3
9	2	1	2	2	2	2	3	13	360	2	1	2	2	1	0	1	1	0	0	1
10	1	2	2	0	4	0	5	3	1	352	24	0	1	0	1	0	1	0	0	0
11	0	3	0	1	1	1	1	5	1	12	370	0	0	0	0	0	1	0	1	2
12	3	7	6	2	4	1	0	0	6	1	0	350	2	1	2	2	3	1	4	1
13	0	18	17	28	23	3	18	5	13	4	1	18	215	3	10	5	1	2	5	4
14	7	17	5	5	18	3	7	8	3	6	7	1	9	259	3	18	5	3	7	5
15	4	16	2	0	7	1	3	1	2	3	2	5	8	4	325	2	4	1	4	0
16	33	4	2	1	2	0	3	1	1	2	0	1	1	0	1	326	0	0	3	17
17	4	0	1	0	1	0	3	2	2	6	0	11	1	0	1	2	324	3	1	2
18	22	1	1	0	0	0	2	0	5	6	1	3	1	1	0	2	6	313	8	4
19	6	2	0	0	2	1	3	0	0	3	0	4	1	1	8	3	94	4	176	2
20	40	2	1	1	1	0	2	1	1	3	0	0	0	3	7	49	19	3	5	113

Table 4.10: Confusion Matrix : 20-NN using Jaccard Distance (Accuracy : 75.33%)

Chapter 5

k-Means Clustering

5.1 Introduction

k-means is one of the simplest unsupervised classification 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 a priori. The main idea is to define k centroids, one for each cluster. These centroids 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 document belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early grouping is done. At this point we need to re-calculate k new centroids as centroid 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 centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more [6].

The objective of the k-Means clustering is to minimize the intra-cluster distances (distance between pair of documents belonging to the same cluster) and maximize the inter-cluster distances (distance between pair of documents belonging to different clusters).

5.2 k-Means on Euclidean Space

k-Means clustering tries to minimize the sum of square of distances of all the documents from the seeds of the cluster it belongs to. If we want to create k clusters $cluster_1, cluster_2, \ldots .cluster_k$ with cluster seeds as $clusterSeed_1, clusterSeed_2, \ldots .clusterSeed_k$, k-Means tries to find a centroid for each cluster in order to minimize the following expression:

$$\sum_{j=1}^{k}(\sum_{doc_i \in cluster_j} dist(doc_i, centroid_j)^2)$$

The $dist()$ in the above expression refers to the euclidean distance. The centroid of a cluster is defined as:

$$centrod_i = \frac{\sum_{doc_j \in cluster_i} doc_j}{clusterSize_i}$$

5.3 Experiments and Results

5.3.1 k-Means

The basic algorithm for k-Means clustering is algorithm 5.1. Initially K different documents that are far apart from each other are selected. This is done by selecting a document at random from the training set and computing its distance between already selected documents and if any of these distance is less than a threshold value $threshold_1$ (a value around 0.9 in case of cosine distance), we reject that document and select some other document randomly from the training set and follow the same process. If all the above distances are more than $threshold_1$, we select that document. After selection, these K documents are set as cluster centers or seeds. All the documents in the training set are assigned to the cluster whose seed it is closest to. After allocation of the documents to a cluster, we find the centroid of each cluster. Centroid $centroid_i$ of cluster $cluster_i$ of size $clusterSize_i$ can be calculated as:

$$centrod_i = \frac{\sum_{doc_j \in cluster_i} doc_j}{clusterSize_i}$$

Adding two documents doc_i and doc_j is adding frequencies of the terms common in doc_i and doc_j and keeping frequencies all other terms (present in one of the documents) as they they are. After finding the centroids of each cluster, we consider them as new seeds and find the distance of each seed from its previous seed. If all of these distances is less than a threshold $threshold_2$ (a value around 0.001 in case of cosine distance), we stop the process and report the new seeds as final cluster seeds, else we continue the process of assigning each document to the cluster and finding the cluster seeds until the condition is fulfilled.

Algorithm 5.1: k-Means Clustering

Assign a set of cluster seeds to a null set $clusterSeeds = \{\}$;
repeat
    Select a document doc_i at random from the training set ;
    for *Each element* $seed_j$ *in* $clusterSeeds$ **do**
        if *Distance between* $centroid_j$ *and* $doc_i < threshold_1$ **then**
            Go to step 3 ;
        end
    end
    Append doc_i to $clusterSeeds$;
until *size of* $clusterSeeds < K$;
for *Each document* doc_i *in training set* **do**
    Find $seed_j$ in cluster $clusterSeeds$ which has minimum distance from doc_i ;
    Assign doc_i to $cluster_j$;
end
Assign $newClusterSeeds = \{\}$;
for $i = 1$ *to* K **do**
    Calculate the centroid of all documents in $cluster_i$ and append it to $newClusterSeeds$;
end
for $i = 1$ *to* K **do**
    if *distance between* j^{th} *element of* $clusterSeeds$ *and* j^{th} *element of* $newClusterSeeds > threshold_2$ **then**
        Make $clusterSeeds$ as $newClusterSeeds$;
        Go to step 11 ;
    end
end
for *Each document* doc_i *in training set* **do**
    Find $seed_j$ in cluster $newClusterSeeds$ which has minimum distance from doc_i ;
    Assign doc_i to $cluster_j$;
end
Report $newClusterSeeds$ as cluster seeds

After performing the k-Means clustering on the training set using $k = 200$ and cosine distance as distance measure, the distribution of the 20 classes over the 200 clusters can be seen in table 5.1, table 5.2, table 5.3 and table 5.4.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	0	0	0	0	0	0	0	0	0	114	2	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	57	0	3	19
3	2	8	5	35	4	1	13	2	2	0	0	7	7	2	4	0	0	1	2	0
4	3	1	0	0	0	2	0	1	1	3	4	3	0	1	3	4	14	12	22	1
5	0	0	4	0	0	0	2	7	1	0	0	0	3	0	19	0	0	0	0	0
6	0	8	10	8	1	22	1	2	3	11	0	3	2	0	0	0	1	0	0	0
7	3	1	7	4	4	4	5	5	1	3	2	27	8	3	0	0	0	0	0	1
8	31	10	6	6	9	9	5	33	47	38	8	16	13	26	26	12	19	15	41	26
9	0	6	11	2	5	12	0	0	0	0	0	0	1	0	0	0	0	0	0	0
10	0	38	1	1	2	15	0	2	0	0	1	1	2	0	2	0	0	0	0	1
11	0	0	1	0	0	0	0	0	0	9	62	0	0	0	1	0	1	0	3	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	42	0	0	0	0	0
13	0	0	0	0	2	0	0	1	0	0	26	0	0	0	0	0	0	0	0	0
14	0	0	1	0	0	2	0	0	0	0	0	0	0	0	0	10	0	18	0	0
15	4	2	3	1	8	0	1	2	2	8	2	1	3	6	3	1	1	3	4	1
16	0	10	6	3	3	2	0	0	0	1	0	4	9	0	1	0	0	0	0	0
17	2	4	2	3	2	1	2	1	0	3	1	0	6	4	0	0	0	2	0	7
18	6	0	0	1	0	0	0	0	0	0	0	4	1	2	2	4	2	0	2	0
19	0	1	3	5	3	2	1	1	1	0	0	0	4	4	0	1	1	0	0	2
20	0	1	2	1	2	2	2	0	1	0	1	1	3	0	2	0	1	0	0	1
21	9	0	0	0	0	1	0	0	3	0	0	6	0	0	0	9	15	2	14	7
22	0	11	56	81	47	0	28	14	19	2	0	0	7	0	1	0	1	0	0	0
23	0	0	0	0	0	0	2	9	0	0	0	1	20	0	2	0	0	0	0	0
24	1	16	3	0	9	25	1	0	0	0	0	0	0	1	1	0	0	0	0	0
25	0	0	0	0	0	0	4	0	0	2	21	0	1	0	0	0	2	0	1	0
26	0	2	0	0	0	0	1	2	0	0	0	0	9	0	0	1	0	0	0	0
27	0	6	1	7	3	7	3	0	0	2	0	0	2	2	2	0	0	1	1	0
28	51	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	29
29	0	0	0	0	0	0	0	0	0	0	0	0	0	55	0	0	0	0	0	0
30	4	5	1	5	3	7	3	4	5	4	7	1	13	9	23	5	3	7	2	0
31	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	11	0	0	0	0
32	0	0	0	1	0	0	0	0	0	4	0	0	20	0	0	0	0	0	0	0
33	0	0	0	0	0	0	4	1	103	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0
35	0	0	0	0	0	0	0	1	0	9	26	0	0	0	0	0	0	1	7	1
36	0	10	1	10	7	3	10	1	0	1	0	3	5	1	3	0	0	0	0	0
37	0	12	22	48	31	4	14	0	0	0	2	0	5	0	0	0	5	0	0	0
38	0	4	43	10	0	2	10	0	0	0	0	4	1	0	0	0	0	0	0	0
39	3	1	0	0	0	0	12	164	7	0	0	0	9	0	0	0	1	0	1	1
40	0	1	0	1	0	1	3	2	1	12	1	3	1	15	3	7	1	0	3	2
41	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0
42	0	0	1	3	11	11	4	0	0	0	0	0	7	2	0	0	0	0	0	0
43	0	0	0	0	0	0	3	0	65	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	1	4	1	0	0	0	0	0	1	0	158	0	2	1
45	1	0	0	0	0	0	0	0	0	0	0	0	0	0	57	0	0	0	0	1
46	1	49	70	4	2	29	0	0	1	0	0	6	2	0	0	0	1	0	0	1
47	2	8	12	16	29	27	2	3	2	1	0	0	3	13	3	0	2	2	1	0
48	2	0	0	0	0	0	2	1	0	0	0	0	4	0	6	6	0	1	0	1
49	2	1	4	1	1	4	0	3	1	0	0	4	3	1	0	8	1	3	0	3
50	4	0	0	0	0	0	1	0	0	0	0	0	0	0	0	6	0	0	66	10

Table 5.1: Class information of clusters by k-Means clustering of training set (1/4)

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
51	0	0	0	0	0	0	0	3	59	0	0	0	0	0	0	0	1	0	0	0
52	3	0	0	0	0	0	0	0	0	0	0	0	0	0	35	0	0	0	0	2
53	0	0	0	0	0	0	0	0	0	0	0	162	0	0	0	0	1	0	1	0
54	1	0	0	0	0	0	0	0	0	37	1	0	0	0	0	3	0	0	0	0
55	0	0	5	18	8	0	9	1	0	0	0	3	2	0	0	1	0	0	0	0
56	5	13	3	2	1	3	19	0	3	4	0	0	11	0	4	16	1	0	2	1
57	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
58	0	10	4	12	10	1	6	0	0	0	0	0	3	0	0	0	0	1	0	0
59	0	0	0	1	0	1	2	1	0	5	0	0	0	4	0	3	0	0	0	1
60	0	0	9	6	14	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
61	7	0	0	0	0	0	0	0	0	0	0	1	0	0	0	109	1	0	0	50
62	0	0	3	5	2	3	7	1	0	0	0	4	5	5	2	1	0	0	0	0
63	10	30	7	15	21	10	19	20	28	23	16	5	24	13	13	0	11	6	9	2
64	1	0	0	0	0	3	3	0	0	0	0	0	11	0	0	0	0	0	0	0
65	0	0	0	0	1	11	2	0	0	0	0	0	0	0	0	0	0	0	0	0
66	0	0	0	0	0	0	0	0	0	2	19	0	0	0	1	1	0	0	0	0
67	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	2	55	0	1
68	0	4	21	1	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
69	3	0	0	0	0	0	0	1	1	3	0	0	0	1	8	1	2	8	0	1
70	0	7	1	3	3	3	5	1	1	2	0	2	25	4	0	1	0	0	2	1
71	3	0	0	1	6	7	2	7	5	4	3	3	1	1	1	1	2	0	2	3
72	0	0	0	0	0	0	0	0	0	0	41	1	0	0	0	0	0	0	0	0
73	10	8	2	3	2	0	2	2	11	6	0	6	4	4	7	3	9	11	15	8
74	0	0	0	12	7	0	0	1	1	0	0	2	12	0	0	0	0	0	0	0
75	0	6	27	5	16	2	14	0	0	0	0	0	6	0	1	0	0	0	0	0
76	0	0	1	0	2	0	9	0	0	0	0	0	0	0	1	0	0	0	1	0
77	0	0	10	0	0	7	0	0	0	0	1	0	0	0	0	0	0	0	0	0
78	0	0	0	0	0	0	0	0	1	2	29	0	0	0	1	0	0	0	0	0
79	0	0	4	0	1	0	2	0	0	0	0	0	1	18	1	0	0	0	0	0
80	0	0	0	0	0	0	1	5	3	0	4	0	3	0	13	0	0	0	1	0
81	0	0	1	14	43	0	3	0	0	0	0	0	9	0	0	0	0	0	0	0
82	0	0	0	0	0	2	0	3	0	0	0	0	7	0	0	0	2	0	0	0
83	1	0	1	0	0	0	0	0	3	1	1	7	0	1	1	0	13	0	26	0
84	0	0	1	0	7	0	0	0	0	0	0	0	0	12	0	1	0	0	0	1
85	0	0	0	0	0	0	1	0	0	0	10	0	0	0	0	0	1	0	0	0
86	0	0	0	0	0	0	0	0	0	48	0	0	0	0	0	0	0	0	0	0
87	0	0	0	0	0	0	12	1	6	2	3	0	0	0	0	0	4	0	1	0
88	0	10	0	0	0	0	1	0	0	0	2	1	2	0	6	0	0	0	0	0
89	0	0	0	1	0	0	1	4	2	0	0	1	19	4	1	3	0	6	0	0
90	0	0	1	0	0	0	0	20	2	0	0	0	0	4	7	0	0	0	0	0
91	18	0	0	0	1	2	0	5	2	2	9	11	0	7	4	8	11	8	19	9
92	0	8	0	1	0	10	0	0	0	0	0	0	1	0	0	0	1	0	0	0
93	0	1	1	2	3	0	105	5	2	0	0	0	2	0	0	0	3	0	0	0
94	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0
95	0	0	0	0	0	0	0	0	0	1	0	0	8	14	0	15	0	0	0	0
96	1	15	11	7	1	21	0	0	2	0	0	2	10	0	4	1	0	2	0	0
97	0	0	0	0	0	0	0	0	0	0	0	0	0	80	0	0	0	0	0	0
98	0	0	2	0	0	1	18	1	1	2	0	0	2	0	0	0	0	0	0	0
99	0	0	0	0	0	0	0	28	5	0	0	0	0	1	5	0	0	0	1	0
100	0	8	0	0	0	1	0	1	0	0	0	0	1	0	3	0	0	0	0	0

Table 5.2: Class information of clusters by k-Means clustering of training set (2/4)

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
101	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	8	0	0
102	0	1	0	3	4	0	2	10	9	2	3	0	0	3	3	0	2	1	0	2
103	0	2	2	9	1	0	2	0	0	0	1	3	2	0	0	0	0	0	0	0
104	0	0	0	0	0	0	0	12	0	0	1	3	0	5	0	4	6	0	30	1
105	0	5	0	7	6	0	4	2	0	1	0	20	21	0	1	0	0	0	9	0
106	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	42	0	1	0	3
107	10	0	2	0	1	0	1	0	0	1	0	1	0	0	1	3	0	0	2	3
108	0	0	9	0	0	0	2	2	1	0	0	0	1	1	3	0	0	1	0	0
109	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	0	1	0
110	0	1	0	2	1	0	4	4	0	0	0	0	7	0	1	1	2	1	1	2
111	0	1	2	1	0	1	1	0	0	0	1	1	0	9	0	0	0	1	0	0
112	0	2	0	0	0	0	0	3	27	0	0	0	0	2	3	0	0	0	0	0
113	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	103	0	0
114	0	0	0	0	0	0	0	11	1	0	0	0	0	0	0	0	0	0	0	0
115	0	3	0	2	2	0	30	0	1	43	39	0	6	0	0	0	1	0	0	1
116	0	0	3	62	15	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0
117	0	0	0	0	0	28	0	0	0	0	0	0	0	0	0	0	0	0	0	1
118	0	2	1	0	0	5	0	0	1	0	0	1	3	0	0	0	0	0	0	0
119	0	12	12	7	8	15	18	4	3	9	2	3	5	4	2	6	1	1	4	0
120	12	2	2	0	1	1	3	9	5	1	1	5	5	7	6	5	9	1	4	1
121	1	0	0	0	0	0	0	1	0	0	0	16	2	1	0	1	11	2	9	1
122	0	0	0	0	0	0	0	0	0	0	0	47	0	0	0	0	0	0	0	0
123	0	1	0	19	31	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0
124	0	0	0	0	0	0	1	24	0	0	0	0	2	0	0	0	0	0	0	0
125	0	1	1	0	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0
126	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	13	0	1
127	0	0	2	0	0	0	3	0	0	1	0	0	0	0	14	1	0	0	0	2
128	0	1	1	0	0	5	2	2	1	0	2	0	9	8	7	0	0	0	0	0
129	0	0	0	0	1	3	0	0	0	0	0	0	0	1	0	0	13	0	1	0
130	0	1	0	0	1	0	5	0	0	30	55	0	5	0	0	0	0	0	0	1
131	0	0	0	0	0	2	0	3	0	2	1	0	0	0	3	0	0	0	0	0
132	0	4	3	1	0	0	7	3	0	0	0	0	0	11	0	0	0	0	0	0
133	1	1	0	1	0	1	0	2	0	4	3	1	1	0	6	0	2	5	2	1
134	0	2	7	2	0	10	0	0	0	0	0	2	1	0	0	1	0	0	0	0
135	0	2	0	0	0	0	0	1	17	2	0	0	1	0	0	0	0	0	0	0
136	1	4	0	5	5	1	0	1	0	0	0	0	4	3	1	0	1	0	0	0
137	0	0	0	0	0	0	0	47	22	2	0	0	2	0	0	0	0	0	1	1
138	0	1	0	0	0	0	0	0	5	0	0	0	0	5	16	0	0	0	0	0
139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	39	1	0
140	2	3	1	3	1	11	6	3	3	5	3	3	3	3	5	0	4	4	2	0
141	0	0	0	0	0	0	0	0	0	0	0	0	4	19	3	0	0	0	0	1
142	0	0	1	0	2	0	3	3	1	0	1	1	2	12	1	2	1	2	0	2
143	0	0	0	0	2	0	0	0	0	9	1	0	0	0	0	0	0	0	0	0
144	0	3	2	1	45	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0
145	0	0	1	0	0	0	9	1	0	1	0	0	0	0	0	0	0	0	2	0
146	0	11	0	1	0	3	0	0	0	0	18	0	0	0	0	0	0	0	0	0
147	6	4	1	2	3	5	1	7	5	4	0	2	14	5	7	12	1	5	2	6
148	58	1	0	1	0	0	1	0	2	1	0	0	0	0	0	102	0	2	3	36
149	0	0	0	1	0	8	0	4	0	0	0	124	2	0	0	0	0	0	0	0
150	0	0	0	1	9	0	0	0	0	0	0	0	10	0	3	0	0	0	0	0

Table 5.3: Class information of clusters by k-Means clustering of training set (3/4)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
151	0	0	0	0	0	0	0	2	0	0	0	0	0	3	4	0	0	0	3	2
152	1	6	1	1	1	2	0	1	1	1	1	0	12	0	1	2	0	2	1	0
153	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	39	0	0	1	21
154	51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	2
155	0	0	0	1	4	0	0	0	0	44	8	0	2	0	0	0	0	0	2	0
156	0	3	0	1	0	1	0	0	0	0	2	2	2	0	0	0	0	0	0	0
157	0	0	2	1	0	0	1	2	8	0	1	0	0	5	0	0	0	0	0	0
158	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	137	0	1
159	0	0	0	0	0	0	0	0	0	34	108	0	0	0	0	0	0	0	1	0
160	1	0	1	0	0	0	0	3	9	0	0	0	0	0	1	0	2	5	51	5
161	0	46	0	1	0	7	0	0	1	0	0	1	0	0	0	0	0	0	0	0
162	8	0	0	1	1	0	0	0	0	0	0	2	0	18	4	4	1	1	0	8
163	0	0	0	0	0	0	2	12	8	0	0	0	0	0	0	0	0	0	0	0
164	49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	1
165	0	3	5	1	8	4	3	3	2	0	0	1	13	8	7	2	0	1	0	1
166	0	0	1	0	5	2	3	0	0	0	0	0	7	0	0	0	1	0	0	0
167	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	49	0	0	0	8
168	8	2	1	1	0	2	3	0	5	5	4	2	0	1	1	4	9	4	4	6
169	0	0	0	1	5	0	4	1	9	6	0	0	14	0	0	0	0	0	0	0
170	0	0	1	0	0	6	0	0	0	0	0	0	1	0	0	0	0	0	0	0
171	0	3	0	2	3	0	0	1	7	0	0	0	0	0	0	1	1	0	0	0
172	0	1	2	40	34	2	7	0	0	0	0	4	8	0	0	0	0	0	0	0
173	0	0	0	0	0	0	4	0	1	0	19	0	1	0	0	0	0	0	1	0
174	8	0	0	0	0	4	0	0	0	1	0	0	0	1	0	16	1	0	1	10
175	0	0	0	0	0	0	3	2	1	4	0	0	2	0	1	11	2	0	4	2
176	1	4	1	0	1	0	2	0	0	0	1	1	2	2	5	2	0	1	4	0
177	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	7	0	0	1
178	1	1	0	1	0	0	1	0	0	5	0	0	0	0	2	1	0	0	1	0
179	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0
180	0	15	107	10	1	89	2	4	2	0	0	0	0	1	0	0	0	0	0	0
181	0	0	0	0	0	0	0	0	0	0	0	6	0	2	1	0	30	2	1	0
182	25	0	0	0	0	2	0	0	0	0	0	0	0	0	0	9	1	1	3	9
183	1	16	2	1	0	0	0	3	0	0	0	3	1	5	0	2	0	0	0	1
184	0	18	1	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
185	0	0	0	0	4	0	5	0	0	0	0	1	45	0	0	0	0	0	0	0
186	1	40	4	0	0	4	0	0	0	0	0	0	0	2	5	0	0	0	0	0
187	0	1	2	0	0	2	1	1	0	0	0	1	0	0	107	0	0	0	0	0
188	0	0	0	0	0	0	0	0	0	7	16	0	0	0	1	0	0	0	0	0
189	0	0	0	0	0	0	31	1	5	0	0	1	0	0	2	0	0	0	0	0
190	0	0	0	1	5	1	0	7	10	0	0	0	2	0	0	1	0	0	0	0
191	0	0	0	0	0	0	0	0	0	0	0	0	0	78	0	0	0	0	0	0
192	15	0	0	0	0	0	0	3	1	0	0	0	1	0	1	4	11	14	5	11
193	0	0	0	0	0	0	1	0	16	0	0	0	0	0	4	0	0	1	0	1
194	1	0	0	0	0	0	0	0	0	0	0	23	0	0	5	0	27	7	49	1
195	0	1	1	2	2	5	3	3	1	0	3	1	3	9	2	2	0	0	0	0
196	3	0	0	1	0	0	1	7	0	0	0	2	0	6	27	1	0	0	3	1
197	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	22
198	0	1	4	20	26	0	14	0	0	0	0	4	4	0	0	0	0	0	0	0
199	0	0	0	0	1	0	1	0	1	0	0	0	2	0	0	0	0	0	0	0
200	0	14	0	0	0	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.4: Class information of clusters by k-Means clustering of training set (4/4)

5.3.2 k_2-NN after k_1-Means

A simple algorithm to assign classes to a test document doc_i is algorithm 5.2. The idea is to first find the k_1 cluster seeds and k_1 clusters using the k_1-Means algorithm (this has already been done in section 5.3.1). Then we find the cluster seed closest to the document doc_i. We consider the documents of the closest cluster as the training set and perform k_2-NN algorithm to assign a class to doc_i.

Algorithm 5.2: k_2-NN classification after k_1-Means Clustering

1 Find 200(k_1) clusters using k-Means clustering on the training dataset ;
2 For a new document doc_i, find the cluster seed closest to it and let that cluster be C ;
3 Find k_2 documents from cluster C that are most similar to doc_i ;
4 Assign doc_i to the class to which the majority of these k_2 documents belong to. In case of ties, assign the class of the document which is most similar to doc_i among the documents involved in the tie. ;

The confusion matrix on applying 1-NN ($k_2 = 1$) after making 200 clusters using k-Means and using cosine distance as distance measure based on algorithm 5.2 is given in table 5.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	180	5	1	2	4	6	6	5	2	7	0	4	1	4	4	47	5	5	5	26
2	6	184	34	16	25	43	8	8	2	7	0	12	20	0	4	3	3	3	6	5
3	4	27	184	28	19	54	24	6	1	4	2	4	8	2	6	3	3	1	6	8
4	4	21	34	186	55	9	26	3	4	12	2	7	19	4	2	1	0	1	1	1
5	4	18	13	56	206	5	18	3	2	11	0	5	23	6	3	1	1	3	3	4
6	4	55	47	7	22	180	12	6	6	6	1	7	12	8	8	3	1	1	7	2
7	2	12	6	36	33	6	216	18	10	3	2	9	16	6	2	4	1	0	6	2
8	6	6	1	8	14	1	13	264	21	8	3	6	20	2	5	2	2	3	9	2
9	8	6	6	4	13	3	5	32	260	12	1	12	8	3	7	3	4	1	7	3
10	7	1	2	2	11	0	19	10	11	262	47	3	7	0	2	2	2	2	4	3
11	8	7	0	1	9	4	16	8	4	36	284	1	3	0	3	1	4	1	7	2
12	15	10	3	5	9	5	8	2	7	9	2	253	14	5	3	2	15	5	21	3
13	11	14	12	17	35	8	16	13	16	10	6	9	187	4	12	5	1	3	7	7
14	16	16	2	7	30	9	5	18	5	6	8	7	30	168	12	14	4	4	31	4
15	10	24	2	6	7	4	8	6	8	13	2	14	8	12	241	4	4	3	13	5
16	41	8	2	5	0	1	0	2	4	3	1	4	3	2	5	261	0	2	6	48
17	6	7	1	2	4	2	1	8	0	9	4	16	4	2	5	3	244	7	23	16
18	46	0	1	9	4	1	0	4	2	5	4	7	3	3	2	5	13	235	20	12
19	3	3	0	2	5	1	1	12	11	5	4	8	4	6	5	4	60	8	146	22
20	50	0	1	4	2	4	0	3	2	3	3	2	2	2	1	40	18	3	8	103

Table 5.5: Confusion Matrix : 1-NN after k-Means using cosine distance (Accuracy : 56.34)

The confusion matrix on applying 3-NN ($k_2 = 3$) after making 200 clusters using k-Means and using cosine distance as distance measure based on algorithm 5.2 is given in table 5.6.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	176	5	1	2	5	7	3	5	2	10	0	3	1	5	3	46	6	5	4	30
2	6	183	31	17	29	40	9	8	3	6	0	12	21	1	4	2	2	2	6	7
3	5	22	201	26	20	45	24	4	2	6	1	4	7	2	5	3	4	1	5	7
4	4	21	33	187	57	10	22	4	3	13	2	8	18	3	1	1	0	1	2	2
5	4	15	14	69	196	8	15	4	2	10	0	3	24	5	3	2	1	3	3	4
6	4	51	52	9	25	174	12	5	6	7	2	7	12	7	8	3	2	2	5	2
7	2	12	5	38	29	8	221	13	6	5	1	11	18	4	2	4	0	0	8	3
8	6	8	2	9	16	2	14	267	16	8	1	3	20	1	2	2	3	3	10	3
9	7	6	5	7	13	4	6	32	251	13	1	12	7	5	7	4	3	1	7	7
10	7	1	2	2	11	0	16	10	12	252	57	3	7	0	2	2	2	4	4	3
11	8	7	1	0	9	4	19	11	2	38	279	1	3	0	2	1	4	1	7	2
12	18	12	5	4	6	7	7	3	6	9	2	250	14	2	4	2	14	6	22	3
13	12	11	13	22	29	12	14	13	19	10	6	10	179	7	12	4	3	3	6	8
14	19	17	2	8	29	11	7	14	4	9	5	7	25	164	13	15	4	4	34	5
15	13	23	1	6	9	4	9	4	8	13	3	16	12	10	234	3	6	3	13	4
16	38	8	4	5	1	2	1	2	3	3	1	3	3	2	5	255	0	2	6	54
17	7	8	2	2	3	2	3	6	1	10	3	13	3	2	3	3	248	9	21	15
18	48	0	0	9	4	2	0	4	1	6	2	7	4	3	2	4	16	231	20	13
19	8	3	0	1	5	1	1	10	8	5	3	10	5	5	7	2	71	12	141	12
20	47	0	0	4	2	4	0	3	2	4	3	0	2	2	1	42	19	3	10	103

Table 5.6: Confusion Matrix : 3-NN after k-Means using cosine distance (Accuracy : 55.66)

5.3.3 k_2-NN after k_1-Means and Cluster Selection

Another algorithm that has been used to assign the class to a new test document doc_i after performing k_1-Means clustering is to choose k_3 closest clusters, then choose a cluster among these k_3 clusters where majority of k_4 closest documents to doc_i lie. Then perform k_2-NN on that cluster. The algorithm for this is algorithm 5.3. Since, during the experiments, the values of k_1, k_3 and k_4 are always taken as 200, 3 and 5 respectively, these values are directly used in the algorithm.

Algorithm 5.3: k_2-NN after k_1-Means and cluster selection

1. Find $200(k_1)$ clusters using k-Means clustering on the training dataset ;
2. For a new document doc_i, find $3(k_3)$ clusters seeds closest to it and let these clusters be C_1, C_2 and C_3 ;
3. Find $5(k_4)$ documents from $C_1 \cup C_2 \cup C_3$ that are most similar to doc_i ;
4. Assign doc_i the cluster C which the majority of these $5(k_4)$ documents belong to. In case of ties, assign the cluster of the document which is most similar to doc_i among the documents involved in the tie ;
5. Find k_2 documents from cluster C that are most similar to doc_i ;
6. Assign doc_i the class which the majority of these k_2 documents belong to. In case of ties, assign the class of the document which is most similar to doc_1 among the documents involved in the tie ;

We do not have any results for algorithm 5.3. This algorithm has been mentioned separately as it would make it easier to explain algorithm 5.4 in the next section.

5.3.4 k_2-NN after k_1-Means, cluster selection and removal of up to 10% documents from each cluster

A modification of algorithm 5.3 is to remove upto 10% documents from each cluster before performing k-NN. This algorithm is algorithm 5.4. The idea is to find the class which has the lowest number of document in each cluster and remove all the documents from that class if removal does not lead to reduction of the original cluster more than 10%. This removal of documents is done from each cluster until no more reduction can be done following the above condition. A demonstration of this can be done using a dummy example of 10 clusters formed by 5 classes containing 100 documents each as shown in table 5.7.

	Class 1	**Class 2**	**Class 3**	**Class4**	**Class 5**	**Total**
Cluster 1	20	**5**	0	76	**3**	**104**
Cluster 2	3	13	18	**2**	4	**40**
Cluster 3	12	6	9	**3**	3	**33**
Cluster 4	13	16	21	9	12	**71**
Cluster 5	2	10	6	**1**	**1**	**20**
Cluster 6	**4**	24	31	0	**2**	**61**
Cluster 7	**1**	9	4	3	0	**17**
Cluster 8	**3**	5	4	**2**	67	**81**
Cluster 9	2	9	5	0	2	**18**
Cluster 10	40	**3**	**2**	4	6	**55**
Total	**100**	**100**	**100**	**100**	**100**	**500**

Table 5.7: Dummy Clusters for Illustration

In the clusters given in table 5.7, for cluster 1, 3 documents of class 5 will be removed first and then 5 documents of class 2 will be removed. No further documents can be removed from cluster 1 as that would lead t removal of more than 10% of documents in cluster 1, that is, no more than 10.4 documents can be removed from cluster 1. No documents can be removed from cluster 4 as removal of 9 documents of class 4 (which is minimum) will lead to removal of more than 10% documents from that cluster. All the documents corresponding to the bold entries will be removed from the clusters. After removal of these docuemnts from the clusters, the algorithm 5.4 is same as that of algorithm 5.3.

Algorithm 5.4: k_2-NN after k_1-Means, cluster selection and removal of up to 10% documents from each cluster

1 Find 200(k_1) clusters using k-Means clustering on the training dataset ;
2 **for** *Each cluster* C_i **do**
3 Note the cluster size $size_i$;
4 Find one of the class *selectedClass* whose document appears the least number of times in C_i ;
5 **if** *(size of* C_i *- Number of documents belonging to selectedClass in* C_i*)* $\geq$ *(*$0.9 \times size_i$*)* **then**
6 Removal of all documents belonging *selectedClass* from cluster C_i ;
7 Go to step 4 ;
8 **end**
9 **end**
10 For a new document doc_i, find 3(k_3) clusters seeds closest to it and let these clusters be C_1, C_2 and C_3 ;
11 Find 5(k_4) documents from $C_1 \cup C_2 \cup C_3$ that are most similar to doc_i ;
12 Assign doc_i the cluster C which the majority of these 5(k_4) documents belong to. In case of ties, assign the cluster of the document which is most similar to doc_i among the documents involved in the tie ;
13 Find k_2 documents from cluster C that are most similar to doc_i ;
14 Assign doc_i to the class which the majority of these k_2 documents belong to. In case of ties, assign the class of the document which is most similar to doc_1 among the documents involved in the tie ;

The confusion matrix on applying 1-NN ($k_2 = 1$) after making 200 clusters using k-Means, reducing upto 10% from each cluster, using cosine distance as distance measure based on algorithm 5.4 is given in table 5.8.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	187	5	0	0	4	5	4	4	1	5	0	4	0	4	4	44	4	6	9	29
2	4	203	32	14	23	41	9	6	1	8	2	13	16	0	4	4	0	0	5	4
3	2	27	218	23	14	44	19	1	2	0	2	6	9	1	5	2	1	4	7	7
4	1	14	31	211	42	9	31	3	4	7	1	5	23	4	1	1	0	0	3	1
5	3	20	13	57	207	6	16	3	2	8	0	3	24	6	3	2	2	4	1	5
6	3	51	47	6	18	205	9	7	4	5	1	6	6	7	9	2	1	0	6	2
7	2	11	3	38	38	6	215	19	9	4	3	9	14	6	2	4	2	0	4	1
8	1	8	2	3	11	1	10	283	23	6	1	4	19	3	3	2	2	3	8	3
9	1	6	3	5	10	1	2	31	287	4	0	9	14	3	5	1	5	2	5	4
10	5	0	1	0	9	0	19	7	7	283	47	3	3	1	1	1	1	2	3	4
11	4	8	0	1	3	2	10	7	6	21	323	1	2	0	3	1	0	1	3	3
12	8	5	5	3	10	3	2	1	2	6	3	310	4	3	0	1	11	5	12	2
13	7	14	13	22	36	6	13	15	8	7	6	10	198	6	10	6	2	2	7	5
14	11	16	2	4	25	9	5	13	8	8	9	5	21	197	5	17	3	7	27	4
15	4	23	3	2	7	0	3	6	5	11	2	9	3	10	282	1	5	1	13	4
16	36	8	1	3	0	1	0	3	1	5	0	2	3	5	2	277	0	3	5	43
17	6	4	0	1	4	3	1	4	1	4	1	18	2	3	3	3	261	4	22	19
18	55	2	1	5	1	0	0	2	1	5	3	6	2	1	1	2	10	252	10	17
19	5	1	0	1	4	2	0	2	4	3	4	3	1	6	0	3	84	6	164	17
20	48	3	0	2	1	3	0	3	3	3	3	1	2	2	1	35	17	1	9	114

Table 5.8: Confusion Matrix : 1-NN after k-Means and reduction of 10% documents from each cluster using cosine distance (Accuracy : 62.09)

The confusion matrix on applying 1-NN ($k_2 = 1$) after making 200 clusters using k-Means, reducing upto 10% from each cluster, using Jaccard distance as distance measure based on algorithm 5.4 is given in table 5.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	201	2	0	0	2	0	0	1	1	3	1	3	0	7	8	41	10	5	2	32
2	6	213	32	15	17	38	7	3	3	7	1	9	18	5	4	7	0	2	0	2
3	7	31	253	26	10	22	10	2	2	2	4	5	9	2	3	0	0	1	2	3
4	1	10	35	227	38	7	30	5	0	3	0	1	22	2	5	1	0	1	2	2
5	2	10	24	63	204	5	20	2	4	4	3	2	24	6	7	0	0	1	2	2
6	3	51	66	7	9	214	7	6	3	3	0	2	5	2	9	0	4	1	2	1
7	1	9	6	24	28	0	275	11	9	4	1	0	10	4	0	5	0	1	1	1
8	3	6	5	6	6	1	10	293	16	5	3	4	17	4	7	2	3	0	4	1
9	3	4	1	2	6	1	8	27	310	6	3	1	7	1	2	1	9	2	4	0
10	5	4	1	2	4	2	5	6	4	320	31	0	5	1	1	1	1	2	0	2
11	5	5	0	1	0	0	6	5	1	25	340	1	2	1	3	1	1	0	2	0
12	1	4	7	2	10	5	0	4	3	4	1	317	3	9	6	3	7	4	4	2
13	1	20	21	29	21	12	20	12	19	8	3	8	183	5	10	5	4	1	4	7
14	14	15	11	13	10	13	6	8	7	9	5	5	19	204	11	12	5	3	20	6
15	4	17	3	0	4	2	4	7	5	3	2	2	10	7	298	4	6	3	11	2
16	26	6	3	2	1	1	0	2	2	2	1	2	6	1	2	282	3	4	12	40
17	4	6	3	3	0	0	3	1	2	4	1	12	4	2	3	0	285	3	10	18
18	29	4	2	3	2	1	1	1	2	8	1	5	2	0	2	9	10	276	14	4
19	6	2	0	1	2	1	1	2	2	5	1	3	0	5	5	8	72	3	171	20
20	41	2	2	1	0	1	2	2	2	3	0	0	4	4	6	42	16	0	5	118

Table 5.9: Confusion Matrix : 1-NN after k-Means and reduction of 10% documents from each cluster using Jaccard distance (Accuracy : 66.17)

The confusion matrix on applying 3-NN ($k_2 = 3$) after making 200 clusters using k-Means, reducing upto 10% from each cluster, using cosine distance as distance measure based on algorithm 5.4 is given in table 5.10.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	180	5	0	0	6	5	1	4	1	6	0	2	0	5	4	49	6	7	8	30
2	4	200	30	18	27	38	9	6	1	8	2	11	16	1	4	3	0	1	5	5
3	3	24	223	24	14	43	20	1	2	1	1	6	8	1	4	1	1	4	6	7
4	1	14	32	216	40	9	27	5	3	7	2	6	21	3	1	1	0	0	3	1
5	5	17	14	74	189	7	14	5	2	8	0	3	26	6	3	1	2	3	1	5
6	3	48	48	4	21	205	10	6	4	6	2	7	7	6	8	2	3	0	4	1
7	2	12	6	47	32	6	207	16	8	5	2	10	19	3	2	3	1	0	5	4
8	1	9	2	3	12	3	9	287	19	6	0	2	20	2	1	2	2	3	9	4
9	2	7	3	7	10	1	2	33	277	5	0	9	13	4	5	1	4	2	5	8
10	6	0	1	0	9	0	16	7	7	276	56	3	3	1	1	1	1	2	3	4
11	4	6	1	0	5	2	14	11	3	21	320	1	2	0	2	1	0	0	3	3
12	9	5	7	2	8	3	3	2	2	6	3	309	5	1	1	1	11	5	11	2
13	7	13	16	27	31	7	9	17	11	7	6	10	191	7	10	6	4	2	7	5
14	11	17	2	5	25	10	5	11	6	10	7	6	18	198	6	17	3	7	29	3
15	6	22	2	3	7	1	4	5	6	10	2	10	8	9	275	2	6	1	12	3
16	38	6	2	3	0	2	1	3	1	5	0	3	4	5	1	271	0	3	5	45
17	8	5	1	2	3	2	3	2	0	5	0	16	2	3	2	3	270	4	17	16
18	57	2	0	5	1	0	0	2	1	6	1	6	3	1	1	2	12	248	10	18
19	8	1	0	0	4	2	0	2	3	5	3	6	2	5	1	1	90	6	161	10
20	43	2	0	2	1	3	1	3	3	4	3	0	2	2	1	43	16	1	9	112

Table 5.10: Confusion Matrix : 3-NN after k-Means and reduction of 10% documents from each cluster using cosine distance (Accuracy : 61.27)

The confusion matrix on applying 3-NN ($k_2 = 3$) after making 200 clusters using k-Means, reducing upto 10% from each cluster, using Jaccard distance as distance measure based on algorithm 5.4 is given in table 5.11.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	191	2	0	0	2	0	0	1	1	3	1	3	0	7	9	45	9	5	2	38
2	6	208	38	15	18	37	6	2	4	4	1	8	20	4	6	9	0	1	0	2
3	9	30	262	27	10	18	6	2	2	1	4	6	5	2	4	0	0	0	2	4
4	1	15	41	218	36	9	30	3	1	2	0	1	22	2	5	1	0	2	1	2
5	2	12	23	68	199	7	17	3	2	4	2	2	23	6	8	0	0	1	3	3
6	4	51	72	7	10	208	5	6	3	3	0	3	6	2	9	0	4	1	1	0
7	1	8	6	25	24	0	277	15	6	5	1	0	11	4	0	4	0	1	1	1
8	3	6	6	5	6	1	11	295	11	6	3	3	21	2	5	2	5	0	4	1
9	2	5	1	3	3	1	7	30	310	5	3	1	8	2	2	2	9	2	2	0
10	5	2	1	1	5	2	5	6	5	319	34	0	4	1	2	1	1	2	0	1
11	6	6	0	0	0	0	5	4	1	24	340	1	2	2	3	1	1	1	2	0
12	2	6	6	2	8	6	0	3	4	5	1	316	4	5	6	3	10	4	3	2
13	1	22	26	30	22	8	19	13	21	8	3	11	173	5	9	7	4	1	3	7
14	12	16	10	12	12	14	5	9	6	11	7	4	16	202	11	13	4	4	24	4
15	3	20	1	3	1	1	4	7	6	2	3	2	13	9	295	2	8	2	10	2
16	33	4	3	2	1	1	0	1	3	3	1	1	6	1	3	274	3	3	12	43
17	5	6	3	3	0	1	3	1	2	4	1	10	4	1	3	1	297	3	8	8
18	31	5	2	3	2	1	1	1	2	9	1	4	2	0	2	10	10	271	16	3
19	7	3	0	0	2	1	1	2	1	7	1	3	0	4	6	5	78	6	168	15
20	40	3	3	0	0	1	1	2	2	3	0	0	3	5	6	50	19	0	4	109

Table 5.11: Confusion Matrix : 3-NN after k-Means and reduction of 10% documents from each cluster using Jaccard distance (Accuracy : 65.48)

The confusion matrix on applying 5-NN ($k_2 = 5$) after making 200 clusters using k-Means, reducing upto 10% from each cluster, using cosine distance as distance measure based on algorithm 5.4 is given in table 5.12.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	178	5	0	0	5	5	2	5	2	5	1	2	0	4	4	51	9	3	8	30
2	8	198	32	15	35	41	9	4	3	5	2	9	11	2	5	2	0	1	3	4
3	6	20	218	26	15	49	21	1	2	3	2	5	5	0	4	0	2	4	6	5
4	1	13	31	212	53	13	20	3	3	7	2	5	17	1	3	1	0	2	4	1
5	4	14	17	68	196	11	10	4	2	9	2	2	27	5	3	1	1	3	1	5
6	3	43	52	5	29	198	6	5	5	8	4	5	10	3	8	1	2	2	5	1
7	1	10	10	46	35	12	206	17	6	6	4	6	15	1	1	3	1	2	5	3
8	1	9	5	2	11	2	11	280	18	3	2	4	22	3	0	2	2	4	11	4
9	7	2	2	8	11	1	1	35	276	7	5	9	9	1	3	1	5	4	6	5
10	5	1	1	1	6	0	22	5	7	281	46	3	5	0	1	2	1	2	4	4
11	3	4	1	0	5	2	14	14	2	24	320	0	1	1	1	1	0	0	3	3
12	12	7	6	1	10	2	3	2	2	5	2	298	5	1	4	2	17	4	11	2
13	10	13	13	35	33	9	11	15	9	7	8	10	178	8	12	5	3	2	7	5
14	11	13	3	2	27	9	5	10	8	9	13	6	19	191	8	15	7	6	30	4
15	5	19	2	2	8	1	3	5	6	11	4	9	10	7	276	1	8	2	12	3
16	45	5	2	3	0	2	0	1	1	5	0	4	4	4	2	280	1	4	6	29
17	7	4	1	2	4	2	2	2	3	7	0	16	1	2	4	4	266	4	22	11
18	58	5	0	5	1	0	0	1	1	7	1	3	2	1	1	5	18	242	11	14
19	6	4	0	0	4	2	0	2	3	7	2	3	2	5	0	1	89	9	162	9
20	51	3	0	1	2	2	0	4	1	6	2	0	1	2	2	41	22	2	11	98

Table 5.12: Confusion Matrix : 5-NN after k-Means and reduction of 10% documents from each cluster using cosine distance (Accuracy : 60.46)

The confusion matrix on applying 10-NN ($k_2 = 10$)after making 200 clusters using k-Means, reducing upto 10% from each cluster, using cosine distance as distance measure based on algorithm 5.4 is given in table 5.13.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	174	5	0	0	5	1	3	2	2	5	1	4	0	10	7	60	8	3	9	20
2	4	198	35	13	32	41	7	6	2	6	3	6	14	2	6	3	0	2	5	4
3	7	16	233	31	14	45	8	4	0	4	3	3	5	1	3	0	2	0	7	8
4	1	9	33	209	68	11	16	5	2	8	1	3	19	0	2	0	0	1	2	2
5	4	12	22	72	198	8	12	3	2	7	2	1	14	6	5	5	1	4	2	5
6	1	45	59	4	31	185	7	3	8	14	5	4	9	0	4	5	2	3	4	2
7	1	15	5	56	45	7	190	16	9	8	5	4	14	2	1	6	2	1	2	1
8	1	8	3	3	12	2	9	273	21	5	3	1	22	3	4	2	3	4	14	3
9	11	2	3	14	8	0	4	28	278	13	4	4	8	2	1	0	5	1	7	5
10	8	1	1	0	7	0	17	3	8	284	51	0	2	0	2	5	3	2	1	2
11	5	6	2	0	4	1	3	13	2	23	333	0	1	0	1	1	0	1	2	1
12	12	7	6	0	14	3	5	6	2	3	0	293	9	0	3	2	14	2	13	2
13	7	19	12	36	33	14	16	16	28	12	6	11	144	7	9	8	3	3	5	4
14	14	13	3	2	29	6	6	9	8	17	10	7	25	185	6	12	8	6	28	2
15	6	23	3	2	14	2	4	5	6	13	3	7	26	6	250	3	5	2	12	2
16	44	6	4	0	0	1	2	3	4	5	0	3	5	3	3	283	2	3	7	20
17	12	5	1	1	4	1	1	2	2	9	0	13	4	2	3	6	270	4	20	4
18	68	5	0	1	1	2	0	1	1	7	1	5	5	0	0	3	16	242	11	7
19	8	4	0	0	3	0	0	3	4	6	2	4	4	5	0	2	90	5	162	8
20	50	1	2	0	4	0	0	4	1	7	2	0	3	5	1	52	26	2	12	79

Table 5.13: Confusion Matrix : 10-NN after k-Means and reduction of 10% documents from each cluster using cosine distance (Accuracy : 59.25)

5.3.5 Class Association

Class association is defined between each pair classes. For this, first n clusters, $cluster_1, cluster_2, \ldots, cluster_n$ are created using k-Means clustering. The formula used to find the class association between $class_i$ and $class_j$ is given in equation 5.1.

$$Class\ Association = \frac{(\sum_{k=1}^{n} min(numDoc_{i,k}, numDoc_{i,k})) \times 100}{\frac{Total\ Number\ of\ Documents}{2}} \tag{5.1}$$

In equation 5.1, $numDoc_{i,k}$ denotes the number of documents belonging to $class_i$ in $cluster_k$. 11314 is the total number of documents in the training set.

The class association between each pair of classes is shown table 5.14. The matrix of class association is always a symmetric matrix.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	8.48	1.25	0.86	0.93	0.98	1.18	1.04	1.80	1.83	1.69	0.95	1.55	1.41	1.85	1.97	3.65	2.13	1.87	2.45	4.01
2	1.25	10.3	4.06	3.69	3.11	4.43	3.11	1.97	2.12	1.82	1.30	1.73	3.51	1.80	1.97	1.20	1.18	1.11	1.21	0.95
3	0.86	4.06	10.4	4.10	3.76	**5.10**	3.18	1.62	1.43	1.14	0.68	1.57	2.51	1.55	1.46	0.81	0.76	0.76	0.77	0.65
4	0.93	3.69	4.10	10.4	**6.13**	2.66	3.83	1.80	1.71	1.44	1.00	1.82	3.64	1.73	1.41	0.88	1.04	0.83	0.97	0.74
5	0.98	3.11	3.76	**6.13**	10.2	2.72	3.87	2.01	1.96	1.69	1.11	1.43	3.92	1.92	1.55	0.84	1.06	0.84	1.02	0.77
6	1.18	4.43	**5.10**	2.66	2.72	10.4	1.94	1.67	1.34	1.50	1.09	1.57	2.52	1.76	1.66	1.14	1.09	1.06	0.91	0.90
7	1.04	3.11	3.18	3.83	3.87	1.94	10.3	2.51	2.33	2.29	1.83	1.55	3.78	1.71	1.78	1.27	1.23	0.81	1.21	0.84
8	1.80	1.97	1.62	1.80	2.01	1.67	2.51	10.5	3.72	1.99	1.27	1.60	2.93	2.51	2.70	1.44	1.78	1.34	2.03	1.53
9	1.83	2.12	1.43	1.71	1.96	1.34	2.33	3.72	10.5	2.29	1.20	1.36	2.38	1.92	2.26	1.18	1.73	1.43	2.10	1.50
10	1.69	1.82	1.14	1.44	1.69	1.50	2.29	1.99	2.29	10.5	3.67	1.14	2.08	1.85	1.89	1.27	1.41	1.21	2.08	1.30
11	0.95	1.30	0.68	1.00	1.11	1.09	1.83	1.27	1.20	3.67	10.6	0.97	1.41	1.14	1.29	0.76	1.16	0.95	1.21	0.74
12	1.55	1.73	1.57	1.82	1.43	1.57	1.55	1.60	1.36	1.14	0.97	10.5	2.33	1.55	1.62	1.39	2.20	1.32	2.31	1.27
13	1.41	3.51	2.51	3.64	3.92	2.52	3.78	2.93	2.38	2.08	1.41	2.33	10.4	2.45	2.56	1.75	1.37	1.30	1.41	1.13
14	1.85	1.80	1.55	1.73	1.92	1.76	1.71	2.51	1.92	1.85	1.14	1.55	2.45	10.5	2.70	1.80	1.52	1.36	1.71	1.50
15	1.97	1.97	1.46	1.41	1.55	1.66	1.78	2.70	2.26	1.89	1.29	1.62	2.56	2.70	10.4	1.46	1.62	1.67	1.92	1.46
16	3.65	1.20	0.81	0.88	0.84	1.14	1.27	1.44	1.18	1.27	0.76	1.39	1.75	1.80	1.46	10.5	1.39	1.50	1.73	3.95
17	2.13	1.18	0.76	1.04	1.06	1.09	1.23	1.78	1.73	1.41	1.16	2.20	1.37	1.52	1.62	1.39	9.65	1.85	3.18	2.05
18	1.87	1.11	0.76	0.83	0.84	1.06	0.81	1.34	1.43	1.21	0.95	1.32	1.30	1.36	1.67	1.50	1.85	9.96	1.80	1.59
19	2.45	1.21	0.77	0.97	1.02	0.91	1.21	2.03	2.10	2.08	1.21	2.31	1.41	1.71	1.92	1.73	3.18	1.80	8.21	2.03
20	4.01	0.95	0.65	0.74	0.77	0.90	0.84	1.53	1.50	1.30	0.74	1.27	1.13	1.50	1.46	3.95	2.05	1.59	2.03	6.66

Table 5.14: Class Association Matrix

The class association matrix seems to agree with the different confusion matrices as large number of misclassifications in the methods we have implemented are between classes with higher class association.

Chapter 6

Hierarchical Agglomerative Clustering

6.1 Introduction

Hierarchical clustering algorithms are either top-down or bottom-up. Bottom-up algorithms treat each document as a singleton cluster (cluster of size one or cluster containing only one document) at the outset and then successively merge (or agglomerate) pairs of clusters until all clusters have been merged into a single cluster that contains all documents. Bottom-up hierarchical clustering is therefore called hierarchical agglomerative clustering or HAC. Top-down clustering requires a method for splitting a cluster. It proceeds by splitting clusters recursively until individual documents are reached [5].

An agglomerative algorithm starts with disjoint clustering, placing each of the n documents in an individual cluster. It successively combines pairs of clusters until finally reaching the point where all documents form one large cluster. In each of the subsequent steps, the two closest clusters will merge. The others remain unchanged [1].

6.2 Algorithm

Algorithm 6.1: Hierarchical Agglomerative Clustering

```
1 Start by assigning each document to a cluster ;
2 Find the closest (most similar) pair of clusters ;
3 if similarity between closest clusters > threshold then
4 |   Note the clusters ;
5 |   End ;
6 else
7 |   Merge the closest (most similar) pair of clusters ;
8 |   Go to Step 2;
9 end
```

6.3 Types of Agglomerative Hierarchical Clustering

Based on the technique to measure the similarity or distance between two clusters, agglomerative hierarchical clustering can be of three types:

- Single Linkage Clustering
- Complete Linkage Clustering
- Average Linkage Clustering

In this section, $clustDist_{p,q}$ will represent the distance between two clusters, $cluster_p$ and $cluster_q$ and $docDist_{i,j}$ will represent the distance between doc_i and doc_j where documents follow the notation as given in section 3.1.2.

6.3.1 Single Linkage Clustering

In the Single-Link method the distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters. This procedure will string objects together to form clusters and the resulting clusters tend to appear as straggly, elongated chains [1].

The distance between two clusters in single linkage clustering is :

$$clustDist_{p,q} = \min_{i,j}(docDist_{i,j}) \mid doc_i \in cluster_p \; and \; doc_j \in cluster_q$$

6.3.2 Complete Linkage Clustering

In the Complete-Link Method, the distances between clusters are determined by the greatest distance between any two objects in the different clusters (i.e., by the "furthest neighbors"). Those two clusters are merged for which the distance between their furthest objects is minimal. This method usually performs quite well in cases where the objects naturally form distinct clumps. If the clusters tend to be somehow elongated or of a chain-type, then this method is inappropriate. The clusters obtained by the Complete-Link algorithm are more compact than those obtained by the Single-Link algorithm [1].

The distance between two clusters in complete linkage clustering is :

$$clustDist_{p,q} = \max_{i,j}(docDist_{i,j}) \mid doc_i \in cluster_p \; and \; doc_j \in cluster_q$$

6.3.3 Average Linkage Clustering

In the Average-Link Method the distance between two clusters is computed as the average of the distances between all points in these clusters. This approach can be regarded as a compromise between the Single- and the Complete-Link Method. Furthermore it is less sensitive to outliers. This means that if an object is quite distinct from the other ones i.e., lies far away from the cluster centroid - this is not likely to skew the clustering result [1].

The distance between two clusters in average linkage clustering, denoting $clustSize_k$ as the size (number of documents in) cluster $cluster_k$, is :

$$clustDist_{p,q} = \frac{\sum_{i,j}(docDist_{i,j})}{clustSize_p \times clustSize_q} \mid doc_i \in cluster_p \; and \; doc_j \in cluster_q$$

6.4 Experiment

The above three hierarchical agglomerative clustering was performed on each class in order to find the clusters within a class. The similarity measure used was cosine similarity. For a class the clusters were formed at different threshold values and results were stored at intervals of 0.05. This was done for each for each class three times (applying the three methods namely single linkage, complete linkage and average linkage) and conclusions were drawn from these results.

Results for single linkage clustering at threshold 0.3 using cosine distance as distance measure is shown in table 6.1.

Class Number	**Size of largest cluster**	**Size of second largest cluster**	**Number of clusters of size 1**	**Number of clusters of size 2**	**Number of clusters of size 3**
1	395	9	52	7	2
2	375	11	104	15	6
3	456	10	103	14	5
4	418	6	140	20	3
5	454	4	78	15	4
6	386	8	103	29	6
7	239	11	192	29	7
8	391	7	102	19	7
9	470	7	73	13	4
10	468	5	68	14	3
11	525	3	55	7	2
12	518	4	55	6	2
13	124	28	137	30	4
14	336	11	102	25	9
15	408	10	78	15	7
16	492	6	68	9	2
17	446	4	54	16	2
18	496	7	33	8	1
19	327	10	54	14	10
20	577	14	122	25	13

Table 6.1: Single Linkage Clustering at Threshold 0.3 using Cosine Similarity

Results for complete linkage clustering at threshold 0.05 using cosine distance as distance measure is shown in table 6.2.

Class Number	Size of largest cluster	Size of second largest cluster	Number of clusters of size 1	Number of clusters of size 2	Number of clusters of size 3
1	44	28	1	6	4
2	31	22	4	11	12
3	65	60	4	14	6
4	42	39	2	16	12
5	22	21	2	11	9
6	35	21	2	15	18
7	23	22	4	13	17
8	60	28	0	13	16
9	35	26	3	6	11
10	33	32	1	12	8
11	32	24	2	7	13
12	73	40	7	5	1
13	19	18	4	16	25
14	56	41	1	18	25
15	30	27	1	8	13
16	45	22	2	10	9
17	50	37	2	5	5
18	103	61	4	5	3
19	28	26	3	2	5
20	34	29	3	9	14

Table 6.2: Complete Linkage Clustering at Threshold 0.05 using Cosine Similarity

Results for average linkage clustering at threshold 0.1 using cosine distance as distance measure is shown in table 6.3.

Class Number	Size of largest cluster	Size of second largest cluster	Number of clusters of size 1	Number of clusters of size 2	Number of clusters of size 3
1	106	67	5	8	3
2	65	49	11	12	5
3	251	73	8	14	3
4	190	41	14	12	6
5	64	45	12	11	2
6	77	58	13	16	13
7	44	40	17	17	10
8	208	41	14	9	10
9	182	34	10	7	6
10	214	48	8	9	4
11	162	81	8	9	4
12	356	41	5	8	2
13	24	23	13	17	13
14	99	55	15	10	18
15	108	51	11	9	9
16	294	44	23	8	3
17	188	97	9	6	0
18	262	156	8	3	1
19	73	65	5	6	5
20	98	73	17	12	8

Table 6.3: Average Linkage Clustering at Threshold 0.1 using Cosine Similarity

Further work can be continued after this by finding and removing the outliers (documents in clusters of small sizes) from each class and then applying a method on the training set after removal of these outliers to classify a new document.

Chapter 7

Naive Bayes Classifier

7.1 Introduction

Naive Bayes has been one of the popular machine learning methods for many years. Bayesian or probabilistic classifiers have been widely used for text categorization [2].

Given a set of n documents $S = \{doc_1, doc_2, \ldots, doc_n\}$, classified along a set C of m classes, $C = \{C_1, C_2, \ldots, C_m\}$, Bayesian classifiers estimate the probabilities of each class C_k given a document doc_i as given in equation 7.1.

$$P(C_k|doc_i) = \frac{P(C_k)P(doc_i|C_k)}{P(doc_i)} \tag{7.1}$$

In the equation 7.1, $P(doc_i)$ is the probability that a randomly picked document doc_j, and $P(C_k)$ the probability that any randomly picked document belongs to C_k and is also called the prior probability of class C_k.

It is very difficult to compute $P(doc_i|C_k)$ as the number of possible documents doc_i is very high. Hence, Naive Bayes assumes that the probability of a given word or term is independent of other terms that appear in the same document. Using this assumption, it is possible to determine $P(doc_i|C_k)$ as the product of the probabilities of each term that appears in the document.

Given a document

$$doc_i = \{termid_{i,1}, termid_{i,2}, \ldots, termid_{i,n_i}, freq_{i,1}, freq_{i,2}, \ldots, freq_{i,n_i}\}$$

(following notations as given in section 3.1.2), the $P(doc_i|C_k)$ can be given by equation 7.2.

$$P(doc_i|C_k) = \prod_{j=1}^{n_i} P(termid_{i,j}|C_k)^{freq_{i,j}} \tag{7.2}$$

The $P(termid_{i,j}|C_k)$ can be calculated by dividing the number of times $termid_{i,j}$ appeared in class C_k by number of times any term appeared in class C_k. To make sure that $P(termid_{i,j}|C_k)$ is never equal to 0 which in turn will cause $P(doc_i|C_k)$ to be 0, we need to add one to frequency of all the terms in all the training documents thus assuming the term that has never appeared in a document to have appeared once.

The probability $P(C_k)$ is known as the prior probability of the class C_k and can be calculated by dividing the number of words in class C_k by number of words that appeared in all the classes combined and can be calculated as given in equation 7.3.

$$P(C_k) = \frac{\sum_{doc_i \in C_k} \sum_{j=1}^{n_i} freq_{i,j}}{\sum_{doc_i} \sum_{j=1}^{n_i} freq_{i,j}} \tag{7.3}$$

The class C_i to be assigned to a new document doc_i can be computed by equation 7.4

$$\begin{aligned}
C_i &= argmax_{C_k} P(C_k|doc_i) \\
&= argmax_{C_k} \{\frac{P(C_k)P(doc_i|C_k)}{P(doc_i)}\} \\
&= argmax_{C_k} \{P(C_k)P(doc_i|C_k)\} \\
&= argmax_{C_k} \{\log(P(C_k)P(doc_i|C_k))\} \\
&= argmax_{C_k} \{\log P(C_k) + \log P(doc_i|C_k)\} \\
&= argmax_{C_k} \{\log P(C_k) + \log(\prod_{j=1}^{n_i} P(termid_{i,j}|C_k)^{freq_{i,j}})\} \\
&= argmax_{C_k} \{\log P(C_k) + \sum_{j=1}^{n_i} freq_{i,j} \times \log(P(termid_{i,j}|C_k))\}
\end{aligned} \tag{7.4}$$

In order to classify a new document doc_i Naive Bayes classifier finds a class for which the $\log P(C_k) + \sum_{j=1}^{n_i} freq_{i,j} \times \log(P(termid_{i,j}|C_k))$ is maximum and assigns the document doc_i to that class.

7.2 Experiments and Results

7.2.1 Naive Bayes

The basic algorithm for Naive Bayes classifier is given in algorithm 7.1. For a new document doc_i we need to find probabilities $P(C_j|doc_i)$ for all possible classes C_j and assign the document doc_i to the class whose probability is maximum among the above probabilities.

Algorithm 7.1: Naive Bayes Classifier

```
Given a document, doc_i ;
for Each Class C_i do
    Calculate x_i = log P(C_k) + Σ_{j=1}^{n_i} freq_{i,j} × log(P(termid_{i,j}|C_k)) ;
end
Find class C for which x_i is maximum ;
Assign doc_i to class C ;
```

The confusion matrix for Naive Bayes as given in algorithm 7.1 is shown in table 7.1.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	221	0	0	0	0	0	0	0	1	0	1	2	0	5	3	55	6	10	8	7
2	0	310	0	11	9	20	0	0	0	0	0	22	3	3	7	0	0	2	2	0
3	3	69	4	126	15	130	0	1	0	0	0	18	2	2	7	0	3	0	13	1
4	0	11	1	307	23	7	7	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	1	22	302	3	5	5	0	0	0	7	11	2	11	0	2	0	2	0
6	0	37	2	10	3	326	2	0	0	0	1	3	1	1	6	0	1	1	1	0
7	0	4	0	47	17	7	245	22	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	3	366	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	378	3	2	0	0	4	2	4	0
13	0	18	0	20	6	1	3	14	2	0	0	52	250	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	2	0	0	0	0	0	0	0	0	0	0	0	2	2	371	3	1	3	7
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	0	0	1	0	0	0	0	1	0	4	0	1	8	1	102	4	184	0
20	42	3	0	0	0	0	0	0	0	0	0	3	0	2	6	68	25	7	6	89

Table 7.1: Confusion Matrix : Naive Bayes (Accuracy : 77.79%)

From the looks of the confusion matrix given in table 7.1, it appeared as if class 3 was responsible for most misclassifications and yet the Naive Bayes classifier was run on the dataset without including class 3 and the confusion matrix for this is shown in table 7.2. Another reason for checking the results excluding class 3 is that the size of vocabulary of class 3 is very high (almost 4 times) as compared to the size of vobabulary of other classes as can be seen in table 2.4 and figure 2.1.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	228	0	0	0	0	0	0	0	1	0	1	2	0	6	3	47	7	8	8	8
2	0	311	0	11	12	18	1	0	0	0	0	21	3	2	6	0	0	1	2	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	10	0	308	26	7	7	1	2	0	1	4	25	0	1	0	0	0	0	0
5	0	12	0	22	307	3	5	5	0	0	0	4	11	2	11	0	1	0	2	0
6	0	41	0	10	3	323	4	0	0	0	1	3	0	1	5	0	2	1	1	0
7	0	4	0	48	19	7	250	20	6	1	2	1	13	4	6	1	3	0	4	1
8	0	1	0	0	0	0	3	371	3	0	0	3	5	0	1	1	2	2	4	0
9	0	1	0	0	0	0	1	9	374	2	0	1	3	0	1	0	1	0	5	0
10	1	0	0	1	0	1	1	2	1	357	21	2	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	1	1	0	1	1	0	1	2	0
12	0	2	0	0	2	1	1	0	0	0	0	376	3	1	0	0	5	1	4	0
13	1	19	0	24	6	1	5	14	2	0	0	42	261	9	6	0	0	2	1	0
14	8	7	0	0	1	0	0	2	2	0	0	0	3	341	5	6	6	5	10	0
15	2	7	0	0	1	0	0	0	0	0	0	1	3	3	363	0	1	2	11	0
16	10	2	0	0	0	0	0	0	0	0	0	0	0	2	2	370	2	1	2	7
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	0	338	1	11	3
18	5	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	3	342	14	0
19	4	0	0	0	1	0	0	0	0	1	0	4	0	1	8	1	102	4	184	0
20	42	3	0	0	0	0	0	0	0	0	0	3	0	2	6	60	22	4	6	103

Table 7.2: Confusion Matrix : Naive Bayes excluding Class 3 (Accuracy : 82.6%)

7.2.2 k-NN after Naive Bayes

Another method used combines the Naive Bayes and k-NN classifiers. The idea is to first classify the training data using the Naive Bayes classifier and record which document has been assigned to which class and actually belongs to which class. For classifying a new document doc_i, first classify doc_i using the Naive Bayes classifier and let the class assigned to it be C_j. Considering all the documents in training set that had been assigned class C_j, perform a k-NN on doc_i to make the final classification. The algorithm for this is algorithm 7.2.

Table 7.3 shows a dummy confusion matrix to explain this process.

	Class 1	**Class 2**	**Class 3**	**Class4**	**Class 5**	**Total**
Class 1	**73**	1	3	**21**	2	**100**
Class 2	3	**86**	7	1	5	**102**
Class 3	5	6	**78**	3	8	**100**
Class 4	**17**	2	4	**74**	3	**100**
Class 5	2	5	8	1	**82**	**98**
Total	**100**	**100**	**100**	**100**	**100**	**500**

Table 7.3: Dummy Confusion Matrix for Illustration

The confusion matrix shown in table 7.3 shows a dummy confusion matrix created on training set (containing 5 classes with 100 documents each) using Naive Bayes classifier. When a new test document doc_i has to be classified and it is classifies as class 1 using the Naive Bayes Classifier, we perform k-NN on document doc_i using all the documents corresponding to entries in column of class 1 in the confusion matrix. Same is the case for other classes.

The actual confusion matrix that we got by training and running the Naive Bayes classifier based on algorithm 7.1 is shown in table 7.4

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	456	0	0	0	0	0	0	1	0	0	0	0	0	0	1	15	0	4	2	1
2	0	554	0	16	1	7	1	0	0	0	0	1	0	1	1	2	0	0	0	0
3	0	69	93	172	20	181	8	4	0	0	1	26	9	2	2	1	1	0	2	0
4	0	9	0	547	7	5	5	1	0	0	1	4	6	1	2	1	0	0	1	0
5	0	4	0	5	549	3	1	1	0	0	0	4	5	2	0	0	1	0	3	0
6	1	10	0	4	1	566	0	0	0	0	0	3	0	1	4	0	0	1	2	0
7	0	1	0	33	7	3	480	17	3	1	3	11	10	4	3	0	2	2	5	0
8	0	0	0	0	0	2	2	575	0	0	0	3	2	2	1	0	6	0	1	0
9	0	1	0	0	0	0	3	3	582	0	0	1	0	0	2	0	3	0	3	0
10	0	1	0	0	0	1	0	0	0	587	4	0	0	1	0	0	3	0	0	0
11	0	0	0	0	0	2	0	0	0	0	592	0	1	2	0	0	0	0	3	0
12	0	0	0	0	0	0	0	0	0	0	0	590	0	0	0	0	2	1	2	0
13	0	2	0	14	4	0	3	3	0	1	1	14	542	1	3	1	0	1	1	0
14	0	1	0	0	0	0	0	0	0	0	0	0	1	589	1	0	2	0	0	0
15	0	1	0	0	0	0	0	0	0	0	0	1	0	2	589	0	0	0	0	0
16	2	0	0	0	0	2	0	0	0	0	0	0	0	0	3	586	1	3	2	0
17	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	539	2	3	0
18	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	559	1	0
19	1	0	0	0	0	0	0	0	0	1	0	6	0	0	1	0	5	3	448	0
20	24	0	0	0	0	0	0	0	0	0	0	0	0	2	1	44	20	6	7	273

Table 7.4: Confusion Matrix : Naive Bayes on Training Dataset

Algorithm 7.2: k-NN after Naive Bayes Classifier

Given a document, doc_i ;
for *Each document* doc_k *in training set* **do**
  Classify doc_k using Bayes Classifier ;
end
Classify doc_i using Naive Bayes Classifier and let that class be C_j ;
Set $newTrainSet$ as set of all documents in training set classifies as C_j ;
Perform k-NN on doc_i using $newTrainSet$;

The confusion matrix created by applying 1-NN (using cosine distance as distance measure) after Naive Bayes as given in algorithm 7.2 is shown in table 7.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	219	0	0	0	0	0	1	0	1	0	0	4	0	5	3	46	6	8	7	19
2	0	287	22	9	13	19	2	1	0	5	1	13	4	3	6	0	0	2	2	0
3	2	43	210	41	16	39	2	1	0	1	0	9	5	2	6	0	3	0	12	2
4	0	15	31	267	22	5	21	0	1	0	2	3	23	0	2	0	0	0	0	0
5	0	11	19	19	284	1	10	5	1	0	0	4	16	2	8	1	2	0	2	0
6	0	34	56	1	4	280	3	1	1	2	1	1	2	0	6	0	1	1	1	0
7	0	5	6	27	15	4	270	15	7	2	2	1	16	3	8	2	2	0	5	0
8	0	1	1	1	0	0	9	357	6	0	0	3	7	0	1	1	2	2	5	0
9	0	1	0	0	0	0	0	10	372	2	0	1	3	1	1	0	0	1	6	0
10	2	0	0	1	0	0	3	3	1	355	19	2	0	0	1	1	1	0	8	0
11	1	0	0	0	0	0	2	0	0	2	387	2	1	0	1	2	0	1	0	0
12	0	2	1	0	0	1	0	0	0	0	0	368	5	2	0	1	5	1	10	0
13	1	15	12	17	9	0	13	13	4	0	0	17	266	10	10	0	1	2	3	0
14	8	7	0	2	1	0	7	2	1	2	0	0	3	332	4	7	3	5	10	2
15	1	7	0	0	0	0	0	0	0	0	0	1	3	2	362	0	2	3	10	3
16	11	1	1	0	0	0	0	0	0	0	0	0	0	2	2	345	2	1	3	30
17	0	0	0	0	0	0	0	0	1	0	0	7	0	2	1	1	330	3	10	9
18	16	0	0	0	1	1	0	1	1	3	0	2	0	0	0	8	2	329	12	0
19	4	0	0	0	1	0	0	1	0	1	0	4	1	1	7	1	91	4	185	9
20	30	1	0	2	0	0	0	0	0	0	0	3	0	2	6	46	17	5	5	134

Table 7.5: Confusion Matrix : 1-NN (using Cosine Distance) After Naive Bayes (Accuracy : 78.85%)

The confusion matrix created by applying 3-NN (using cosine distance as distance measure) after Naive Bayes as given in algorithm 7.2 is shown in table 7.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	216	0	0	0	0	0	0	0	1	0	1	3	0	5	3	51	6	10	7	16
2	0	296	18	7	14	18	1	0	0	2	0	16	3	3	6	0	0	2	3	0
3	3	46	218	38	14	31	2	1	0	1	0	12	2	2	7	0	3	0	13	1
4	0	11	27	275	24	6	11	0	1	0	2	5	28	0	2	0	0	0	0	0
5	0	13	17	21	287	1	5	5	0	0	0	6	14	2	9	0	2	0	3	0
6	0	34	47	1	3	292	2	0	1	0	1	1	3	1	6	0	1	1	1	0
7	0	5	5	26	15	5	264	19	6	1	2	2	18	3	9	2	3	0	5	0
8	0	1	0	0	0	0	5	365	3	0	0	4	5	0	1	1	3	2	6	0
9	0	1	0	0	0	0	0	10	372	2	0	1	3	1	1	0	1	1	5	0
10	2	0	0	0	0	1	2	2	1	355	21	2	0	0	1	0	1	1	8	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	373	3	2	0	0	4	2	9	0
13	0	18	7	20	7	0	3	14	2	0	0	24	271	11	10	0	1	2	3	0
14	8	6	0	2	1	0	1	2	1	2	0	0	3	339	5	6	3	6	10	1
15	1	7	0	0	0	0	0	0	0	0	0	1	3	2	365	0	1	4	10	0
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	364	3	4	3	11
17	0	0	0	0	0	0	0	0	1	0	0	6	0	1	1	1	337	3	12	2
18	3	0	0	0	1	0	0	1	1	3	0	2	0	0	0	6	2	344	12	1
19	4	0	0	0	1	0	0	1	0	1	0	3	0	1	8	1	96	4	185	5
20	35	3	0	0	1	0	0	0	0	0	0	2	0	2	6	60	19	7	5	111

Table 7.6: Confusion Matrix : 3-NN (using Cosine Distance) After Naive Bayes (Accuracy : 79.9%)

The confusion matrix created by applying 5-NN (using cosine distance as distance measure) after Naive Bayes as given in algorithm 7.2 is shown in table 7.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	217	0	0	0	0	0	0	0	1	0	1	2	0	5	3	52	6	11	8	13
2	0	306	11	8	10	18	0	0	0	0	0	19	3	3	6	0	0	2	3	0
3	3	52	207	41	14	35	0	1	0	0	0	14	1	2	7	0	3	0	13	1
4	0	10	30	279	24	6	8	0	1	0	2	6	24	0	2	0	0	0	0	0
5	0	11	18	19	289	1	5	5	0	0	0	6	13	2	11	0	2	0	3	0
6	0	36	46	1	3	294	0	0	0	0	1	2	2	1	6	0	1	1	1	0
7	0	3	5	36	16	7	251	20	6	1	4	2	17	3	9	2	3	0	5	0
8	0	1	0	0	0	0	5	365	3	0	0	4	5	0	1	1	3	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	2	0	0	1	0	1	1	8	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	375	3	2	0	0	4	2	7	0
13	0	16	6	20	7	1	3	14	2	0	1	25	271	11	10	0	1	2	3	0
14	8	6	0	1	1	0	1	2	1	1	1	0	3	340	5	6	4	6	10	0
15	0	6	0	0	0	0	0	0	0	0	1	1	3	2	365	0	1	4	10	1
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	370	3	1	3	8
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	0	0	1	0	0	0	0	1	0	3	0	1	8	1	98	4	185	4
20	36	3	0	0	0	0	0	0	0	0	0	3	0	2	6	62	22	7	6	104

Table 7.7: Confusion Matrix : 5-NN (using Cosine Distance) After Naive Bayes (Accuracy : 79.89%)

The confusion matrix created by applying 10-NN (using cosine distance as distance measure) after Naive Bayes as given in algorithm 7.2 is shown in table 7.5.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	217	0	0	0	0	0	0	0	1	0	1	2	0	5	3	55	6	10	8	11
2	0	306	9	9	9	17	0	0	0	0	0	22	3	3	7	0	0	2	2	0
3	2	53	190	54	13	36	0	1	0	0	0	16	2	2	7	0	3	0	13	2
4	0	10	26	285	23	5	7	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	15	21	291	1	5	5	0	0	0	6	11	2	11	0	2	0	3	0
6	0	37	42	2	3	296	0	0	0	0	1	2	2	1	6	0	1	1	1	0
7	0	4	4	39	17	6	251	21	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	4	365	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	1	0	0	1	0	0	0	0	0	375	3	2	0	0	4	2	6	0
13	0	18	5	20	6	1	3	14	2	0	0	29	268	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	370	3	1	3	8
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	0	0	1	0	0	0	0	1	0	3	0	1	8	1	102	4	185	0
20	37	3	0	0	0	0	0	0	0	0	0	3	0	2	6	66	24	7	6	97

Table 7.8: Confusion Matrix : 10-NN (Using Cosine Distance) After Naive Bayes (Accuracy : 79.69%)

7.2.3 k-NN after Naive Bayes and removal of values less than 14 from each column of confusion matrix

One more method used is an extension to algorithm 7.2 in order to improve the accuracy given by that algoritm. In this method, documents belonging to class C_i in $newTrainSet$ are removed if the number of documents from class C_i in $newTrainSet$ are less than 14. For understanding, he confusion matrix shown in table 7.3 shows a dummy confusion matrix created on training set (containing 5 classes with 100 documents each) using Naive Bayes classifier. Only the bold values will be considered while performing the k-NN classification and rest is same as algorithm 7.2. We chose the value 14, as there were no 12 or 13 present in the confusion matrix created by Naive Bayes on training set geiven in table 7.4 and hence there was a jump from 11 to 14. The algorithm for this is algorithm 7.3.

Algorithm 7.3: k-NN after Naive Bayes Classifier with Removal of Classes whose documents appear less 14 times after applying Naive Bayes

```
Given a document, doc_i ;
for Each document doc_k in training set do
    Classify doc_k using Bayes Classifier ;
end
Classify doc_i using Naive Bayes Classifier and let that class be C_j ;
Set newTrainSet as set of all documents in training set classifies as C_j ;
for Each class C_i do
    if Number of documents of C_i in newTrainSet ¡ 14 then
        Remove documents belonging to class C_i from newTrainSet ;
    end
end
Perform k-NN on doc_i using newTrainSet ;
```

The confusion matrix created by applying 1-NN (using cosine distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	219	0	0	0	0	0	0	0	1	0	1	2	0	5	3	45	6	10	8	19
2	0	294	26	6	9	16	1	0	0	0	0	19	4	3	7	0	0	2	2	0
3	2	43	211	41	10	38	2	1	0	0	0	13	6	2	7	0	3	0	13	2
4	0	16	31	256	23	5	23	0	1	0	2	6	27	0	2	0	0	0	0	0
5	0	11	21	17	289	1	6	5	0	0	0	6	12	2	11	0	2	0	2	0
6	0	35	61	2	3	277	2	0	0	0	1	1	3	1	6	0	1	1	1	0
7	0	3	6	22	17	5	270	15	6	1	2	2	17	5	9	2	3	0	5	0
8	0	1	0	0	0	0	10	359	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	1	1	0	1	1	2	1	355	21	2	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	3	0	0	1	0	0	0	0	0	373	5	2	0	0	4	2	4	0
13	0	16	13	16	6	0	3	14	2	0	0	25	271	12	10	0	1	2	2	0
14	8	5	1	1	1	0	0	2	1	1	0	0	3	342	5	6	3	6	10	1
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	11	1	1	0	0	0	0	0	0	0	0	0	0	2	2	344	3	1	3	30
17	0	0	0	0	0	0	0	0	1	0	0	8	1	0	1	1	330	3	11	8
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	1	0	1	0	0	0	0	1	0	3	0	1	8	1	94	4	184	8
20	30	3	0	0	0	0	0	0	0	0	0	3	0	2	6	46	17	7	6	131

Table 7.9: Confusion Matrix : 1-NN (Using Cosine Distance) with Removal of Classes whose documents appear less 14 times after applying Naive Bayes (Accuracy : 79.35%)

The confusion matrix created by applying 1-NN (using Jaccard distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	215	0	0	0	0	0	0	0	1	0	1	2	0	5	3	41	5	10	8	28
2	0	296	25	4	9	17	2	0	0	0	0	17	5	3	7	0	0	2	2	0
3	3	42	232	34	10	26	0	1	0	0	0	15	5	2	7	0	3	0	13	1
4	0	15	38	251	22	5	23	0	1	0	2	7	26	0	2	0	0	0	0	0
5	0	14	12	15	299	1	5	5	0	0	0	6	11	2	11	0	1	0	2	1
6	0	33	56	3	3	283	2	0	0	0	1	1	3	1	6	0	1	1	1	0
7	0	4	5	15	17	6	283	11	6	1	2	2	14	5	9	2	3	0	5	0
8	0	1	0	0	0	0	13	356	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	1	0	0	1	0	0	0	0	0	375	5	2	0	0	4	2	4	0
13	0	18	11	16	6	1	3	14	2	0	0	23	272	12	10	0	1	2	2	0
14	7	3	3	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	1
15	1	5	1	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	6	2	0	0	0	0	0	0	0	0	0	0	0	2	2	357	3	1	3	22
17	0	0	0	0	0	0	0	0	1	0	0	8	1	0	1	1	324	3	11	14
18	2	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	1
19	4	0	0	0	1	0	0	0	0	1	0	4	0	1	8	1	93	4	184	9
20	25	3	1	0	0	0	0	0	0	0	0	2	0	2	6	46	13	7	6	140

Table 7.10: Confusion Matrix : 1-NN (Using Jaccard Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 80.14%)

The confusion matrix created by applying 3-NN (using cosine distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	216	0	0	0	0	0	0	0	1	0	1	2	0	5	3	51	6	10	8	16
2	0	302	17	7	9	17	0	0	0	0	0	20	3	3	7	0	0	2	2	0
3	3	48	215	39	9	31	2	1	0	0	0	17	3	2	7	0	3	0	13	1
4	0	13	26	274	23	6	11	0	1	0	2	7	27	0	2	0	0	0	0	0
5	0	12	19	19	289	1	5	5	0	0	0	6	12	2	11	0	2	0	2	0
6	0	35	49	1	3	290	2	0	0	0	1	1	3	1	6	0	1	1	1	0
7	0	3	4	25	17	7	263	19	6	1	2	2	17	5	9	2	3	0	5	0
8	0	1	0	0	0	0	6	363	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	0	0	1	2	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	3	0	0	1	0	0	0	0	0	375	3	2	0	0	4	2	4	0
13	0	18	7	20	6	0	3	14	2	0	0	26	270	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	3	6	10	1
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	367	3	1	3	11
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	335	3	11	3
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	1	0	1	0	0	0	0	1	0	3	0	1	8	1	97	4	184	5
20	35	3	0	0	0	0	0	0	0	0	0	3	0	2	6	60	19	7	6	110

Table 7.11: Confusion Matrix : 3-NN (Using Cosine Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 79.95%)

The confusion matrix created by applying 3-NN (using Jaccard distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	215	0	0	0	0	0	0	0	1	0	1	2	0	5	3	51	5	10	8	18
2	0	304	14	8	9	16	0	0	0	0	0	20	4	3	7	0	0	2	2	0
3	3	48	225	40	10	22	0	1	0	0	0	15	4	2	7	0	3	0	13	1
4	0	14	30	271	23	5	13	0	1	0	2	6	25	0	2	0	0	0	0	0
5	0	13	9	20	298	0	5	5	0	0	0	7	11	2	11	0	2	0	2	0
6	0	35	51	2	3	287	2	0	0	0	1	2	2	1	6	0	1	1	1	0
7	0	3	6	21	17	5	278	12	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	9	360	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	378	3	2	0	0	4	2	4	0
13	0	18	10	18	6	1	3	14	2	0	0	23	271	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	2	0	0	0	0	0	0	0	0	0	0	0	2	2	366	3	1	3	12
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	334	3	11	4
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	1	0	1	0	0	0	0	1	0	3	0	1	8	1	98	4	184	4
20	32	3	0	0	0	0	0	0	0	0	0	3	0	2	6	55	18	7	6	119

Table 7.12: Confusion Matrix : 3-NN (Using Jaccard Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 80.44%)

The confusion matrix created by applying 5-NN (using cosine distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	217	0	0	0	0	0	0	0	1	0	1	2	0	5	3	53	6	10	8	13
2	0	306	10	8	9	17	0	0	0	0	0	22	3	3	7	0	0	2	2	0
3	3	54	202	42	11	36	0	1	0	0	0	16	3	2	7	0	3	0	13	1
4	0	11	30	278	23	6	8	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	17	20	291	1	5	5	0	0	0	6	11	2	11	0	2	0	2	0
6	0	36	42	2	3	295	2	0	0	0	1	2	2	1	6	0	1	1	1	0
7	0	4	3	35	17	7	253	20	6	1	2	2	16	5	9	2	3	0	5	0
8	0	1	0	0	0	0	6	363	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	2	0	0	1	0	0	0	0	0	376	3	2	0	0	4	2	4	0
13	0	18	5	20	6	1	3	14	2	0	0	27	270	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	0	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	1
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	369	3	1	3	9
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	0	0	1	0	0	0	0	1	0	4	0	1	8	1	98	4	184	4
20	36	3	0	0	0	0	0	0	0	0	0	3	0	2	6	62	22	7	6	104

Table 7.13: Confusion Matrix : Confusion Matrix : 5-NN (Using Cosine Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 79.85%)

The confusion matrix created by applying 5-NN (using Jaccard distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	216	0	0	0	0	0	0	0	1	0	1	2	0	5	3	54	6	10	8	13
2	0	310	7	8	9	17	0	0	0	0	0	20	4	3	7	0	0	2	2	0
3	3	54	212	43	12	24	0	1	0	0	0	17	2	2	7	0	3	0	13	1
4	0	11	31	275	23	5	11	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	11	20	296	1	5	5	0	0	0	7	11	2	11	0	2	0	2	0
6	0	37	46	2	3	290	2	0	0	0	1	2	2	1	6	0	1	1	1	0
7	0	3	5	31	17	5	268	13	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	6	363	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	378	3	2	0	0	4	2	4	0
13	0	18	9	19	6	0	3	14	2	0	0	26	269	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	2	0	0	0	0	0	0	0	0	0	0	0	2	2	369	3	1	3	9
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	1	0	1	0	0	0	0	1	0	3	0	1	8	1	99	4	184	3
20	33	3	0	0	0	0	0	0	0	0	0	3	0	2	6	61	20	7	6	110

Table 7.14: Confusion Matrix : 5-NN (Using Jaccard Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 80.27%)

The confusion matrix created by applying 10-NN (using cosine distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	217	0	0	0	0	0	0	0	1	0	1	2	0	5	3	55	6	10	8	11
2	0	307	8	9	9	17	0	0	0	0	0	22	3	3	7	0	0	2	2	0
3	2	54	191	54	13	33	0	1	0	0	0	17	2	2	7	0	3	0	13	2
4	0	11	25	285	23	5	7	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	14	21	292	1	5	5	0	0	0	7	11	2	11	0	2	0	2	0
6	0	37	38	3	3	297	2	0	0	0	1	3	1	1	6	0	1	1	1	0
7	0	4	4	39	17	6	251	21	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	4	365	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	378	3	2	0	0	4	2	4	0
13	0	18	4	20	6	1	3	14	2	0	0	30	268	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	1	1	0	0	0	0	0	0	0	0	0	0	2	2	370	3	1	3	8
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	1	0	1	0	0	0	0	1	0	3	0	1	8	1	102	4	184	0
20	37	3	0	0	0	0	0	0	0	0	0	3	0	2	6	66	24	7	6	97

Table 7.15: Confusion Matrix : 10-NN (Using Cosine Distance) After Naive Bayes and Removal of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 79.77%)

The confusion matrix created by applying 10-NN (using Jaccard distance as distance measure) after Naive Bayes and removing values less than 14 as given in algorithm 7.3 is shown in table 7.9.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	218	0	0	0	0	0	0	0	1	0	1	2	0	5	3	55	6	10	8	10
2	0	310	6	8	9	17	0	0	0	0	0	22	3	3	7	0	0	2	2	0
3	3	53	215	42	14	21	0	1	0	0	0	17	2	2	7	0	3	0	13	1
4	0	11	26	284	22	5	8	0	1	0	2	7	24	0	2	0	0	0	0	0
5	0	12	7	21	300	0	5	5	0	0	0	7	11	2	11	0	2	0	2	0
6	0	37	36	5	3	297	2	0	0	0	1	3	1	1	6	0	1	1	1	0
7	0	4	3	38	17	5	260	15	6	1	2	2	13	5	9	2	3	0	5	0
8	0	1	0	0	0	0	6	363	3	0	0	4	5	0	1	1	4	2	6	0
9	0	1	0	0	0	0	0	10	371	2	0	1	3	1	1	0	2	1	5	0
10	2	0	0	1	0	1	1	2	1	355	21	3	0	0	1	0	1	1	7	0
11	1	0	0	0	0	0	1	0	0	1	389	2	1	0	1	2	0	1	0	0
12	0	2	0	0	0	1	0	0	0	0	0	378	3	2	0	0	4	2	4	0
13	0	18	5	20	6	0	3	14	2	0	0	33	265	12	10	0	1	2	2	0
14	8	6	0	1	1	0	0	2	1	1	0	0	3	342	5	6	4	6	10	0
15	1	6	0	0	0	0	0	0	0	0	0	1	3	3	365	0	1	4	10	0
16	7	2	0	0	0	0	0	0	0	0	0	0	0	2	2	369	3	1	3	9
17	0	0	0	0	0	0	0	0	1	0	0	9	0	0	1	1	337	3	11	1
18	3	0	0	0	0	0	0	1	1	3	0	3	0	0	0	4	2	347	12	0
19	4	0	0	0	1	0	0	0	0	1	0	4	0	1	8	1	102	4	184	0
20	34	3	0	0	0	0	0	0	0	0	0	3	0	2	6	67	20	7	6	103

Table 7.16: Confusion Matrix : 10-NN (Using Jaccard Distance) After Naive Bayes with RRemoval of Classes whose documents appear less 14 times in confusion matrix column after applying Naive Bayes on training set (Accuracy : 80.35%)

Chapter 8

Summary, Conclusion and Future Work

8.1 Summary

The results (accuracies) of different methods using k-NN and k_2-NN with k_1-Means applied on the 20 newsgroup dataset can be summarised by table 8.1.

Method	Using Cosine Distance	Using Jaccard Distance
1-NN	68.27	72.34
3-NN	68.79	73.45
5-NN	69.38	74.75
10-NN	70.47	75.21
20-NN	69.92	75.33
1-NN After k-Means	56.54	-
3-NN After k-Means	55.66	-
1-NN After k-Means and removal of 10% documents from each cluster	62.09	66.17
3-NN After k-Means and removal of 10% documents from each cluster	61.27	65.48
5-NN After k-Means and removal of 10% documents from each cluster	60.46	-
10-NN After k-Means and removal of 10% documents from each cluster	59.25	-

Table 8.1: Accuracies obtained by different methods on 20 ewsgroup dataset (1/2)

The results (accuracies) of different methods using Naive Bayes and k-NN after Naive Bayes applied on the 20 newsgroup dataset can be summarised by table 8.2. Since Naive Bayes does not use any distance measure, the accuracy of Naive Bayes is mentioned separately here and its is 77.79% and 82.6% without including class 3.

Method	**Cosine Distance**	**Jaccard Distance**	**Cosine Distance Exluding Class 3**	**Jaccard Distance Excluding Class 3**
1-NN After Naive Bayes	78.85	-	82.45	-
3-NN After Naive Bayes	79.9	-	82.8	-
5-NN After Naive Bayes	79.89	-	82.88	-
10-NN After Naive Bayes	79.69	-	82.71	-
1-NN After Naive Bayes and Removal of values less than 14	79.35	80.14	82.73	82.87
3-NN After Naive Bayes and Removal of values less than 14	79.95	80.44	82.74	83.02
5-NN After Naive Bayes and Removal of values less than 14	79.85	80.27	82.7	82.88
10-NN After Naive Bayes and Removal of values less than 14	79.77	80.35	82.64	82.87

Table 8.2: Accuracies obtained by different methods on 20 newsgroup dataset (2/2)

8.2 Conclusion

We tried to form a hierarchical classifier for classifying the 20 newsgroup dataset. We tried different methods and all methods gave better results when a hierarchical structure was followed. The results improved further when outliers were found and removed. k_2-NN after k_1-Means method gave better accuracy when 10% of documents were removed from each cluster. k-NN after Naive Bayes Classification gave better results when the documents corresponding to the values less than 14 in confusion matrix created on training dataset were removed. As can be seen from table 8.2, there is an improvement of 2.65% (200 documents) that is from 77.79% (5859 documents) in Naive Bayes to 80.44% (6059 documents) in 3-NN after Naive Bayes and removal of values less than 14. Another finding done is that Jaccard Distance gave better accuracy than the cosine distance for classification of 20 newsgroup dataset.

8.3 Future Work

The following can be further implemented to check whether the accuracy can still be improved:

- The methods used in the dissertation can be applied on documents using tf-idf (term frequencyinverse document frequency) weighing scheme.
- The outliers of each class can be identified after applying hierarchical agglomerative clustering on each class and one can come by with a method to classify a new document after removal of these outliers.
- Support vector machines (SVM) can also be used to perform hierarchical clustering in similar ways to those in this dissertation.
- The methods that give the best performance can be run on different datasets to find out which methods actually improve the accuracy of classification in general.

Bibliography

[1] Brücher, H., Knolmayer, G., Mittermayer, M.A.: Document classification methods for organizing explicit knowledge. Institut für Wirtschaftsinformatik der Universität Bern (2002), http://www2.warwick.ac.uk/fac/soc/wbs/conf/olkc/archive/oklc3/papers/id237.pdf

[2] Cachopo, A.M.d.J.C.: Improving methods for single-label text categorization. Ph.D. thesis, Universidade Técnica de Lisboa (2007), http://web.ist.utl.pt/acardoso/docs/2007-phd-thesis.pdf

[3] Cunningham, P., Delany, S.J.: k-nearest neighbour classifiers. Multiple Classifier Systems pp. 1–17 (2007), http://csiweb.ucd.ie/UserFiles/publications/UCD-CSI-2007-4.pdf

[4] Huang, A.: Similarity measures for text document clustering. In: Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand. pp. 49–56 (2008), http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.332.4480&rep=rep1&type=pdf

[5] Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. Cambridge University Press (2008), http://nlp.stanford.edu/IR-book/html/htmledition/hierarchical-agglomerative-clustering-1.html

[6] Matteucci, M.: A tutorial on clustering algorithms, http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html

[7] Rennie, J.: 20 newsgroup data set, http://qwone.com/~jason/20Newsgroups/