TEXT CLASSIFICATION USING MACHINE LEARNING ALGORITHMS

SUBMISSION DATE: AUGUST 2, 2018

SUBMITTED BY:

Fahim Hasnat (14101043)
Md. Mazidul Hasan (14301104)
Nayeem Hasan Khan (14301113)
Asif Ali (12201068)
Department of Computer Science and Engineering

Supervisor:

Amitabha Chakrabarty, Ph.D

Assistant Professor

Department of Computer Science and Engineering

Declaration

We, hereby declare that this thesis is based on results we have found ourselves. Materials of work from researchers conducted by others are mentioned in references.

Signature of Supervisor

Amitabha Chakrabarty, Ph.D.

Assistant Professor

Department of Computer Science and Engineering

BRAC University

Signature of Authors

Fahim Hasnat (14101043)

Md. Mazidul Hasan (14301104)

Nayeem Hasan Khan (14301113)

Asif Ali (12201068)

ABSTRACT

Financial, educational and communal activities produce enormous amount of data. Automatic text classification has significant application in content organization, point of view extraction, evaluation analysis, spam filtering and sentiment analysis. Automatic classification of text documents requires information extraction, machine learning and Natural Language processing. We have proposed a probabilistic framework for text classification. Proposed classification model is composed of three major modules i.e. pre-processing of unstructured text, learning of probabilistic model and the classification of unseen data by using learned model. This framework is trained and tested by using "20 newsgroup" dataset containing twenty different news categories i.e. politics, sports, religions and pc hardware. We have used both supervised and unsupervised algorithms to get the full insight on the relationships among various text classification techniques. Highest accuracy of 84.51% was achieved for 4 categories by Naïve Bayes among the other Supervised Algorithms we used and 62.79% homogeneity was achieved for unsupervised algorithms for 4 categories which demonstrates the effectiveness score of proposed automatic text classification approach.

Keywords: *Text Classification, Machine learning, Pre-processing, Feature Extraction, Naïve Bayes, SVM, KNN, Decision Tree*.

Acknowledgement

At first, we would like to express our deepest gratitude to Almighty Allah for keeping us safe and sound for putting our best effort and successfully complete research work. Secondly, we would like to thank and show our respect to our honorable Supervisor Dr. Amitabha Chakrabarty, for his experience, assistance, guidance and patience in conducting research work and preparing this report. We are very thankful and humbled to have him as our supervisor. His tireless support and supervision ensure our progress towards the completion of this thesis work. We thank our parents and our beloved friends for the support, moral help and aids. They help us a lot with their direct or indirect participation by giving us suggestion which give us courage to face all kinds of difficulties. Our profound gratitude to Tasnia Ashrafi Heya, research assistant of Department of Computer Science and Engineering, BRAC University for her excellent support and guidance regarding the research and preparation of this report. Last but not the least, we would like to thank specially our very own BRAC University for providing us the opportunity to support and help us to conduct this research work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
Introduction

This chapter gives a general review of the purpose of this research work and an overview of the entire research. It also includes the basic idea of Machine learning (ML) data preprocessing and feature extraction. Machine learning algorithms allow the software applications to become more accurate in predicting outcomes without being explicitly programmed [1]. It focuses on the development of computer programs that can access the data and use it to learn by themselves. The basic premise of machine learning is to build algorithms that can receive raw input data, use statistical analysis to predict an output while updating itself and fetching new data for applying the learning. The process involved in machine learning are similar to the data mining and predictive modeling. Both require searching through data to look for patterns and adjusting program action accordingly [2].

Nowadays, with the development of growing internet technology, more and more network information has been presented digitally, the number of text on the database have dramatically increased and is becoming scattered. The biggest problem of text classification is high dimension of feature space and text representation of sparse vectors [3]. In the past, people used the KNN algorithm, Naive Bayesian algorithm, Decision Tree algorithm, Rough Set algorithm and Support Vector Machine to classify text, but these methods were realized by single neural network, and did not understand really the importance of feature extraction [3]. Feature extraction is the process where the key features are extracted that help to classify the text. In recent years, researchers have begun to focus on the research that can reduce the dimensionality of text classification as well as improve text feature extraction algorithm. The two major considerations are how accurately the data is classified and overall system efficiency. Accuracy ensures that text is categorized in correct class i.e. sports data is classified in sports, fashion data in fashion and so on and the other factor is efficiency which ensures overall classification and model's performance. The news classification has been in talks for last few years. Research on document level or large text classifications is now more popular compared to classifying short texts. The reason behind this is that the noisy data for short news classification, which does not actually support entire process resulting in considerable less inaccuracy as well as efficiency. Another fact is the learning process is short in large

text classification. Lots of researches have been carried out by the researchers on text classifications including news group classification producing sufficient outputs. There had been numerous findings in the past in the same area including emotions classifications, synonyms classifications, twitter news classifications, and so on. In all of these text classifications, it is proved that data is limited for classifying news texts performing metrics i.e. time, accuracy, efficiency, and many other related factors with great and improved results [6]. News classifications of short texts, which have taken place in past include classifying emotions on basis of short news group, classifying short texts, classifying financial news, automated news classification, classifying news using N-gram and classifying news of social media i.e. Twitter are helping to grow the area of classifying the text. In addition to this, classifying full text news or document level news categorization is a vast area as compared to headlines classification or short text classification. The paper basically focuses at news type classification introducing a probabilistic approach for classifying news type where each of the news type is categorized into its defined class respectively i.e. system will read the description and it will categorizes it into suitable category i.e. sports, entertainment, fashion, and others. The classes are self-defined in the training data set and the two data sets have been prepared explicitly for this purpose including training and test dataset.

1.1 Motivation

Machine learning evolves from artificial intelligence and study of pattern recognition. Digitization has changed the way we process and analyze information. Now a days, a lots of data are being digitized every day. Modern machine learning now have the accuracy to categorize any kind of information and deliver us the content we need. Daily Newspaper is such a sector where a lot of data are updated daily, and if anyone want to find any specific information about a topic he/she has to go through every paper which is almost not possible. Here comes the need of text classification, where we can generate any kind of class of our interest and get the data easily. The text classification has been adopted by a growing number of organization to effectively manage the ever growing inflow of unstructured information. The core purpose of the text classification is to increase the discoverability of information and to serve those kinds of knowledge which will help us to make strategic decision. Text classification is mostly needed when the task involves business-specific context that are very complex which is done by

manual text classification. Manual text classification takes a lot of time (depends on the dataset) but it is highly detailed and accurate.

1.2 Objective

The main objectives of this research are following:

- Categorize all text of news paper
- Search faster and smarter

1.3 Methodology

In this research, we build a model to classify the "20 Newsgroup" dataset using Unsupervised and Supervised approach. Firstly, the loaded dataset will be preprocessed. Then we applied feature extraction process where we extract those key words that we were going to use for prediction of the selected topic. After that we used machine learning algorithms in different parameters to bring out the best result in a given parameter. In supervised process the data set has labeled information which include numeric value and categorical value. In unsupervised learning process the test data will not have any classification or labeled information. In this learning system our prediction will be based on clustering. Clustering finds pattern and groups the unlabeled data.

1.4 Thesis Overview

Chapter 1 is the introduction of our research work. We also have mentioned our motivation and objective to do this research as well as short overview of the methodology we followed.

Chapter 2 consists of the literature review where we have demonstrated the background study that we have done for this thesis

Chapter 3 consists of workflow, dataset cleaning algorithms and feature extraction.

Chapter 4 is the section where we have presented algorithms and implementation of the overall research.

Chapter 5 contains the conclusion and future possibilities and planning of this research.

CHAPTER 2
Literature Review

This chapter contains the background study required for this research work. Our findings include understanding about Machine Learning (ML) Algorithms, Pre-processing, feature extraction, Text selection and text segmentation. This chapter also refers to related works and their accuracy. Machine Learning is paradigm that may refer to leaning form past experience (Which in this case previous data) to improve future performance [17]. Its main focus is to learn automatically and also modify and improve of algorithms based on past experience without any external assistance from human. Text classification is the process of defining a label for a previous unseen documents [19]. It is an active and important area where machine learning algorithms are used frequently.

Automated text classification has always been an important research sector since the implementation of digital document [21]. The necessity of text classification is very high because we have to deal with large amount of text which are being digitized every day.

2.1 Machine Learning

We will discuss about the rear history required for our research paper in this section. Our work covers understanding of Machine Learning Algorithm, Data Pre-processing, feature selection, feature extraction and Text classification methods. Machine Learning is a group of algorithm that enables software application to generate probably the highest accurate in calculating outcomes without being explicitly programmed [1]. The common proposition of machine learning is to create or develop algorithms that receive input data and make statistical analysis for predicting an output. Machine Learning programs are also built to learn and improve over time whenever new data become available. We use machine learning techniques for automated text classification. It has many application such as self-driving cars, spam filtering, web search, graphic recognition system, identification of document genre, authorship attribution, automated essay grading, classification of news articles etc. [2][4].

2.1.1 Machine Learning Algorithms

The imminent need to access the rising availability of documents in digital form, the content-based document management tasks have earned a remarkable status in the

information systems area. Text classification also known as text categorization is the job of allocating predefined classes to free-text documents. Machine Learning is an inductive process which builds automated text classifier by learning from a set of pre-classified documents. This process has an advantage over knowledge engineers or domain experts for the structure of the classifier as it has highest accuracy comparable to that gained by human experts [3]. So, machine learning techniques have the advantage for better understanding in every aspects from theoretical point of view and performance surety in parameter guidance.

2.1.2 Supervised Learning Algorithms

In this paper we have discussed three categories supervised learning, unsupervised learning and semi-supervised learning for approaching text classification techniques [4]. Under supervised learning a program is trained on pre-defined dataset. It analyzes training data and gives a suppositional function and the function will be used to calculate output from a given input. Those who do not give desired output will be calculated from a given new data. For example to train a spam classifier algorithm, a training dataset of human tagged spam or non-spam emails is used and then uses the algorithm to classify whether a new email without tag is spam or not [2]. For text classification each document is leveled by zero or more categories and new text will be classified by learning a classifier. Documents which are labeled will be considered as positive and others as a negative example. The task is to find a weight vector for a text classifier which classifies new text documents. K-Nearest Neighbor classifier, Naïve Bayes, Decision Trees, Support Vector Machines, and Decision Rules Classification includes in supervised learning algorithms [4]. We will take a look of these classifier algorithms in this paper.

K-Nearest Neighbor

K-NN is an excellent supervised learning algorithm which stores all the cases and based on similarity measurement it classifies new cases. It works on an assumption that documents which are closed in the space belong to the same class [5]. By using some similarity measure such as Euclidean distance measure etc., the weight of the categories of the neighbor document can be found for a fix amount of the nearest neighbors for all training samples.

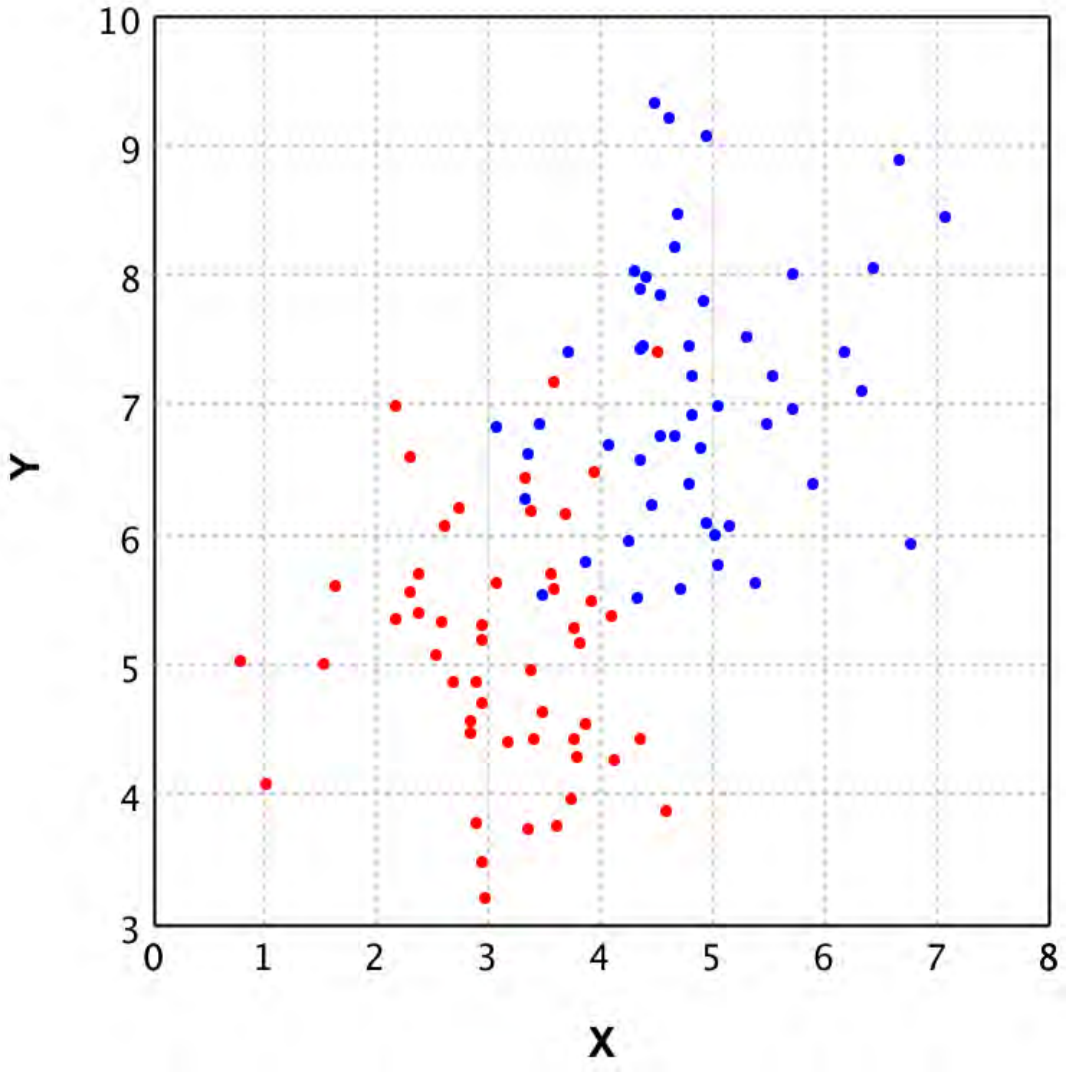

Figure 2.1.2 K Nearest Neighbor. [7]

$$Dis(x,y) = \sqrt{\sum_{i=1}^{d}(xi - yi)}\text{^}2 \quad \text{..... (i)}$$

This method is effective and easy to implement but it becomes slow when it uses all features in computing distance as the irrelevant features declines its accuracy.

Naïve Bayes

Naïve Bayes s one of the most basic text classification techniques with different applications in email spam detection, document categorization, documents categorization, language detection etc. It is a probabilistic method based on Bayes Theorem that makes independence assumptions. By using Bayes Theorem it predicts the following probabilities by reading a set of examples in attribute value representation. The feature order and presence of one feature does not affect each other because of its

independent assumption in classification task [4]. The only drawbacks of this method is, when dependencies arises it does not perform and give low performance.

Support Vector Machine

Support Vector Machine is one of the most powerful tools for text classification. The algorithm addresses the common problem of learning to differentiate the positive and negative attributes of a given class which is achieved by a linear or non-linear separating surface in the input space of the dataset. It is based on the Structural Risk Minimization principle [9]. The ability to learn independently of the feature space makes SVM really suitable for text classification. The major advantage of SVM is it remains unchanged when documents that do not belong to the support vectors are removed from the training dataset but this is relatively more complex than other algorithm where it also requires high time and memory consumptions [4].

Decision Tree

When decision tree is used for text classification, the internal nodes are considered as tests and leaf nodes as categories. As long as it reaches a certain leaf which represents the goal for the classification, the tree is exploring the structure from the root node [5]. The every internal node and branch from a node test one attribute and one value respectively. The goal attribute is not defined as a result we use the attribute which gives the maximum information and the leaf node predicts the class or category for the classification.

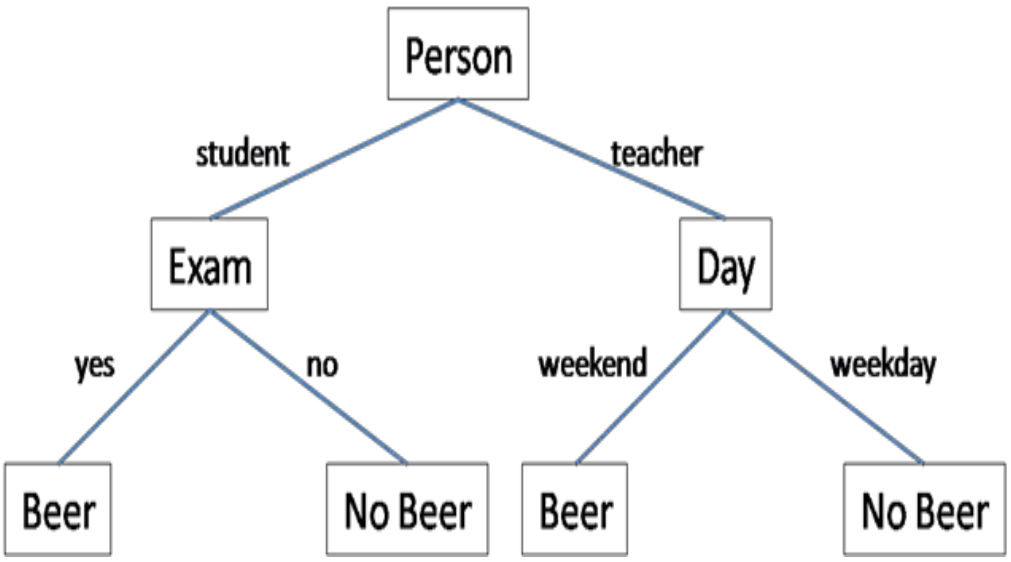

Figure 2.1.2 Decision Tree [8]

The above example shows the attribute person is the root node and the instances are spilt based on their values it take for the root node. The instance which gives the maximum information for the root attribute person is chosen as the decision and so we can say selecting person gives the highest information at that level. The edges represents values and the instances are divided according to each child nodes [6]. Though this approach has been successful over the past but due to swapping of training tuples it becomes ineffective as many training data do not fit in memory decision tree construction [5].

Table 2.1.2.1 Pros and Cons of different text classification algorithms

Classification Algorithm	Pros	cons
KNN	• Effective • Non-parametric • Easy to implement	• Time consumption • Not ideal based on accuracy
Naïve Bayes	• Easy to implement compared to other • Work well on small amount of data	• Perform poorly when dependencies arises
SVM	• Large spaces of feature • Work well on multi-label classification • Work for linear and non-linear data	• Not perform well for highly co-related data
Decision Tree	• Easy to understand • Easy to generate	• Training time is costly • Need human experts to construct or update rules

2.1.3 Unsupervised Learning Algorithms

The Unsupervised learning is a method that is used to classify unsorted information even there is no categories provided. Basically, it means to expose the machine to large volume of varied data and allow it to learn and infer from data. Unsupervised learning algorithms can perform more difficult procedures than supervised learning. Hierarchical Clustering Techniques, Partitional Clustering

Techniques, Kohonen's Self organizing Network etc. some clustering methods used for unsupervised learning.

Hierarchical Clustering

In this process, the two most similar clusters are joined and continue to put together until all the objects are in the same cluster. Hierarchical clustering algorithms make a cluster hierarchical in the form of tree structure called dendrogram [4]. We can pick any number form this tree in terms of output. Starting with N cluster where one for each data point, we have N-1 clusters when we merge the two clusters that are closest to each other. Then we calculate the distances between clusters by considering the distance between two clusters and the average distance between all their respective members. This process will be repeated until we get one cluster of N data points (dendrogram). Generally, the number of clusters is equal to the number of intersection points of the horizontal line with the vertical lines in the dendrogram [10]. This process have two subtypes: Divisive Hierarchical and Agglomerative Hierarchical Clustering. Divisive hierarchical is used by grouping all the information into one cluster and successively splitting these clusters. Besides, agglomerative works by sequentially merging similar clusters.

Partitional Clustering Techniques

In partitional clustering, each object belongs in exactly one cluster. Each cluster can be represented by a cluster representative. Here, an initial partition is constructed belong to n objects and partition these objects into k clusters and then clustering solution is reformed iteratively by moving documents from one cluster to another [4]. This method can be implemented by K-means clustering where we cluster our data into k groups and assign each data point to one of the nearest centroid's cluster. The distance of nearness is calculated often by Euclidean distance formula. After that we move the centroids to the center of their clusters [10]. Iteration will be continued until a stopping criterion is achieved. There is another process called Bisecting K-means which achieves a hierarchy of clusters by repeatedly applying K-means algorithm. A cluster is selected and the selected cluster is split into two for k=2 where the cluster with less similarity or that containing the maximum number of documents can be chosen to split.

Kohonen's Self Organizing Network

Kohonen's self-organizing network is a special type of neural network. In this process, an input vector represents to the network and the output is compared with the target vector. After that it checks whether they are differ and if so then the weights of the network are slightly switched to reduce the error in the output. Until the network gives the desired output the process will be repeated many times and with many sets of vector pairs [11]. These are used in the application for visualization aid. Though Kohonen's SOM are more complex and outstretched but able to make it easy for humans to explore relationships between huge amount of data [4].

2.1.4 Semi-supervised Learning Algorithms

Semi-supervised learning is in between of supervised and unsupervised learning. It is a class of supervised learning. Unlabeled data and labeled data are used by these algorithms to classify new unlabeled text document. The earliest idea of using unlabeled data was self-learning or self-training [13].Semi-supervised algorithm uses small amount of labeled data and a large amount of unlabeled data typically. They use labeled data to help to identify the specific group of web page types. Then the algorithm is trained on unlabeled data to define the boundaries of those web page types [12].

Co Training

This machine learning algorithm used when there are only small amounts of labeled data and large amounts of unlabeled data. This algorithm trains two classifiers separately on two sufficient and unnecessary views. As an example, two attributes which are sufficient for learning and independent to the other given class label uses the prediction of each classifier on unlabeled examples of the other.

An example of co-training algorithm can be COREG algorithm. It uses two k-nearest neighbor regressions with different distance metrics. Coreg can be effective to exploit unlabeled data to improve regression estimates [14].

EM Based

It is Expectation- Maximization algorithm. By estimating of a generative model through iterative Expectation- Maximization, these algorithms are used to train classifiers. Based on multinomial, text documents are represented with a bag of words model [4]. The theoretical basis for Expectation- Maximization shows that with sufficiently large amounts of unlabeled data generated by the model will lead to more

probable model. This approach rests on the assumption that the generative model is correct [15].This model is much simplistic to represent. But it has a problem of local maxima.

Graph Based

It is the theory that study graph models. Graphs are the mathematical structures used to describe the pairwise relations between objects [16].Graph based methods are generally non-parametric. It works on a closed data set and at the time of training, test set is relevant. It is assumed that the data is embedded within a low-dimensional numerous by a graph. Within a weighted graph, each data sample is represented by vertex [4].

2.2 Text Categorization

Text categorization is the process of assigning predefined categories of previously unseen documents [24]. Main purpose of text categorization is to include processing features, extracting irrelevant features against the feature in database and categorize a set of documents into predefined categories [25]. The categorization of texts helps to understand and creates opportunities to make more specific category for that raw data. The purpose is to run faster and efficiently. Text categorization helps Machine learning algorithms to run smoother and in more controlled way possible.

2.2.1 Pre-Processing

Data preprocessing is the process where processing algorithms are applied on raw data for future processing procedure. Data preprocessing transforms the data into a format where it will be more effectively and easily processed for the later ML algorithms. Main goal is to represent each data as a feature vector and separate the raw text individually [25]. It is important to select the proper words that will hold the proper value and meaning which will have some weight after the applied of the algorithms. There are a lot of preprocessing tools and methods are available like stop word removal,sampling, transformation, denoising, normalizing etc[26]. Sampling is a method where a representative subset is created form a large number of raw data. Transformation is a process where the raw data is manipulated to produce a single data. Denoising is the way to remove the noise from the data. Normalization is to organize data in more efficient way. But the most important sector is stop word removal.

2.2.2 Feature Selection

Feature subset selection is the process where we remove irrelevant and redundant features as much as possible [29]. This reduces the length of dataset and enables the learning algorithms work faster. Feature selection generally have two parts [29]: a selection algorithm that help to generate from the given subset which tries to find an optimal subset; and another one is evaluation algorithms that evaluate how suitable the proposed dataset is. However without the suitable stopping criteria the feature selection process will not work perfectly. There are generally three types of feature selection methods: wrapper method, filter method, embedded methods. Wrapper method is the way where different combination are prepared, evaluated and compare to other combination, where it selects the section of set of features as a search problem. A predictive model is used to score based on the model accuracy. This search process may be methodical where it can use best-first search. An example is the feature elimination algorithm.

Filter, feature selection methods are used to apply scoring to each feature. The highest scored features are selected for the used features or selected to remover from the main dataset. Some examples of some filter methods include Chi square test, information gain and correlation scores. Embedded methods learn while the model is being created it tracks those features which has a great contribute to the good accuracy. Regularization is the most common embedded feature selection method. Examples of embedded methods are LASSO, Elastic Net and Ridge Regression.

2.2.3 Advantages of Feature Selection

We have used feature methods for following reasons:

- Unnecessary features and low sample features have noise in their data, in this situation, classification will remove these noises and help to generate a good performance.
- Complete control of how many and what type of features will generate the algorithms.
- Helps to set minimum set of features to generate output.
- Easy to understand the generated output.

Preprocessing steps includes Stop Word Removal, Stemming, and Document Indexing.

Stop Word Removal

Stop word removal is one of the most commonly used processing technique right now. Search engines used this process to ignore all those words or character that has no value or less value in the time of generating accuracy form the dataset. At the time of creating the index, most engines are programmed to remove some certain words that has less weight. Mainly the list of words or characters that are not added to the final data set is called the stop word list. This saves both time and space as it removes the words at indexing time and ignored at searching time. We remove these set of words as they carry no information [23] (i.e. pronoun, preposition, conjunctions). In English language there are about 400-500 stop words [23]. These are some examples of stop words which helps us to build the system.

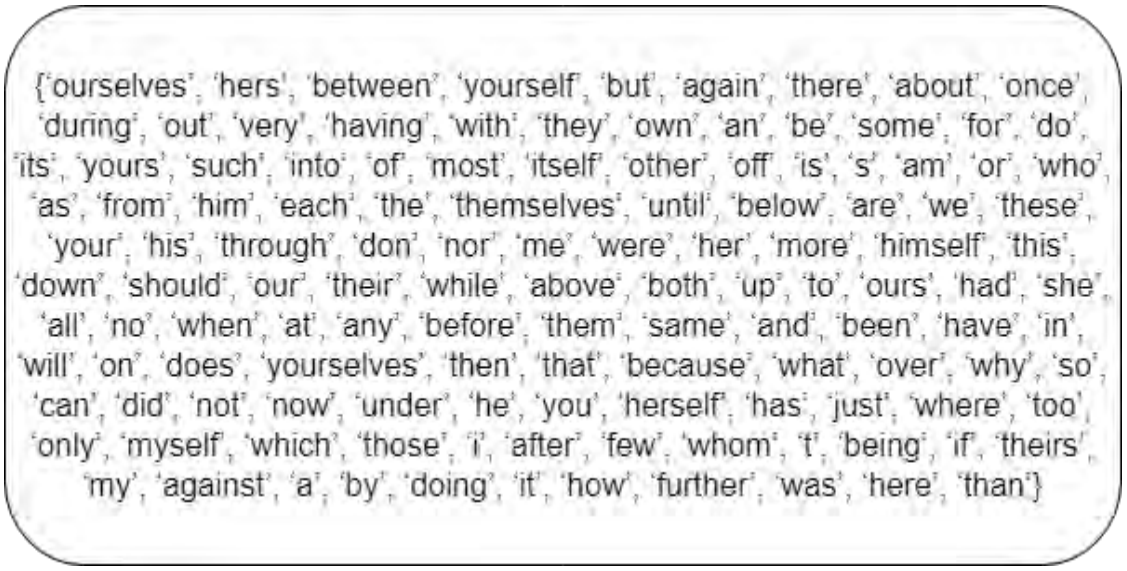

Figure 2.2.1 Stop Words

By removing stop words from the dataset we get high quality language model which is preferred for machine learning algorithms [30]. There are many hypothesis about stop word removal. These can appear at the time of model training [30]:

1. Stop words can herm inference, means noise from frequent words prevents algorithms to reorganizing in content bearing words. By this the pattern of sentence could be changed as well as the meaning.

2. Noise from frequent words does not alter on non-stop words, means no effect on inference.

3. Stop words can improve inference, means it do not herm the pattern of the main meaning of the sentence.

Stemming

Stemming is the process to reduce a bunch of similar words and give them a root word. Stemming is including word and giving a set of words same type of meaning [23]. Here root word is called lemma. Stemming is very important in natural language processing (NLP). In a very big data set a lot of word can be similar, if we can remove the similar meaning words the size of our dataset will reduce and it will reduce run time. When we found a similar type of word we give a particular weight to it for measuring [23]. When a new word is found it is selected as a new type. To get the best result the process of lemma is used. To find lemma, stemming is performed by different machine learning algorithms. Some algorithms will simply strip recognized prefix and suffix. But these type of implementation creates a lot of error which is not suitable for learning algorithms. For an example 'responsively' can be reduced to a word like 'respon'. In suffix stripping the algorithms knows the suffix words and removes the suffix. Another one is Lemmatization. Here the algorithm breaks down the word to its root directory. Then it categorize the word type according to rules of grammar. Another type is Stochastic, it gets the form of how the root words could be inflected and it removes the infected forms of words.

2.2.4 Feature Extraction

Feature extraction is the process to pull out specified data that is significant for particular process. It is used to extract a subset of new features from original feature set, by means of some functional mapping and keeping data Information as much as possible [27]. Most popular feature extraction technique is Principle Component Analysis. In this method it extracts the lower dimension space by analyzing the covariance structure of multivariate statistical observations [28]. Another one is Linear Discriminant Analysis. Here it projects high-dimensional data into lower dimensional space [28]. Variable by variable data cleaning is straight forward filter approach. There are many authors' focuses on duplication of data, which reduce the run time of algorithms.

Filter evaluation is the process to data reduction, it does not consider the activities. The values that are suspicious due to their relationship to a specific probability

distribution, say a normal distribution with a mean id of 5, a standard deviation of 3 and a suspicious value of 10[29]. Table 1 shows some examples of how this metadata can help on detecting a number of possible of data quality problems.

Table 2.2.4.1 Types of values to filter

Problems	Metadata	Examples/Heuristics
Illegal values	Cardinality	e.g., cardinality (gender)> 2 indicates problem
	max, min	max, min should not be outside of permissible range
	variance, deviation	variance, deviation of statistical values should not be higher than threshold
Misspellings	feature values	sorting on values often brings misspelled values next to correct values

Incomplete data is a big problem while dealing with the raw datasets. There can be many factors for this incompletion of data. The common problem occurs because of "unknownness". Data can be missing because of the value is forgotten or lost. Similarly data can be unknown in result because of certain features is not applicable for it or the value that is in don't care mode [29]. To recover these values or to work with the data set we need to handle the missing data. Many methods are available such as method of ignoring instance with unknown feature values, most common feature values, mean substitution, regression or classification methods hot deck input, method of treating missing feature values as special values etc. Method of ignoring instance with unknown feature values is one of the simplest methods. Here it ignores values which have at least one unknown feature. Another one is the concept about the most common feature. In the dataset which occurs on same class is selected as the value of the unknown feature.

Next one is the mean substitution, here it substitutes a features mean value to fill in missing data values on remaining cases. Another one is the regression or classification methods. It is based on complete case data for a given feature. Hot deck imputation is similar case with the missing value and substitute with the most similar cases. In method treating missing feature value, it treats itself unknown and make new value for the future.

2.2.5 Bag of words

Bag of words is a segment that treats every word a feature. It is widely used in text classification area. This is a way of modeling data with machine learning. The main idea is to categorize data, analyze and classify different bag of words. When we try to model the text it become more difficult as it has a lot of noise inside it. So to make a category out of these messy data, bag of word comes up. It tries to segment a set of data which will be the representation of a feature. This set will be the core thing to select other similar items [32]. It tries to find the occurrence of words in the document but it is not concern where the words occur in the document. It represents the text that describes the occurrence of text in the dataset. It consist of two things one is vocabulary of known things and another one is measure of presence of words [31]. First step is to collect the data and in our case here is the dataset. Then it designs the vocabulary, means it finds the unique words and create document vector. In case of document vector the words are converted into binary vector where it puts a value for each unique word [31]. For example, if we consider 2 text samples:

1. “the fifa 2018 world cup has become the most exciting world cup ever”

2. “the fifa 2018 world cup is becoming the most viewed world cup”.

Here the text sample is the form of a dictionary where vectors are formed to represent the count of each word in every sentence.

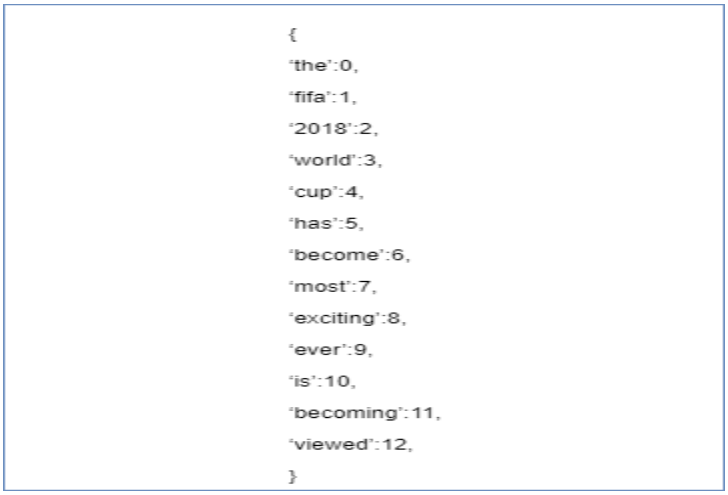

Figure 2.2.5 Distributed value for each

In figure 2.2.5, we have 12 unique words which do not overlap with each other. For this each text sample will generate 12 element vectors. Which will indicate the occurrence of the words in the sentence.

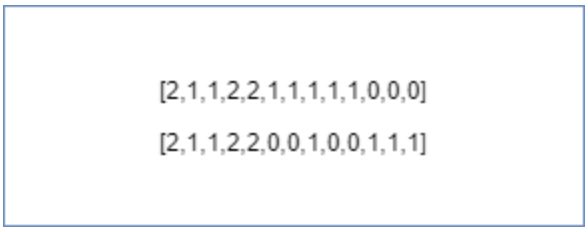

Figure 2.2.5 Weight of each word per line

In figure 2.2.5, each element represents the number of words that are present in the sentences. For the first row the word “the” comes first and the occurrence of it is twice in the first sentence. That’s why it is marked as 2 and next word is “fifa” which occurs two times in the same sentence. For the first sentence the word “viewed” do not appear so it is not valued as 0. Similarly all other words are taken into this process [33].

There are some problems in scoring of words, one of the main reason is the high frequency words dominate the document. But this may not contain much important information [31]. For an example, the word “the”, this word can stand almost in front of anything, but do not contain much information to separate from other. One approach is to rescale the frequency of the word of how often they appear in the data. And another concern is to differentiate between two almost similar sentences. To solve this problem TF-IDF come forward. This scoring approach is called Term Frequency – Inverse Document Frequency.

CHAPTER 3

Workflow and Feature Extraction

3.1 Workflow

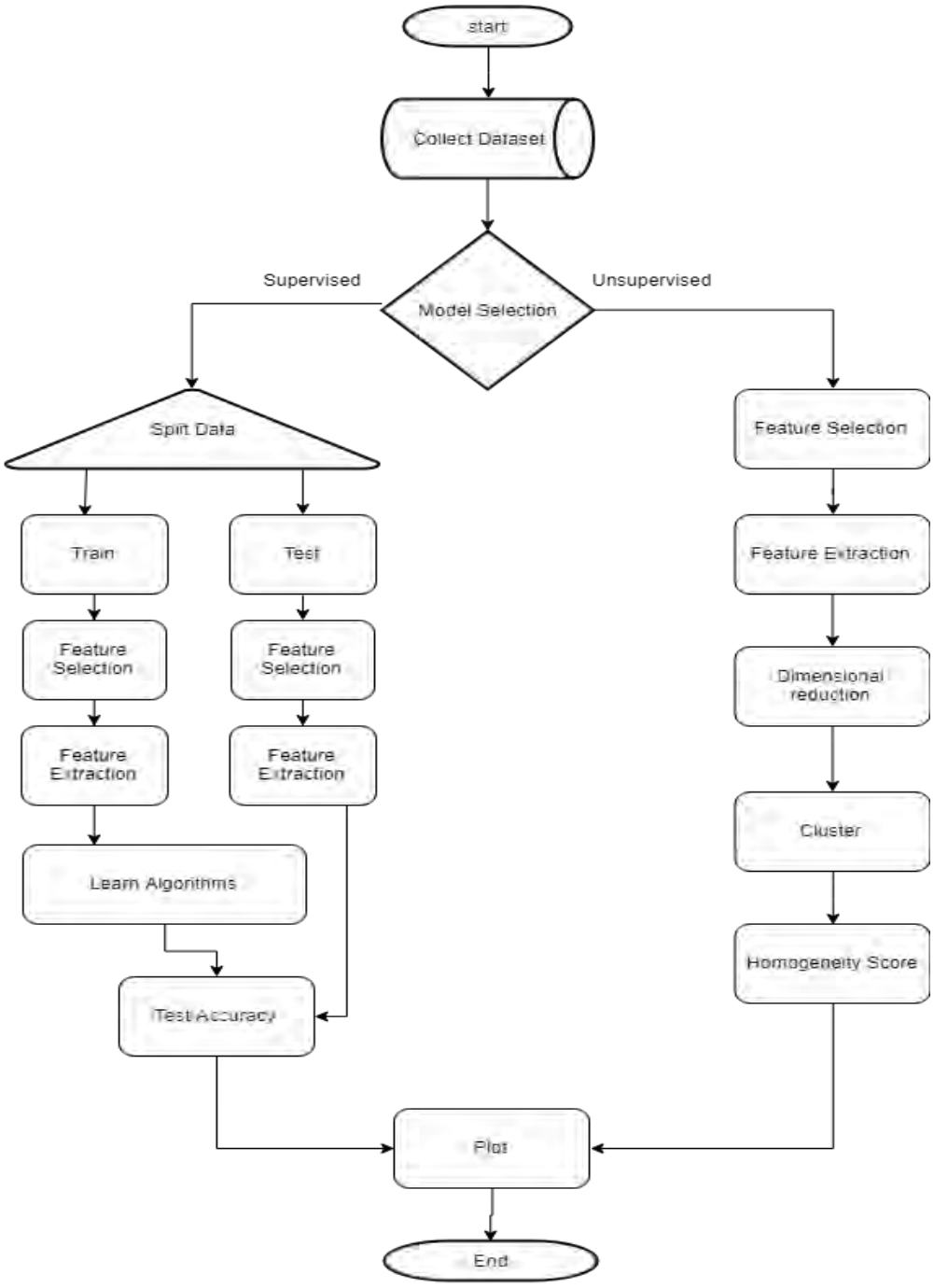

Figure 3.1 Proposed System

This chapter will shed light on the process which we followed. We started by loading the dataset. Then we split the dataset to train and test. We cleaned the test and train set using several pre-processing methods such as stemming, lemmatization and removing stop words and we also removed headers, footers and quotes. Following, we extracted features using bag of words representation. Now the features are ready for supervised algorithms. We have used 4 supervised algorithms: Naïve Bayes, Support Vector Machine, K-Nearest Neighbor and Decision Tree. At last, we plotted the accuracy of the classifiers tested by the test set. For unsupervised algorithm we have used the whole dataset and vectorize the dataset. For fitting the unsupervised algorithms, we normalized the dataset using dimensionality reduction. We have used K-means and Mini Batch K-means for clustering. We then plotted the homogeneity scores of unsupervised algorithms.

3.1.1 20 Newsgroup Dataset

The 20 Newsgroups dataset is commonly used for text mining applications. It was collected by Ken Lang. The 20 Newsgroups data set is a test collection of approximately 20,000 newsgroups documents that 1000 documents were taken from each of the newsgroups. It is divided across 20 different newsgroups. The category topics are related to computers, politics, religion, sports, and science. Each document belongs to exactly one newsgroup, but there is a small fraction of the articles belong to more than one category. The data collection is the well-known 20-Newsgroups (20NG) dataset.

Table 3.1.1.1: 20 Newsgroups Categories

comp.graphics	rec.autos	sci.crypt
comp.os.ms-windows.misc	rec.motorcycles	sci.electronics
comp.sys.ibm.pc.hardware	rec.sport.baseball	sci.med
comp.sys.mac.hardware	rec.sport.hockey	sci.space
comp.windows.x		

misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Table 3.1.1.2: Documents in 20 Newsgroups

Topics	# Documents in testing	# Documents in training	# Documents
alt.atheism	319	480	799
comp.graphics	389	584	973
comp.os.ms-windows.misc	394	591	985
comp.sys.ibm.pc.hardware	392	590	982
comp.sys.mac.hardware	385	578	963
comp.windows.x	395	593	988
misc.forsale	390	585	975
rec.autos	396	594	990
rec.motorcycles	398	598	996
rec.sport.baseball	397	597	994
rec.sport.hockey	399	600	999
sci.crypt	396	595	991
sci.electronics	393	591	984
sci.med	396	594	990
sci.space	394	593	987
soc.religion.christian	398	599	997
talk.politics.guns	310	465	775
talk.politics.mideast	364	546	910
talk.politics.misc	376	564	940
talk.religion.misc	251	377	628

Total	**7532**	**11314**	**18846**

The categories of the dataset are shown in Table 3.1.1.1 some news-groups, for example, the category “comp.sys.ibm.pc.hardware” and “comp.sys.mac.hardware” are very similar to each other. An example of the 20 newsgroups dataset document shown in Fig 1. From the document example, it contains more than headers such as subject and from. Subject header holds the title of document and from header holds the email address for the sender.

Moreover, there are different versions of the dataset. The first version known as original dataset contained 19997 documents. The second version that shows in Table 3.1.1.2 contains 18846 documents. It’s called “bydate” version. Another version is 18828 version which is more cleaned and has 18828 documents. We used the "bydate" version since cross-experiment comparison is easier (no randomness in train/test set selection), newsgroup-identifying information has been removed and it's more realistic because the train and test sets are separated in time. For the purpose of testing the dataset by supervised algorithms we have split the dataset into 80% training and 20% test data.

3.2 Applied Methods For Cleaning

The Main challenge of working with text data is cleaning. For our thesis we tried to clean the data to some extent so that the data would be more realistic and would generalize to other documents that aren’t from this window of time.

From: mathew <mathew@mantis.co.uk>

Subject: Re: university violating separation of church/state?

Organization: Mantis Consultants, Cambridge. UK.

X-Newsreader: rusnews v1.01

Lines: 29

dmn@kepler.unh.edu (...until kings become philosophers or philosophers become kings) writes:

>Recently, RAs have been ordered (and none have resisted or cared about

> it apparently) to post a religious flyer entitled _The Soul Scroll: Thoughts

> on religion, spirituality, and matters of the soul_ on the inside of bathroom

> stall doors. (at my school, the University of New Hampshire) It is some sort

> of newsletter assembled by a Hall Director somewhere on campus. It poses a

> question about 'spirituality' each issue, and solicits responses to be

> included in the next 'issue.' It's all pretty vague. I assume it's put out

> by a Christian, but they're very careful not to mention Jesus or the bible.

> I've heard someone defend it, saying "Well it doesn't support any one religion.

> " So what??? This is a STATE university, and as a strong supporter of the

> separation of church and state, I was enraged.

>

>What can I do about this?

It sounds to me like it's just SCREAMING OUT for parody.Give a copy to your

friendly neighbourhood SubGenius preacher; with luck, he'll run it through the

mental mincer and hand you back an outrageously offensive and gut-bustingly

funny parody you can paste over the originals.

I can see it now:

The Stool Scroll

Thoughts on Religion, Spirituality, and Matters of the Colon

Figure 3.2 Dataset Sample

Figure 3.2 shows a sample document. Here we can see some noises. Noisy text does not comply with rules basic program uses to identify and categorize words, phrases and clauses in a particular language. Poor spelling and punctuation, typographical errors noises the data. For cleaning the data first while loading the data we will not include headers, footers, quotes. Headers, footers, quotes are irrelevant for categorization. The function that load 20 newsgroup data provide a parameter called remove telling it what kind of information to strip out of each file. The next step we used was to remove special characters such as "#", "@" and "/". In this stage we used python Scikit-learn module to load the dataset and then regex module to replace all these special characters as " " (space). The regex expression we used: " |(\\(.*?){)|}|[!$%^&*#()_+|~\-={}\[\]:\";'<>?,.\/\\]|[0-9]|[@] ". We also used regex to remove extra space. Then we have done stemming. Stemming is a part of linguistic studies in morphology and artificial intelligence information retrieval and extraction. Stemming extract meaningful information from vast sources.

Recognizing, searching and retrieving more forms of words returns more results. When a form of a word is recognized it can make it possible to return search results that otherwise might have been missed. That additional information retrieved is why we use stemming in our data pre-processing phase. Often, the best results can be attained by using the basic morphological form of the word: the lemma. To find the lemma, stemming is performed by an individual or an algorithm. Stemming uses a number of approaches to reduce a word to its base from whatever inflected form is encountered.

It can be simple to develop a stemming algorithm. Some simple algorithms will simply strip recognized prefixes and suffixes. A stemmer for English operating on the stem 'cat" should identify such strings as "cats", "catlike", and "catty". A stemming algorithm might also reduce the words "fishing", "fished", and "fisher" to the stem "fish". The stem need not be a word, however. The most popular stemming algorithm is "Porter stemmer". However, we used "Snowball stemmer" instead of "Porter stemmer". "Snowball stemmer" is also called "Porter2" stemming algorithm. Porter stemmer stem aggressively and error prone. On the other hand, Snowball stemmer can handle many words nicely. For a simple comparison we give these words as inputs to both the stemmers: "python", "pythoner", "pythoning", "pythoned", "pythonly".

Table 3.2.1: Differences between types of Stemmer

Porter Stemmer	Snowball Stemmer
>>python	>>python
>>python	>>python
>>python	>>python
>>python	>>python
>>pythonli	>>python

Table 3.2.1 shows a better view of the differences. Algorithm used in Snowball Stemmer provides a significant reduction in the complexity of the rules associated with.

For the next phase we have used “Lemmatizing”. The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

However, the two words differ in their flavor. Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma. Stemming increases recall while harming precision. As an example of what can go wrong, note that the Porter stemmer stems all of the following words: “operate”, “operating”, “operates “, “operation”, “operative”, “operatives”, “operational” to “oper”. However, since “operate” in its various forms is a common verb, we would expect to lose considerable precision on queries. The word "better" has "good" as its lemma. This link is missed by stemming, as it requires a dictionary look-up. In terms of implementation, lemmatization is usually more sophisticated. We implemented it by using python NLTK module “lemmatizer”. At the end of the pre-processing phase we have blocked all the stop words. "stop words" usually refers to the most common words in a language. These are some of the most common, short function words, such as “a”, “the”, “is”, “at”, “which” etc. Removing these stop words will result in less term frequencies and more realistic data.

3.2.1 Feature Extraction Process

Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power, also it may cause a classification algorithm to overfit to training samples and generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy. There are many feature extraction methods. Most famous is "Bag of Words" which is also known as "Tf-Idf Vectorizer". TF-IDF reflect the context of the sample better than other [33]". Tf-Idf (Term Frequency – Inverse Document Frequency) reflect the context of the sample better than other [33]. There is another method of feature extraction called "Hashing Vectorizer" which only uses TF (Term Frequency) for extracting features. TF-IDF consists of two values. One of them is Term Frequency (TF) and other one is Inverse Document Frequency (IDF). The common way to determine TF is to taking the raw frequency of a term and divide by the maximum frequency of any term in the document [33].

$$TF = 0.5 + 0.5 * freq(l)/\max(freq(k)) \quad \text{..... (ii)}$$

Here, k=all words in document and l= tram in document. And another one is to take the log of the inverse of the proportion of documents c obtaining the term.

$$\text{IDF} = \log(m/len(n)) \quad \text{..... (iii)}$$

Here m= $document_{count}$ and n= $documents\ containing\ term$

Table 3.2.1.1 Process of TF-IDF

Unique Words	TF		IDF	TF-IDF	
	A	B		A	B
I	1	1	Log(2/2)	0	0
Am	1	1	Log(2/2)	0	0
fine	1	1	Log(2/2)	0	0

not	0	1	Log(2/1)	0	1

Here we can see that we have two sample dataset as D1 and D2. Here we the separated unique words are taken into consideration. The first line is set as true and the second line is set as false. And the difference of these two sentence is just “not”. If we do not use TF-IDF here our program will just ignore the value of “not” as it will not contain much value. So after applying TF-IDF we can see the combined result which is almost same for two sentences except one. Which is the negative symbol to separate the two sentences. So in this process we can differentiate between two similar types of data.

Before cleaning and fitting the vectorizer the shape of train and test set was (15076, 20823) (3770, 20823). After cleaning and fitting the vectorizer the dataset the shape of train and test set was (15076, 13116) (3770, 13116). So we can clearly see cleaning the dataset has reduced the term frequencies of the dataset.

CHAPTER 4
Implementation and Results

4.1 Supervised Algorithms

After cleaning and feature extraction the documents can be easily represented in a form that can be used by a ML algorithm. Many text classifiers have been proposed in the literature using machine learning techniques, probabilistic models, etc. We have used four of the most used machine learning algorithms for text classification: Naïve Bayes, Support Vector Machine, K-Nearest Neighbors and Decision Tree.

4.1.1 Naive Bayes

Naive Bayes is often used in text classification applications and experiments because of its simplicity and effectiveness. Naive Bayes is a kind of classifier which uses the Bayes Theorem. Bayes theorem named after Rev. Thomas Bayes [https://en.wikipedia.org/wiki/Thomas_Bayes]. It works on conditional probability. Conditional probability is the probability that something will happen, given that something else has already occurred. For a document **d** and a class **c**, and using Bayes' rule,

$$p(c|d) = [p(d|c) * p(c)]/p(d)] \quad \text{..... (iv)}$$

Here, classes are the categories the documents belong. P(c) is the total probability of a class. We have used Multinomial Naïve Bayes rather than Gaussian Naïve Bayes algorithm. Multinomial Naïve Bayes works well for data which can easily be turned into counts, such as word counts in text. Here simply the documents are represented as a set of features (x1, x2, x3, …). We have set the parameters of Multinomial Naïve Bayes as "alpha=.01", "prior=false". Here "alpha=.01" counter the problem with maximum likelihood which is if any document didn't match any classes Naïve Bayes will mark it as 0 probability. Zero probabili1es cannot be conditioned away. By "alpha=.01" we have done Laplace smoothing (add-.01). By giving "prior=false" We ensured that every document consist a uniform prior

4.1.2 K nearest Neighbors

K nearest Neighbors classifier is based on the assumption that the classification of an instance is most similar to the classification of other instances that are nearby in the vector space. Compared to other text categorization methods such as Bayesian classifier, KNN does not rely on prior probabilities, and it is computationally efficient. The main computation is the sorting of training documents in order to find the knearest neighbors for the test document. The number of k determines how many neighbor it will take and measure the similarity.

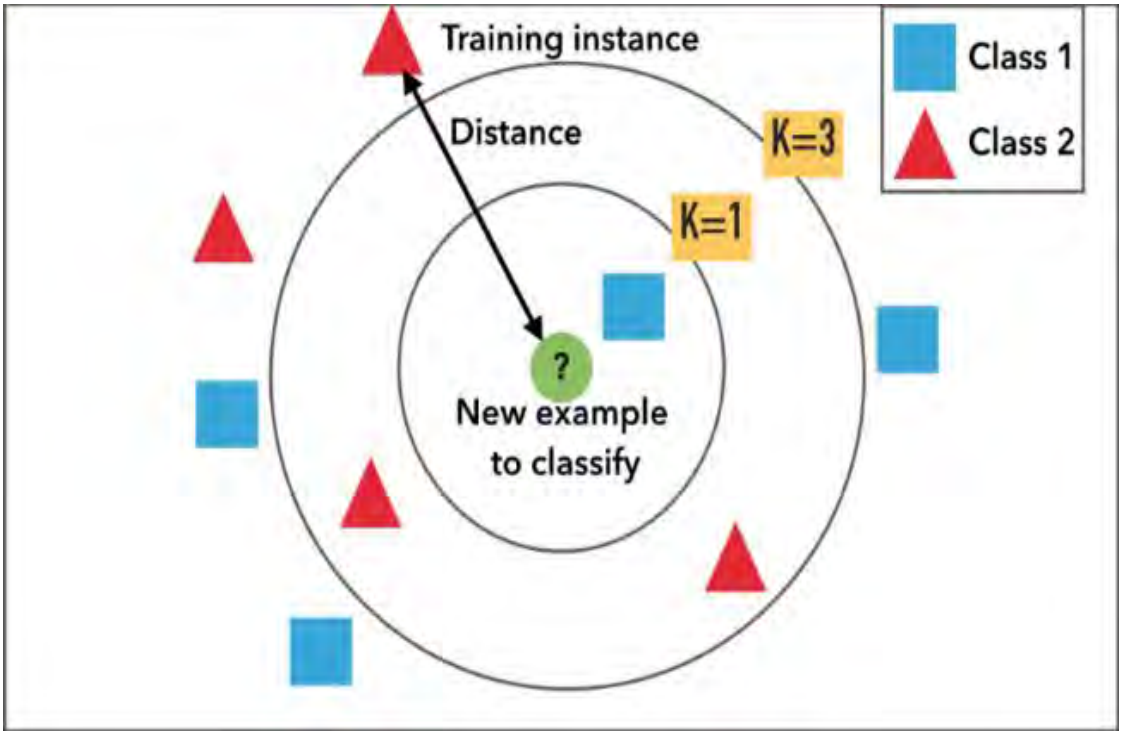

Figure 4.1.2 Basic KNN example [20]

To classify a class-unknown document ***X***, the *k*-Nearest Neighbor classifier algorithm ranks the document's neighbors among the training document vectors and uses the class labels of the k most similar neighbors to predict the class of the new document. The classes of these neighbors are weighted using the similarity of each neighbor to ***X***, where similarity is measured by Euclidean distance or the cosine value between two document vectors. The cosine similarity is defined as follows:

$$sim(X, Dj) = \frac{\sum_{t_a \in (X \cap D_e)} X_a * d_{ae})}{||X||_2 * ||D_e||_2} \quad \text{..... (v)}$$

Where X is the test document, represented as a vector; D_e is the eth training document; t_a is a word shared by X and D_e; X_a is the weight of word t_a in X; d_{ae} is the weight of word t_a in document D_e; $||X||_2 = \sqrt{x_1^2 + x_2^2 + x_3^2 + \cdots}$ is the norm of X, and $||D_e||_2$ is the norm of D_e. Here we have used k = square root of training samples. Using square root of training samples is the thumb rule of getting optimal value.

4.1.3 Decision Tree

Decision Tree creates a training model which can used to predict class or value of target variables by learning decision rules inferred from training data. The Decision tree algorithm tries to solve the problem, by using tree representation. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label. Decision Tree is based on the strategy of divide and conquer.

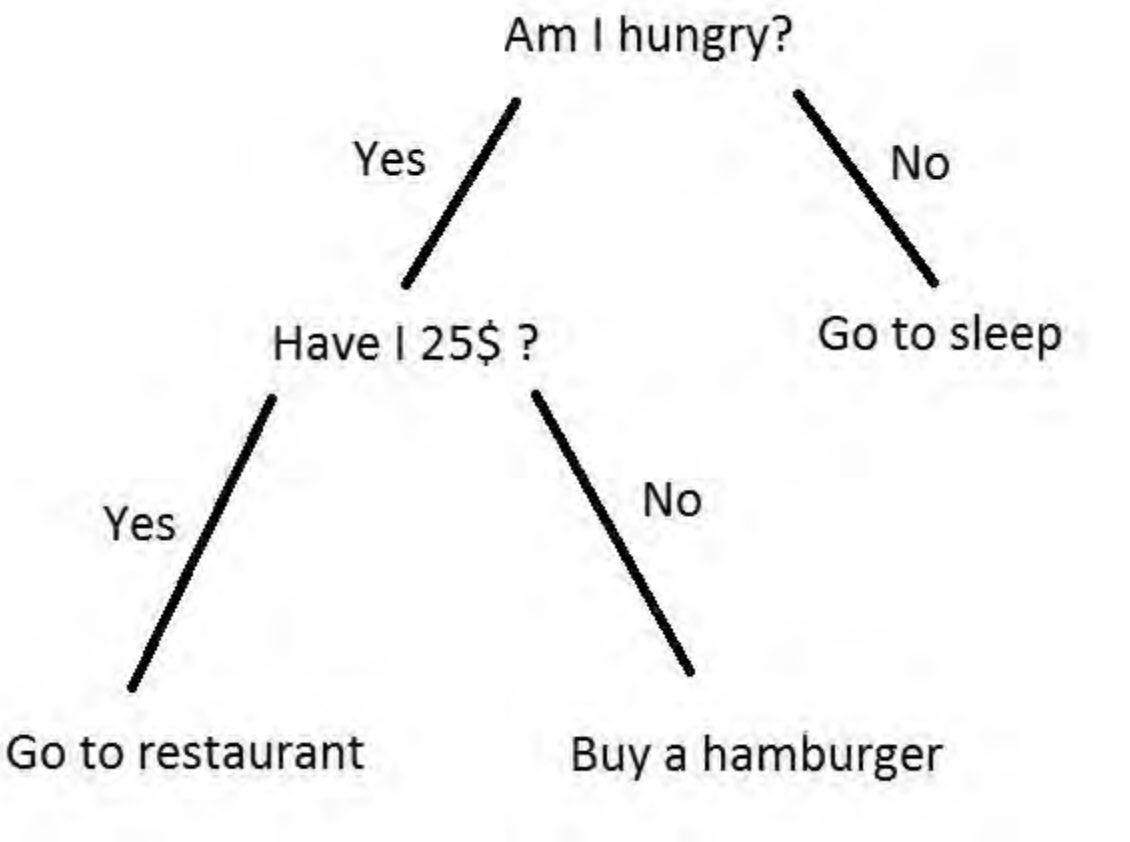

Figure 4.1.3 Decision Tree [22]

In general, this philosophy is based on the successive division of the problem into several sub problems with a smaller number of dimensions, until a solution for each of the simpler problems can be found. While implementing we have set the parameters as “criterion=gini", “splitter=best”. Here criterion measure the quality of a split. To measure

the quality, we have used Gini Impurity. Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset [22]. At each node while the tree splits, we chose the "best" parameter to choose the best split it possibly can.

4.1.4 Support Vector Machine

Support Vector Machine focus only on the points that are the most difficult to tell apart, whereas other classifiers pay attention to all of the points. The intuition behind the support vector machine approach is that if a classifier is good at the most challenging comparisons then the classifier will be even better at the easy comparisons. Basically, the goal of the support vector machine to design a hyperplane that classifies all training vectors in two classes. The best choice will be the hyperplane that leaves the maximum margin from both classes.

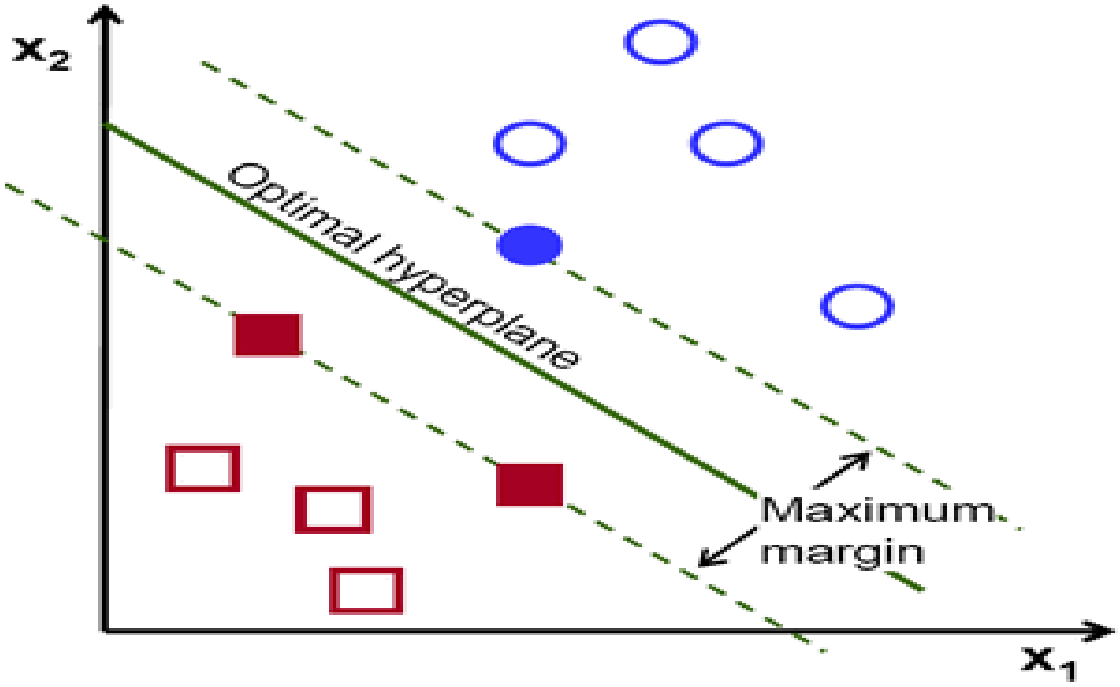

Figure 4.1.4 Basic SVM Diagram [18]

We have used LinearSVC of python module to implement Support Vector Machine. LinearSVC is implemented in terms of liblinear. Liblinear is a linear classifier for data with millons of instances and features.The parameter we have given was: "loss=hinge", "penalty=l2", "tol=1e-2", "max_iter=100", "random_state=50". "Loss=hinge" gives a linear SVM and penalty is a regularization term which is set to l2 implementing l2 regularization. The regularization parameter serves as a degree of importance that is given to miss-classifications. Here "tol" value is a constant that

multiplies the regularization term. “max-iter” represents the number of passes over the training data. We have run 100 epoches by “max_iter=100” parameter. Lastly classification scores depend on “random_state”. It is the seed of the pseudo random number generator to use when shuffling the data.

4.2 Unsupervised Algorithms

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

$$j = \sum_{j=1}^{k} \sum_{i=1}^{n} \left|\left| x_i^{(J)} - C_J \right|\right|^2 \quad \text{..... (vi)}$$

Here, k=number of cluster, j=objective function, n=number of cases, Cj=centroid for cluster j, Xi = case i .We have used kmeans clustering with the same 20 newsgroup dataset and with k = number of categories. Before training we used SVD (Singular Value decomposition) to reduce the dimensionality. As the dimensionality of data increases, the volume of the space increases, in a sense the data becomes more and more sparse (scattered). The parameters we have used: “init=k-means++”, “max_iter=100”, “n_init=1”. Here “init=k-means++” selects initial cluster centers for k-mean clustering in a smart way to speed up convergence. We have run the k-means algorithm with 100 iterations at a time with the “max_iter=100” parameter. “n_init=10” function determines 10 times the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.

4.2.1 MiniBatch Kmeans

MiniBatchKmeans is a modified version of kmeans clustering. Time complexity of kmeans clustering is O(n*K*I*f), where n is the number of documents, K is the number of clusters we want, I is the number of iterations and f is the number of features in a particular record. It can be clearly seen that this will take a lifetime for the original algorithm to cluster data. On the other hand, MiniBatchKMeans algorithm takes small batches (randomly chosen) of the dataset for each iteration. It then assigns a cluster to each data point in the batch, depending on the previous locations of the cluster

centroids. It then updates the locations of cluster centroids based on the new points from the batch. The update is a gradient descent update, which is significantly faster than a normal Batch K-Means.

4.3 Result

For evaluating scores, we have measured F1-scores. F1-scores is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score: p is the % of selected items that are correct, and r is the % of correct items that are selected. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

$$F1 = 2 * \frac{precision * recall}{precision + recall} \quad \text{.....(vii)}$$

We have run our classifiers for 3 different length of categories: 20, 10 and 4. The more categories the less accurate result and more time needed to fit the classifiers.

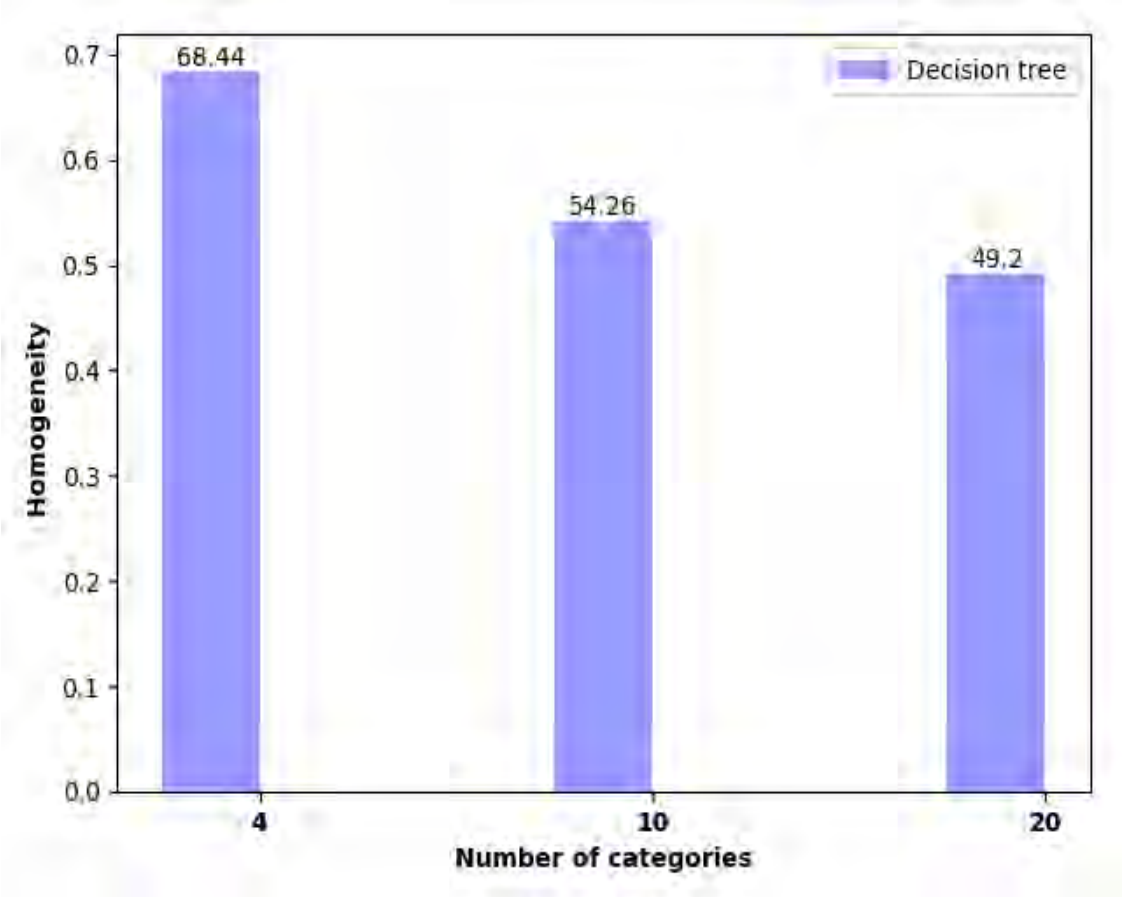

Figure 4.3.1 Decision Tree

Decision tree creates nodes of the tests and create subgroups until each of the set addressing one class. However, Decision tree creates biased trees if some classes

have more features from others thus lead to misclassification. Again, a small change can cause the tree to change its path so it is unstable. Here, we can see decision tree has a low f1 score of 49.2% when we used the full dataset and 68.44% when we used only the 4 categories. Since we have some categories which are really close like 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware'. This causes the tree to misclassify. Our data includes categorical variables with different number of levels, information gain so decision trees is biased in favor of those attributes with more levels.

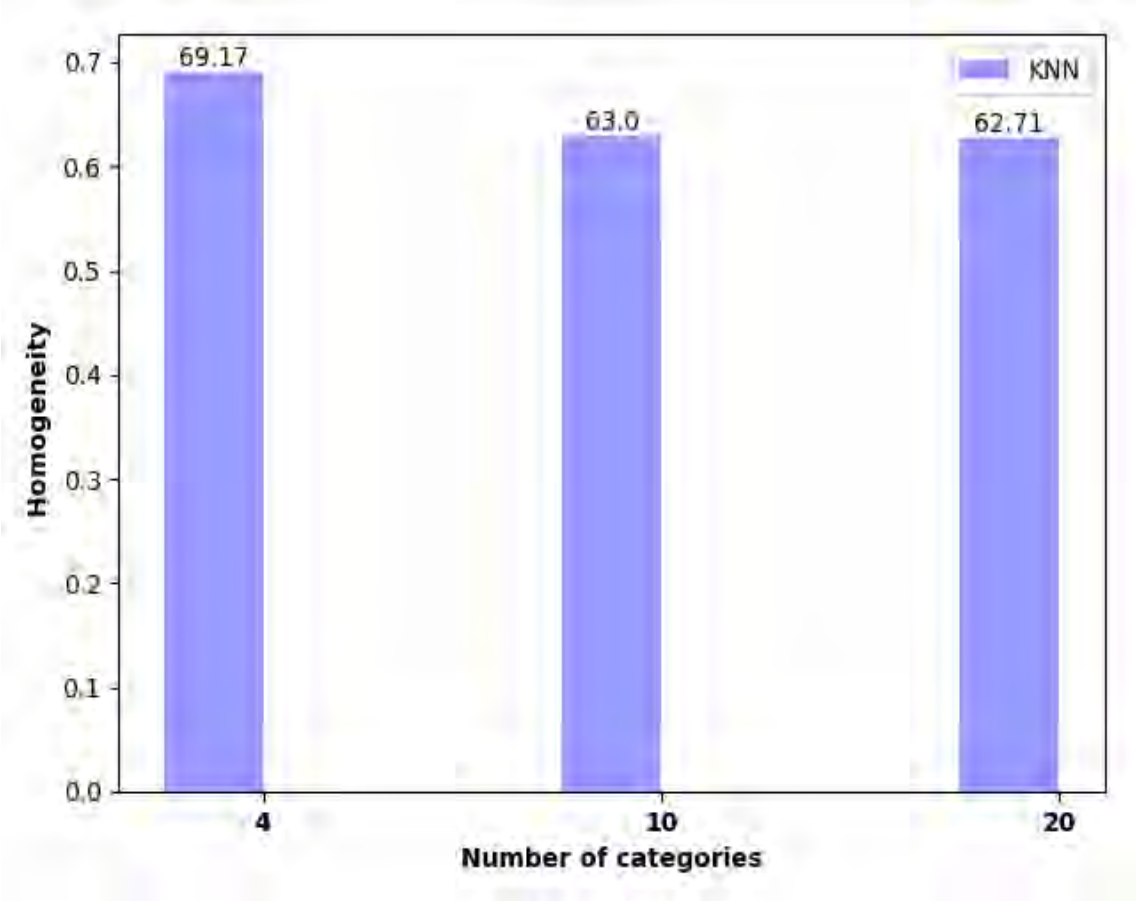

Figure 4.3.2 KNN

KNN works based on minimum distance from the query instance to the training samples to determine the k-nearest neighbors. After we gather k nearest neighbors, we take simple majority of these k-nearest to be the prediction of the query instance. However, KNN doesn't handle categorical variable very well. It uses the class labels of nearest neighbors to determine the class label of unknown record. We had to determine the k for the best result because, if k is too small it is sensitive to noisy points and if k is too large neighborhood may include points from other classes. Here, while

categorizing for 20 categories KNN's accuracy is degraded to 62.71% because there are many features and every feature have weights and determining classes using only distance caused misclassifying as there can be many irrelevant features. For a small amount of data like for only 4 categories KNN shows a prediction of 69.17%.

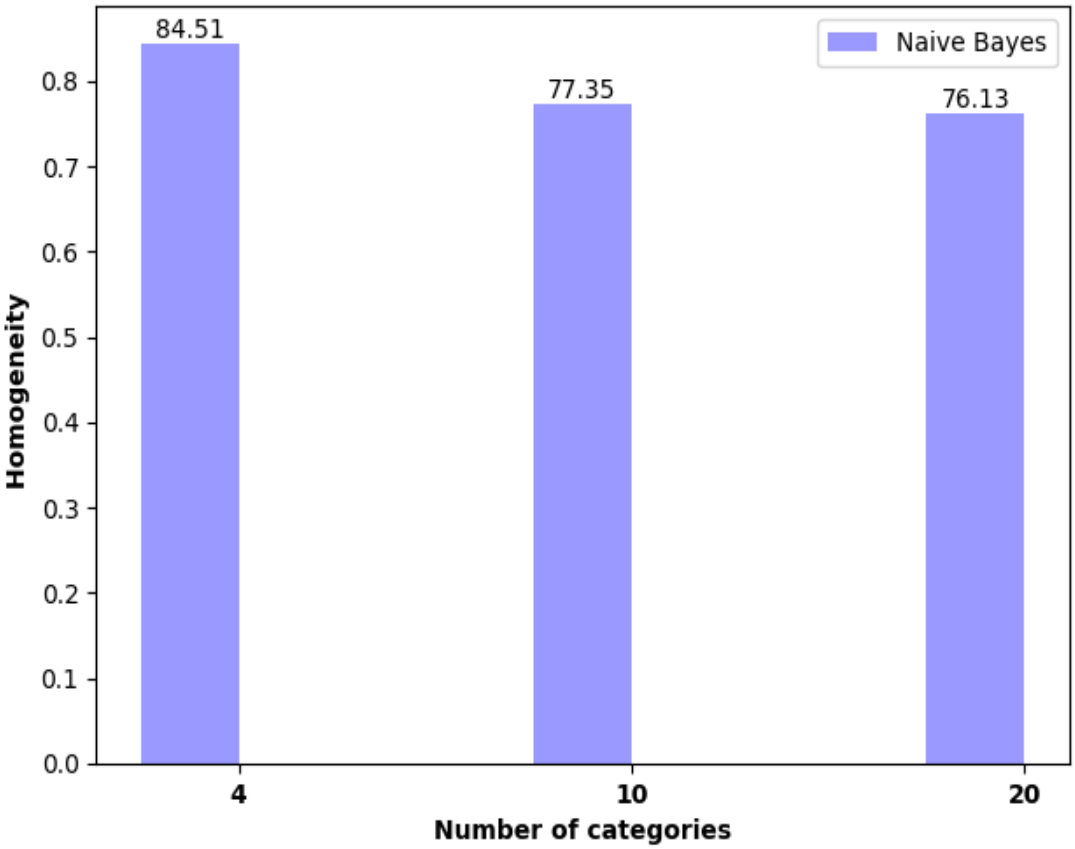

Figure 4.3.3 Naïve Bayes

Naïve Bayes classifier is suitable for classification with discrete features like word counts for text classification. Naïve Bayes use probability to predict classes. We can see that while predicting for 4 categories it scored 84.51%. While classifying Naïve Bayes ranks the features in terms of frequency. This sometimes degrade performance. Again, Naïve Bayes classifier makes a very strong assumption on the shape of data distribution like any two features are independent given the output class. Due to this, accuracy got lower while categorizing 20 categories that is 76.13%.

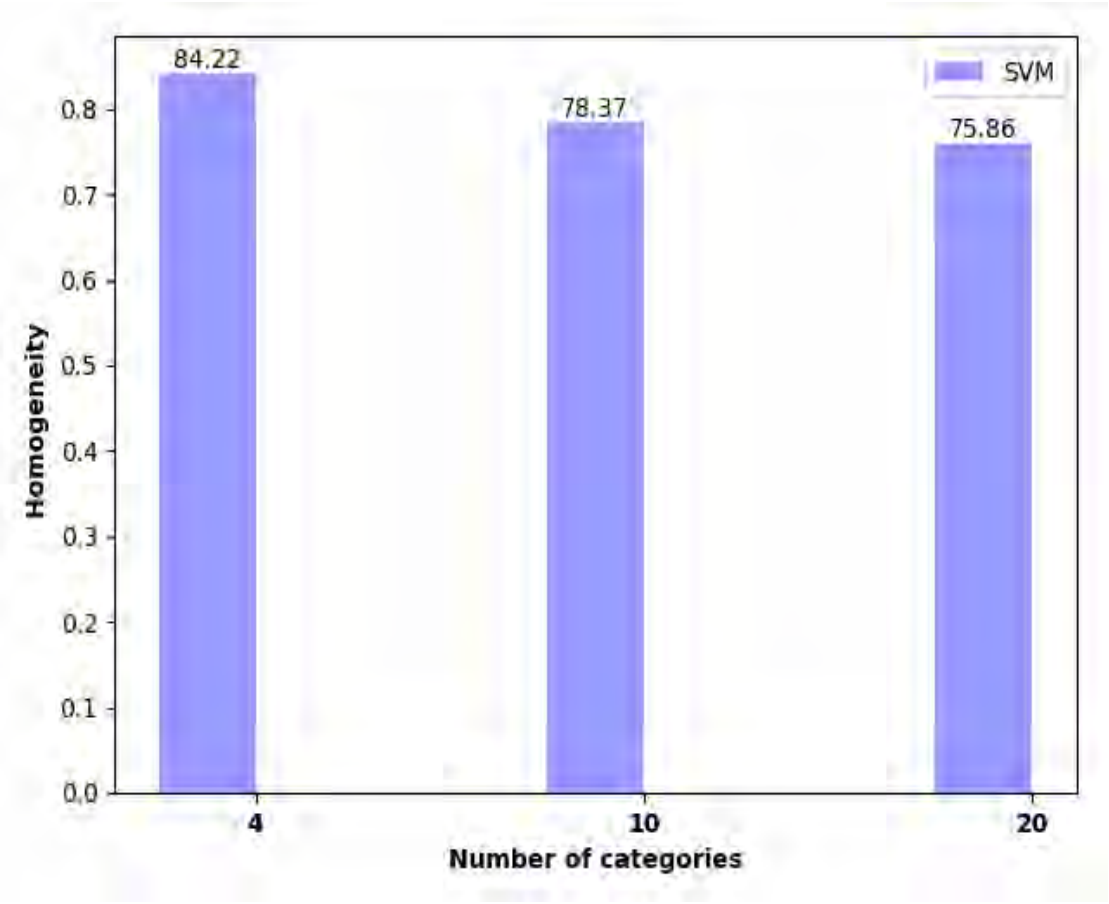

Figure 4.3.4 SVM

SVM (support vector machine) doesn't depend on the number of features. All features ranked according to their binary information gain. Naïve Bayes classifier is trained using only those features which is high ranked. However, features ranked lowest still contain considerable information and are somewhat relevant. SVM use these features to predict. SVM creates a hyperplane which is as far away from the data as possible. Thus, it gives good accuracy by considering the low ranked features. We can see for 4 categories the f1 score is 84.22%. However, we have used bag of words representation which loses sequantiality information and leads to performance degrade. That's why when we try to categories for 20 groups accuracy degraded to 75.86% because there are many features and bag of words representation describes every word independently.

Since there are no labels in unsupervised learning, it's near impossible to get a reasonably objective measure of how accurate your algorithm is. We have evaluated our unsupervised algorithm by the homogeneity. Homogeneity means all of the observations with the same class label are in the same cluster. There is another

measurement for accuracy called Completeness. It means all members of the same class are in the same cluster. There is no such declaration of best results between k-means clustering and minibatch kmeans because unsupervised algorithms improve their result over time and give different result at different time. However, we have run both the classifiers 100 times and saved the scores for plotting.

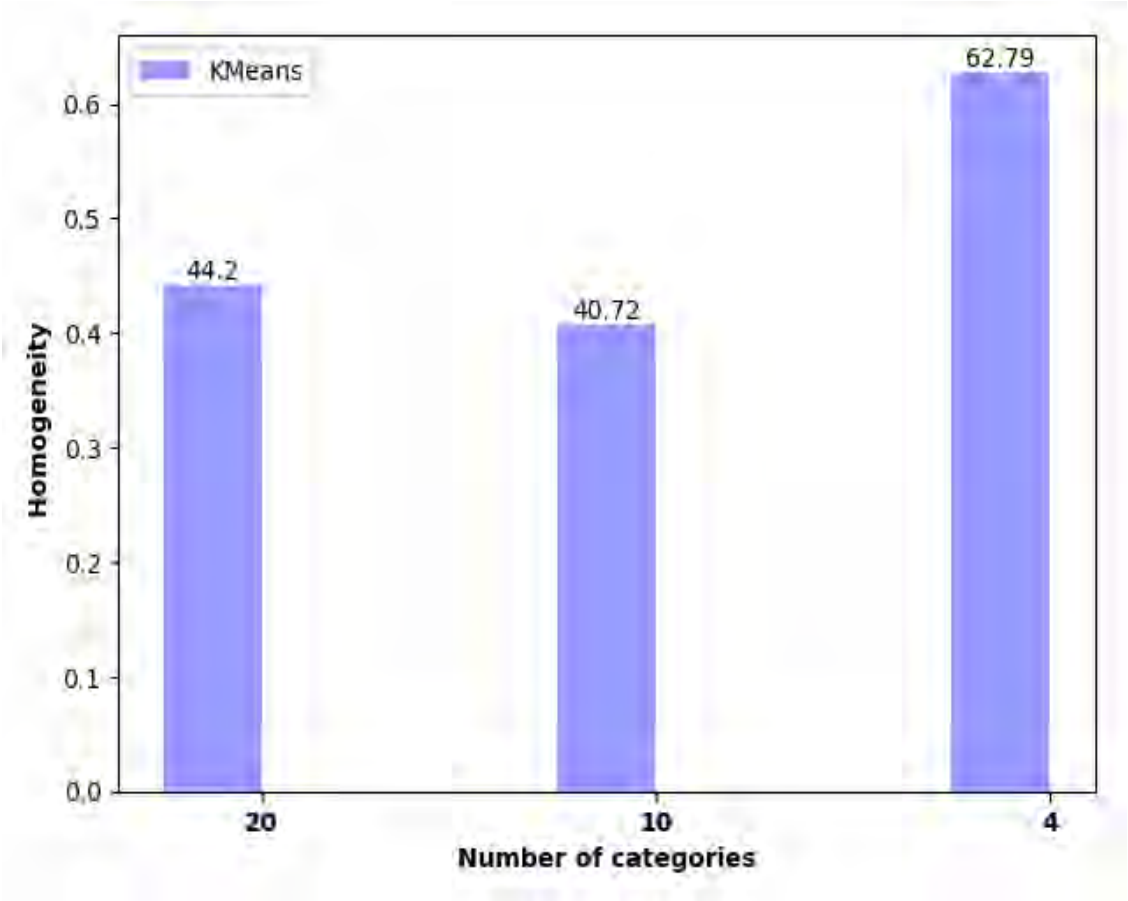

Figure 4.3.5 K-Means

We are using features and every feature have weight it will try to cluster features into a number of mutually exclusive clusters. In figure 4.3.5, we see k means clustering algorithms has a homogeneity of 44.2% for 20 categories and 40.72% for 10 categories and 62.79% for 4 categories. This is the result after 100 iterations. Results vary because unsupervised algorithms learn from the environment, the more time it takes to run the more accuracy it will get.

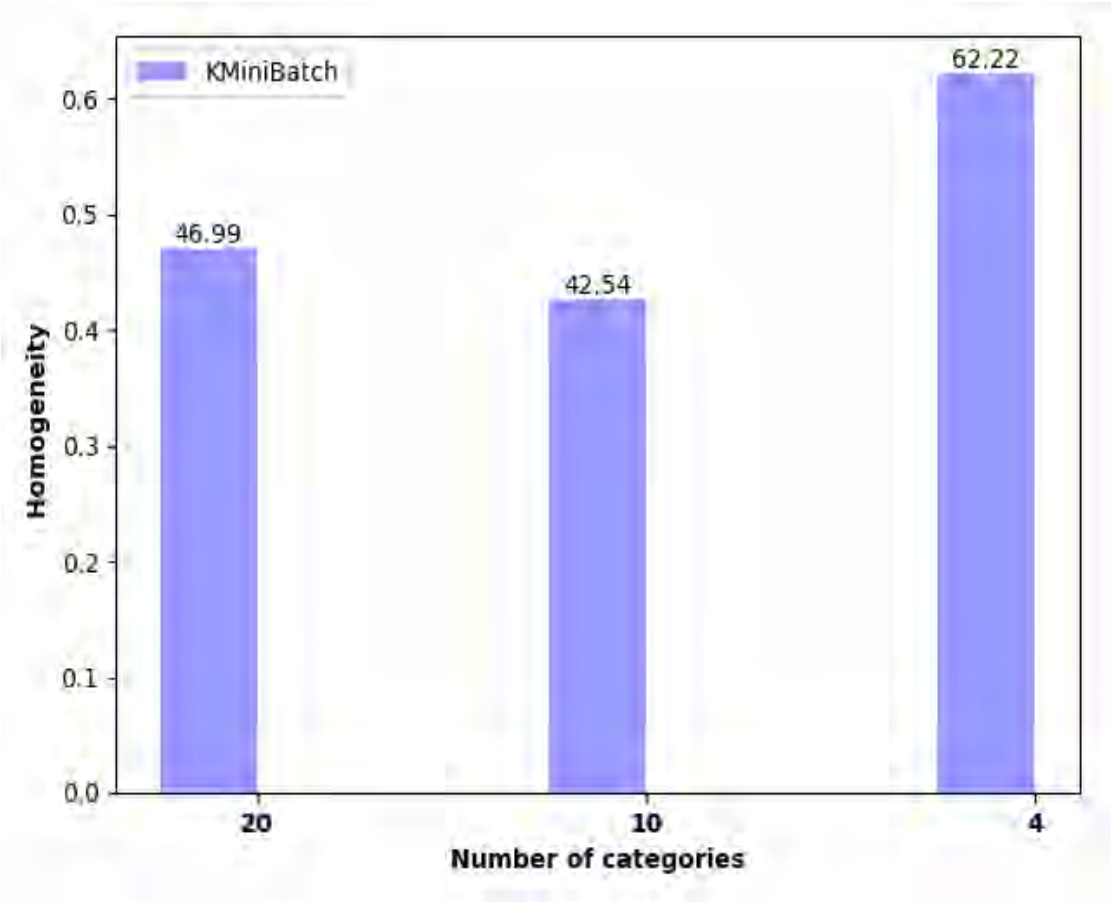

Figure 4.3.6 Min Batch K-means

Mini Batch K-means is faster while clustering 20 categories because it works in batches. It divides the whole samples in some batches for each iteration and then cluster centroids based on the new points from the batch. K-means and Mini Batch K-means differs only on this matter. We can see that in Figure 4.3.6 for 20 categories results have improved because of the batching process. For 20, 10 and 4 categories Mini Batch K-means has homogeneity of 46.99%, 42.54% and 62.22%.

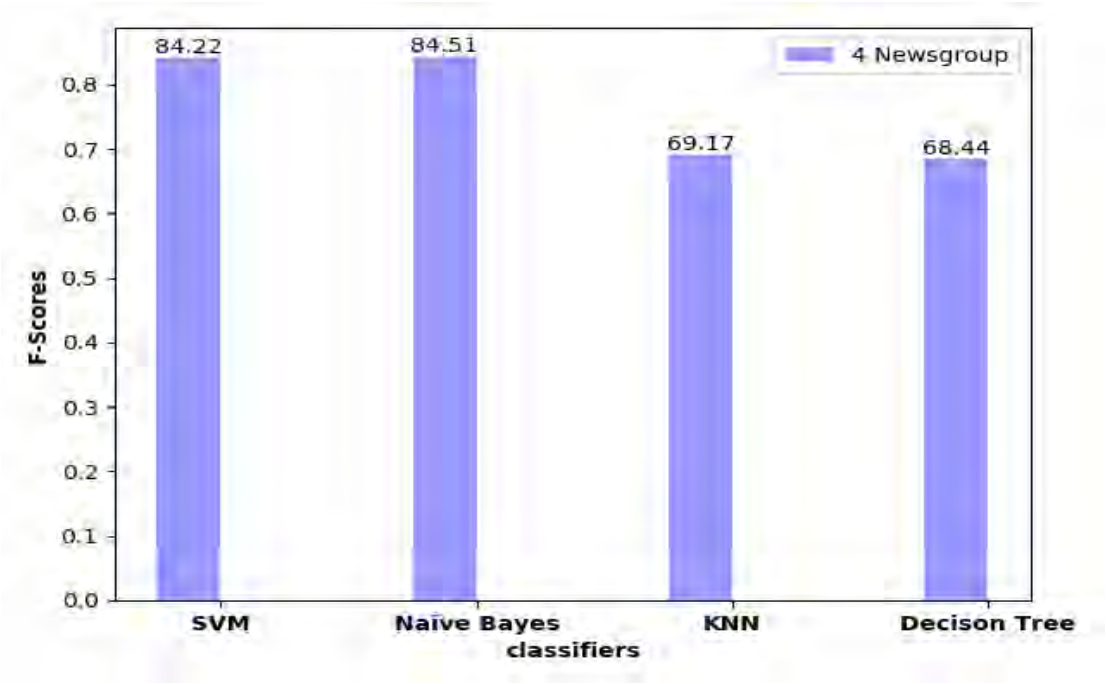

Figure 4.3.7 Comparison of Supervised Algorithms for 4 categories

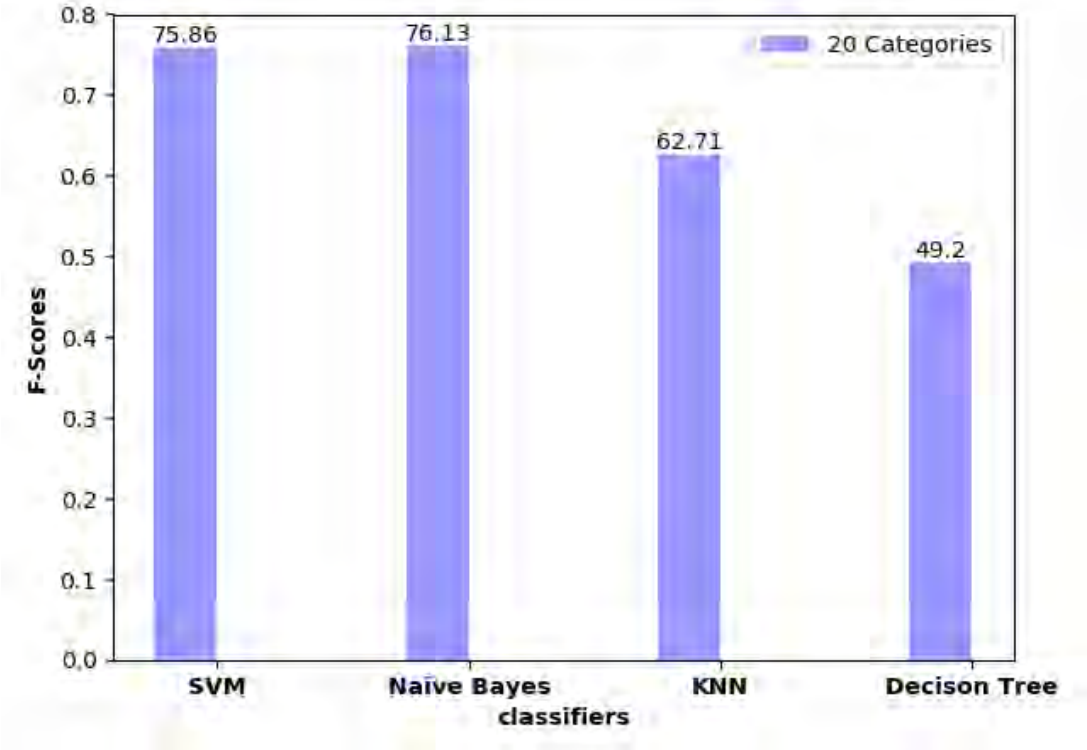

Figure 4.3.8 Comparison of Supervised Algorithms for 20 categories

In figure 4.3.7 and figure 4.3.8 we have compared f1 score of support vector Machine, Naïve Bayes, K-nearest Neighbor and Decision Tree for 20 and 4 categories. We can see that Naïve Bayes gives the best result. SVM is pretty close with Naïve

Bayes. The reason behind such result is that he naive Bayes classifier makes two bold assumptions:

- The probability of occurrence of any word given the class label, is independent of the probability of occurrence of any other word, given that label.
- The probability of occurrence of a word in a document, is independent of the location of that word within the document.

When these two works together it's called a bag-of-words model, each document then is literally just a bunch of words thrown together. Support Vector Machine uses features ranked lowest to determine the class. This method also gives good result. But like Naïve Bayes it doesn't use probability of given class and doesn't consider best features. Moreover, SVM doesn't handle bag of words representation very well. This causes the classifier to be less accurate than Naive Bayes. KNN and Decision tree gave very poor scores because of their incompatibility with the given data. For Decision Tree any small change causes the tree to change the path and misclassify. There are many classes which has close features. This cause Decision Tree to give such poor result than other classifier. K-nearest Neighbor doesn't handle categorical data very well. KNN works based on minimum distance from the query instance to the training samples to determine the k-nearest neighbors and every feature have weights and determining classes using only distance caused misclassifying as there can be many irrelevant features.

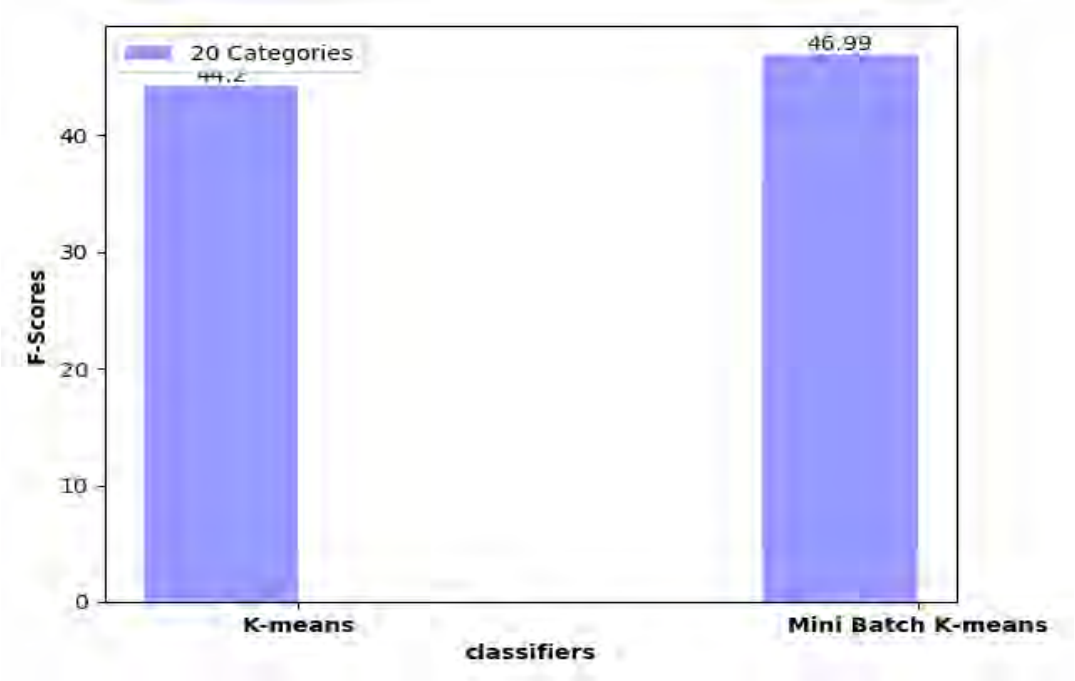

Figure 4.3.9 Comparison of Unsupervised Algorithms for 20 categories

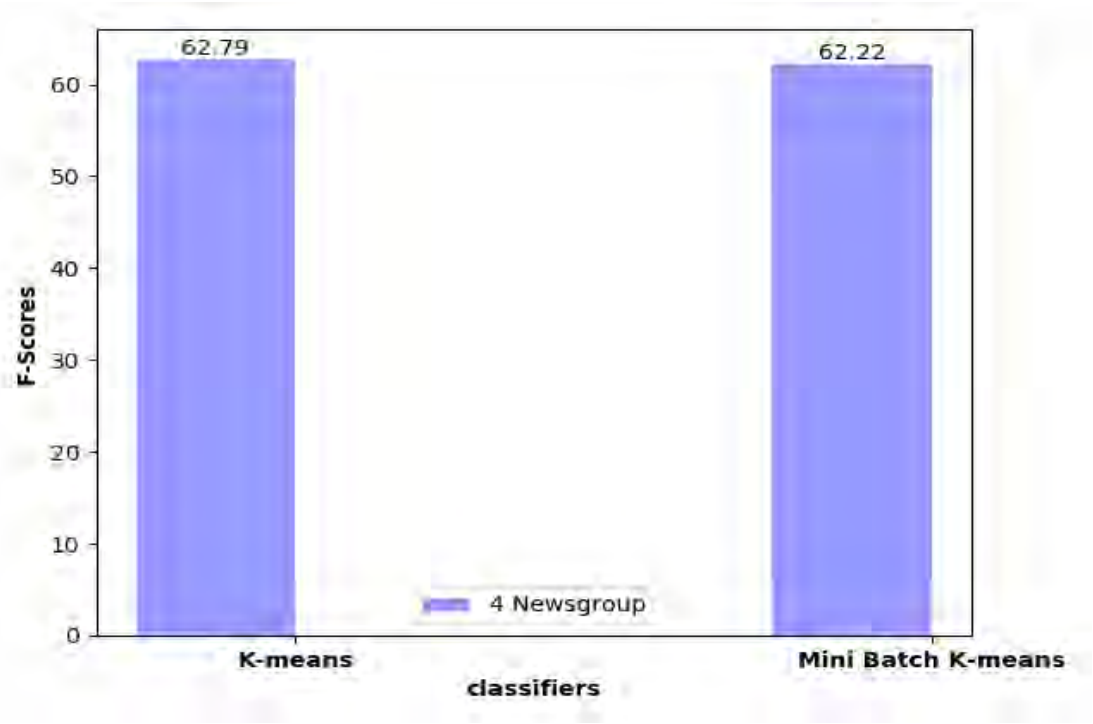

Figure 4.3.10 Comparison of Unsupervised Algorithms for 4 categories

In figure 4.3.9 and figure 4.3.10, we have compared homogeneity scores of K-means clustering algorithm and Mini Batch K-means algorithm for 20 and 4 categories. Here we can see for 20 categories Mini Batch K-means scored better than K-means. This is because Mini Batch handle large amount of data by batching process thus this is faster and give better result than k-means. However, for 4 categories we see K-means and Mini Batch K-means scored almost same because 4 categories consist small amount of data and doesn't necessarily need batching. This result will vary every time we run the classifiers because unsupervised algorithms doesn't cluster the same away every time. Different initial cluster can result in different final clusters.

CHAPTER 5
Conclusions

This chapter consists of a brief summary of this research and demonstrates the future aspects and possibilities. We presented an extensive comparative study of text classification using machine learning algorithms which are KNN, SVM, Naïve Bayes and Decision Tree algorithms. All these algorithms helped us to improve the accuracy to predict the result. The preprocessing is a very important part for the prediction accuracy so we cleaned the dataset with Lemmatizing and stemming algorithms. Due to the digitization of text a lot of data are being stored easily, text classification is the process where we categorize the text and machine learning helps us to make the process superfast, smarter and efficient.

In our study we have found many great works in text categorization. These works helped us to guide us. Here we tried to collaborate different algorithms which fetched only one dataset. This help us to train the data more accurately. But we were unable to get high accuracy for huge amount of data. Low powered pc and less refine data could be one of the issues for getting less accuracy for big amount of data processing. Another concern our unsupervised algorithm is slow because of the time to train the system. We have made the prediction according to a fix dataset so it is very difficult to say that we will get the same amount of accuracy in real life uses. Since the English language sentence structure and use of word is changing very frequently. So we need a live dictionary tool that would make the rules and structure of the present time language. So we propose to make such kind of dictionary with the public accessibility.

5.1 Future Possibilities

Text classification with the help of machine learning system has a lot of opportunities to improve. The world is changing very fast and the way of solving problems are becoming smarter and more efficient. To cope with the need of vast people everything need to be updated. As we are working with text we could make more accurate result by setting the uses of one word on different situation and by focusing on its type. It means how many type of uses of a single word depending on the position of the word

in the sentence. This would help to select which sentence will be on which categories as a result it will help to classify the text. Another thing is to make good accuracy for the large amount of data processing, for this we could use different machine learning algorithms and compare the results between them. In many cases hybrid algorithms output is more accurate and working time is much less.

In our work Decision tree have less accurate result. To overcome this problem our future work will be applying Radom Forest, as it operate by constructing a multitude of Decision trees at running time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. In the future we will be working with semi-supervised learning algorithms. However to get this good amount of accuracy we need to get more generalized text and the text classification will have to be more efficiently used. But the main approach should be towards the semi-supervised learning algorithms as new document can be assigned to more relevant category.

REFERENCES

[1] Margaret Rouse.What is machine learning (ML)? - Definition from WhatIs.com. (2016). Retrieved from https://searchenterpriseai.techtarget.com/definition/machine-learning-ML

[2] Jenny (Xiao) Zhang.10 Machine Learning Terms in Simple English. (2015, August 01). Retrieved from http://jennyxiaozhang.com/10-machine-learning-terms-in-simple-english/

[3] Heller K, Rulik B (2016) Ctenosciara alexanderkoenigi sp. n. (Diptera: Sciaridae), an exotic invader in Germany? Biodiversity Data Journal 4: E6460. https://doi.org/10.3897/BDJ.4.e6460. (n.d.). Empirical Studies on Machine Learning Based Text Classification Algorithms. doi:10.3897/bdj.4.e6460.figure2f

[4] Gore R, Gaikwad S (2015) Checklist of Fabaceae Lindley in Balaghat Ranges of Maharashtra, India. Biodiversity Data Journal 3: E4541. https://doi.org/10.3897/BDJ.3.e4541. (n.d.). doi:10.3897/bdj.3.e4541.figure2f

[5] Payal R. Undhad, Dharmesh J. Bhalodiya Text Classification and Classifiers: A Comparative Study. Retrieved May 3, 2018, from https://www.ijedr.org/papers/IJEDR1702319.pdf

[6] Srinivasan Ramaswamy. (2016). Multiclass Text Classification A Decision Tree based SVM Approach. Retrieved May 12, 2018, from https://pdfs.semanticscholar.org/64d4/3db78cdf2b7eaacd0e949003f410d28a2ee2.pdf

[7] Mike de Waard. Machine Learning for Developers by Mike de Waard. (2018). Retrieved from https://xyclade.github.io/MachineLearning/

[8] BADGERATI. (2010). Machine Learning – Decision Trees. (2018). Retrieved from https://computersciencesource.wordpress.com/2010/01/10/year-2-machine-learning-decision-trees/

[9] Soumojit Sarkar, Nivedita Rajani. (2016). Text Categorization by Unsupervised Learning. Retrieved May 17, 2018, from https://pdfs.semanticscholar.org/6978/3974b7102eef8d9ec1b1cd66a9a0f672733f.pdf

[10] Maini, V. (2017, August 19). Machine Learning for Humans, Part 3: Unsupervised Learning. Retrieved May 25, 2018, from https://medium.com/machine-learning-for-humans/unsupervised-learning-f45587588294

[11] B.H.ChandraShekar, Dr.G.Shoba. (2016). Classification Of Documents Using Kohonen's Self-Organizing Map. Retrieved June 5, 2018, from http://www.ijcte.org/papers/99-G592-613.pdf

[12] Castle, N. (n.d.). What is Semi-Supervised Learning? Retrieved from https://www.datascience.com/blog/what-is-semi-supervised-learning

[13] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien.(2006).Semi-Supervised Learning(Rep.). (n.d.). Retrieved June 6, 2018, from https://www.molgen.mpg.de/3659531/MITPress--SemiSupervised-Learning.pdf

[14] Zhi-Hua Zhou and Ming Li. (2016). Semi-Supervised Regression with Co-Training. Retrieved June 6, 2018, from http://ijcai.org/Proceedings/05/Papers/0689.pdf.

[15] Chapelle, O., Scholkopf, B., & Zien, A. (2010). *Semi-supervised learning*. Cambridge, MA: MIT Press. Retrieved June 6, 2018, from https://www.cs.cmu.edu/~tom/pubs/NigamEtAl-bookChapter.pdf.

[16] S. Jin, W. Chen and J. Han, "Graph-based machine learning algorithm with application in data mining," *2017 Third International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Kolkata, 2017, pp. 269-272. doi: 10.1109/ICRCICN.2017.8234519

[17] Kajaree Das, Rabi Narayan Behera. (2017). Machine Learning for Intelligent Information Access. *Machine Learning and Its Applications Lecture Notes in Computer Science,*274-280. Retrieved June 10, 2018, from http://www.rroij.com/open-access/a-survey-on-machine-learning-conceptalgorithms-and-applications-.pdf.doi:10.1007/3-540-44673-7_15

[18] Dr. Lance Eloit.(2018).Support Vector Machines (SVM) for Al Self-Driving Cars. Retrieved June 12, 2018, from https://aitrends.com/ai-insider/support-vector-machines-svm-ai-self-driving-cars/

[19] Sam Scott, Stan Matwin. Feature Engineering for Text Classification. (n.d.). Retrieved June 15, 2018, from https://pdfs.semanticscholar.org/6e51/8946c59c8c5d005054af319783b3eba128a9.pdf

[20] Adi Bronshtein. (2017). A Quick Introduction to K-Nearest Neighbors Algorithm. Retrieved June 15, 2018, from https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7

[21] Ikonomakis, Emmanouil & Kotsiantis, Sotiris & Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. WSEAS transactions on computers. 4. 966-974.

[22] Simon T Adams, Stephen H Leveson. (2012). Clinical Prediction Rules. Retrieved July 5, 2018, from https://www.bmj.com/content/344/bmj.d8312. (n.d.). doi:10.3897/bdj.1.e1005.

[23] Aurangzeb Khan, Baharum Baharudin, Lam Hong Lee, Khairullah khan. (2016). *A Review of Machine Learning Algorithms for Text-Documents Classification*. Retrieved July 5, 2018, from http://www.jait.us/uploadfile/2014/1223/20141223050800532.pdf. doi:10.18411/d-2016-154

[24] Shweta C. Dharmadhikari, Maya Ingle, Parag Kulkarni.Empirical Studies on Machine Learning Based Text Classification Algorithms. Retrieved July 5, 2018, https://pdfs.semanticscholar.org/22c2/1d58114a6da2d9f520b3e179d32ad1c278ab.pdf. (n.d.). doi:10.3897/bdj.4.e7720.figure2f

[25] Pritam C. Gaigole, L. H. Patil , P.M Chaudhari.Preprocessing Techniques in Text Categorization. Retrieved July 5, 2018, https://pdfs.semanticscholar.org/ff34/7657082e70347a916548a9fe567ab791162a.pdf.(n.d.). doi:10.3897/bdj.4.e7720.figure2f

[26] Margaret Rouse.What is data preprocessing? - Definition from WhatIs.com. (n.d.). Retrieved July 8, 2018, from https://searchsqlserver.techtarget.com/definition/data-preprocessing

[27] N. Elavarasan, Dr. K.Mani. (2016) . A Survey on Feature Extraction Techniques. Retrieved July 9, 2018, from https://pdfs.semanticscholar.org/f1be/6ab3a4b2c036005c9fbe136cd3bb24cf7303.pdf

[28] Nisha Ranjani, Karthikeyan.K. (2016). A SURVEY ON RELEVANCE FEATURE SELECTION METHOD FOR TEXT CLASSIFICATION. Retrieved July 11, 2018, from https://pdfs.semanticscholar.org/8ba0/7e48d0f91d60abdc50b39e2d2c33938847eb.pdf

[29] S. B. Kotsiantis, D. Kanellopoulos, P. E. Pintelas. (2016) .Taxonomic revision of Rochefortia Sw. (Ehretiaceae, Boraginales). Biodiversity Data Journal 4: E7720. https://doi.org/10.3897/BDJ.4.e7720. (n.d.). *Data Preprocessing for Supervised Leaning*. doi:10.3897/bdj.4.e7720.figure2f

[30] Schofield, A., Magnusson, M., & Mimno, D. (2017). Pulling Out the Stops: Rethinking Stopword Removal for Topic Models. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. doi:10.18653/v1/e17-2069

[31] Jason Brownlee. (2017). A Gentle Introduction to the Bag-of-Words Model. (2017, November 21). Retrieved June 8, 2018, from https://machinelearningmastery.com/gentle-introduction-bag-words-model/

[32] S. Lazebnik, A. Torralba, L. Fei-Fei, D. Lowe, C. Szurka. (2010). *Bag-of-Words models*. (n.d.). Lecture presented at Bag-of-Words models in National University of Singapore, Singapore. Retrieved June 10, 2018, from https://cs.nyu.edu/~fergus/teaching/vision_2012/9_BoW.pdf

[33] Ongspxm. (2014). Bag of Words (BoW) - Natural Language Processing. (n.d.). Retrieved from https://ongspxm.github.io/blog/2014/12/bag-of-words-natural-language-processing/