CERTIFICATE

The thesis titled **'TEXT CATEGORIZATION USING TF-IDF VALUE FOR A TEXTUAL DATASET AND PREDICTING THE CLASS LABEL FOR THE DOCUMENTS'** submitted by,

1.	**Sanket Dhabale**	**151080004**
2.	**Pranay Manthanwar**	**151080007**
3.	**Pankaj Dandewad**	**151080056**
4.	**Chetan Ghodam**	**151080057**

is found to be satisfactory and is approved for the Degree of Bachelor of Technology in Information Technology.

Mahesh Shirole
Examiner
Associate Professor, C.S. and I.T. Department
Project Guide

Prof.
External Examiner

Date: 15 / 5 /2019

Place: V.J.T.I., Mumbai

CERTIFICATE

This is to certify that **Sanket Dhabale (151080004), Pranay Manthanwar (151080007), Pankaj Dandewad (151080056), Chetan Ghodam (151080057)** students of **Bachelor of Technology, Information Technology** as a team have completed the report entitled, **"TEXT CATEGORIZATION USING TF-IDF VALUE FOR A TEXTUAL DATASET AND PREDICTING THE CLASS LABEL FOR THE DOCUMENTS"** to our satisfaction**,**

Prof. Mahesh Shirole	**Dr. M M Chandane**
(Asst. Professor, IT Department)	**(Head of Computer Science And IT Department)**
Project Guide	

Dr. Dhiren Patel
Director, VJTI

CERTIFICATE

The report "**TEXT CATEGORIZATION USING TF-IDF VALUE FOR A TEXTUAL DATASET AND PREDICTING THE CLASS LABEL FOR THE DOCUMENTS"** submitted by **Sanket Dhabale, Pranay Manthanwar, Pankaj Dandewad, Chetan Ghodam** is found to be satisfactory and is approved for the Degree of **Bachelor in Technology, Information Technology**.

Prof. Mahesh Shirole
(Associate Professor, IT Department)
Project Guide

External Examiner

Date:

Place:

Acknowledgements

We would like to take this opportunity to express our sincere gratitude towards our mentor and project guide, Associate Professor Mahesh R. Shirole for his constant support and encouragement. We would like to thank him for devoting his time and effort to this project. His words of encouragement and wisdom have kept us on track throughout this project and he has helped us achieve our results.

We would also like to take this opportunity to thank our parents for bearing with us and taking care of us while we set about this journey. They have been the pillars of support that helped us work on this project with dedication and spirit.

Finally, we would like to express our gratitude towards our alma mater, Veermata Jijabai Technological Institute, for the extremely conducive environment it has provided and the infrastructure it has made available to students like us to pursue our projects.

Declaration of the Student

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources.

I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission.

I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature of the students

Sanket Dhabale (151080004)	**Pranay Manthanwar (151080007)**
Pankaj Dandewad (151080056)	**Chetan Ghodam (151080057)**

Date:

Contents

List of Figures

List of Tables

ABSTARCT

Considering the idea of predicting and classify of documents based on various features like the document text data and other metadata associated with the documents, along with generating labels for the list of documents. So we are going to classify the documents accordingly and generate a specified class for the dataset. For that we need to explore the fields of Machine Learning and provide an appropriate data classification model for the project. Most of the dataset is text based so many algorithms like Naïve Bayes, KNN, Random Forest algorithm, etc. would be the perfect algorithm to predict and classify the documents according to the expected and obtained output. Also the end performance can be observed through the confusion matrix and the accuracy with which the classification is done.

CHAPTER 1

INTRODUCTION

This chapter provides the background and motivation for the work carried out in the project .We will study about what led to development of ‘Text Categorization Using Tf-Idf Value For A Textual Dataset And Predicting The Class Label For The Documents’. We will study about how the need for software arose and what the advantage are of it. We will also look at previous attempts in these fields and currently which project is going on. Finally we state the problem statement and also developed possible solution for the same.

Document classification or document categorization is a problem in Library Science, Information science and computer science. The task is to assign a document to one or more Classes or Categories. This may be done "manually" (or "intellectually") or algorithmically. The intellectual classification of documents has mostly been the province of library science, while the algorithmic classification of documents is mainly in information science and computer science. The problems are overlapping, however, and there is therefore interdisciplinary research on document classification.

The documents to be classified may be texts, images, music, etc. Each kind of document possesses its special classification problems. When not otherwise specified, text classification is implied.

Documents may be classified according to their classes or according to other attributes (such as document type, author, printing year etc.). In the rest of this article only classification is considered.

1.1 Background

There are lot of application of text classification in the commercial world. For example, news stories are typically organized by topics; content or products are often tagged by categories; user can be classified into cohorts based on how they talk about a product or brand online.

However, the vast majority of text classification articles and tutorials on the internet or binary text classification such as email spam filtering (spam vs ham), sentimental analysis (positive vs negative). In most cases, our real world problem are much more complicated than that. Therefore this is what we are going to do today classify the document into pre-defined classes.

1.2 Current Scenario

Since the age of internet many of the sports, businesses, advertisement companies, political matters, etc documents are released in different languages and it's quite difficult to predict their documents type or category in which they fall and thus they land up at an unstable position for their expanding their agendas about the information they are providing. Thus our project comes into picture. The project mainly deals with the prediction and classification of the documents based on the text document they had provided along with the dataset. It is more convenient, reliable and also provides the user with wide range of classified and categorized data about the documents.

At first, we have to train our model to classify text data according to their label and provide the test dataset to the model. Thus error in the classification can be calculated. This allows our scope to improve the model.

We are about to use machine learning algorithm that provides appropriate string data classification with maximum accuracy.

1.3 Motivation

We can see the increase in the amount of information available online over various platforms available to us. Also we can see that the number of languages used for text categorization is present in abundance. With abundance of information in various languages, complexity in classification arises. As the text can be in English, Spanish, Italian or any regional languages like Marathi, Hindi. Thus we can't assign a class just comparing it to another language.

Text classification is different from binary classification. Multiple types of documents are present that is assigning a label to a document is not binary but requires much more computation. Not to cause any bias in assigning labels due to multiple classes is the main motivation behind the project. To reduce the unnecessary consumption of resources like time for classification, man handling etc

Classification techniques have been applied to

- Spam Filtering , a process which tries to discern e-mail spam messages from legitimate emails
- Language Identification, automatically determining the language of a text
- Genre Classification, automatically determining the genre of a text.
- Readability Assessment, automatically determining the degree of readability of a text, either to find suitable materials for different age groups or reader types or as part of a larger text simplification system
- Sentiment Analysis, determining the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document.
- Health-Related Classification using social media in public health surveillance.
- Article Triage, selecting articles that are relevant for manual literature curation, for example as is being done as the first step to generate manually curated annotation databases in biology

1.4 Problem Statement

The Problem

Starting from the internet age, many more documents have been flooding over the internet. Lots and lots of documents are found within a minute. Thus assigning a binary class to a document is not preferred and it is not so convenient to do so. Also it is very important to classify these online documents for easy and fast access for computational purposes or finding anything on WWW.

Our proposed system does the job of classifying and labelling a document its accurate class. Along with English text classification and label the documents perfectly. Two separate databases will be required for the algorithm to understand and classify the document accordingly.

The Proposal

Step1. The first step is the Dataset Preparation step which includes the process of loading a dataset and performing basic pre-processing. The dataset is then split into train and validation sets.

The steps involved in the pre-processing are:

1. Tokenize multi-line comments into single sentences
2. Tokenize each sentence into words
3. Remove stop words
4. Calculate the TF-IDF values
5. Lemmatize the words
6. Generate the feature set

Step 2: The pre-processed data to the classifiers to predict the labels/category to which the text belongs.

We have used five classifiers for the project which are

1. Multinomial Naïve Bayes.
2. Random Forest Classifier.
3. K Nearest Neighbour (KNN).
4. Support Vector Machine.
5. Logistic Regression Classifiers.

In the final we compare all the performance of various classifiers using text classification. We observed that each classifier takes up a different approach for classification as mentioned in the DATA and CLASSIFIERS.

CHAPTER 2

LITERATURE REVIEW

In our literature survey regarding the relevant domains and research papers to be focused on for the proposed solution, we mainly focused on research work in the fields of document classification, text classification etc. Our prime focus during the literature survey has been to find out research papers from various sources that can be realized or cited in our solution.

2.1 Introduction

Here we present a few key approaches which we believe may provide different classification techniques applicable across different database in Table no. 2.1 as shown below

Table no. 2.1: Summary of the Literature Review

Sr no	Paper Title	Input type	Model used	Remark
1	Text Categorization with Support Vector Machines: Learning with Many Relevant Features	Text Document	Polynomial and RBF kernels	It uses Support Vector Machine for the text classification. They are fully Automated and eliminate the need of manual parameter turing.
2	Interaction of Feature Selection Methods and Linear Classification Models	Text document	Naïve Bayes, Perceptron-I, Linear SVM	SVM give the more accuracy than the other classifier.

3	An Empirical Comparison of Text Categorization Methods	Message	Latent Semantic Analysis (LSA), SVM, KNN	In these paper tf idf used for generalizing the class. LSA used for reduced dataset. Svm for classification.
4	Integrating Feature and Instance Selection for Text Classification	Text Document	FIS(feature and Instance Selection),Naï ve bayes	Pre-processing the dataset by selecting the feature and instance. It searches for the specific features in a document to classify with majority of features belonging to a classes
5	Pruning Training Corpus to Speedup Text Classification	Text document	KNN	Pruning of dataset by clustering method. Document can be pruned before if it is outlier or with no critical Information about classes
6	Accuracy improvement of automatic text classification based on feature transformation and Multi-classifier combination	Text Data	Euclidean distance, SVM-Linear, Linear discriminant function	The study done in this report tells us that the use of multiple classifiers can be done for better and efficient classification of documents
7	Combining Multiple K-Nearest Neighbour Classifiers for Text Classification by Reducts	Text Document	K- nearest Neighbour, KNN Classifier,RkNN	It uses basic K- nearest neighbour for the classification. Alone K-nearest neighbour is sufficient so multiple feature set has been put to use. It combines multiple KNN classifiers.
8	Feature Selection using Improved Mutual Information for Text Classification	Text Document	Naive Bayes Classifier, Best individual features(BIF), Sequential forward	In text classification, usually a document representation using a bag-of-words approach is employed. This representation scheme leads to very high dimensional feature space

			selection (SFS).	
9	Discretizing Continuous Attributes in AdaBoost for Text Categorization	Text Document	AdaBoost	On the idea of adaptive boosting, a version of boosting in which members of the committee can be sequentially generated after learning from the classification mistakes of previously generated members of the same committee, AdaBoost
10	A comparative study on feature selection in text categorization	Text Document	KNN, Linear Least Square Fit (LLSF), Document frequency, Information gain, Chi-test	Document Frequency (DF) Threshold is the simplest for vocabulary reduction. Easily scale s to very large corpus. the IG or DF combined with KNN or LLSF gives us efficient results for the classification.
11	Text Categorization with Support Vector Machines.	Text Document	SVM	In this we study about (SVM) support vector machines .The SVM are capable of effectively processing feature vectors of some 10 000 dimensions, given that these are sparse.And also support vector machines provide a fast and effective means for learning text classifier's.
12	Text categorization based on Concept indexing and principal component analysis.	Text Document	Concept indexing, principle component analysis, svm, KNN ,Bayesians classifier	They uses the vector space model and feature selection of the text document is represented by a vector and all subsequent calculation based, many ML
13	Improving SVM Text Classification Performance	Text Document	SVM	support vector machines (SVM), when applied to text classification provide excellent precision, but poor

	through Threshold Adjustment			recall.So to improve Recall we customizing SVMs
14	Feature Selection Algorithms to Improve Documents Classification Performance	Text Document	multi-agents systems, feature selection, Information retrieval , text learning	In this we use the feature selection algorithms were evaluated in order to improve documents' classification performance
15	An evaluation of statistical approaches to text categorization."	Text Document	KNN, LLSF ,neural network and WORD, cross method evalution	In these paper is a study of Windrow-Hoff, k-nearest neighbour, neural networks and the Linear Least Squares Fit mapping are the top-performing classifiers, while the Roccio approaches had relatively poor results compared to the other learning methods. KNN is the only learning method that has scaled to the full domain of MEDLINE categories, showing a graceful.
16	Text categorization based on Concept indexing and principal component analysis	Text data	Concept indexing, principle component analysis , KNN, Baysean classifier.	Concept indexing is simple and effective way to reduce dimension. For effective in data compression and feature extraction we use PCA,they applied pca to ci subspace and result into the categorise text data.
17	A Comparison of Word- and Sense-based Text Categorization Using Several Classification Algorithms	Lexical Database	MAP,ML,vers on space,KNN, Recursive Version of the MAP algorithm, Maximum Likelihood	Database and distinction between the word and senses. It contains the large number of noun ,verb etc of English language .WordNet provide carefully worked out word and sense vocabularies for English language, as well as the

			(ML) Classification	membership of each word into a number of senses.the
18	Automatic detection of text genre	Text Data	Corpus logistic Regression , Neural Network	They first linguistic research on genre that uses quantitative method then identify the genres : genetic cues , these cues that have figured prominently in previously work on genre. Then applied method like corpus , logistic Regression , Neural Network. For each genre facet ,it compare our result using surface cues
19	Techniques for Improving the Performance of Naive Bayes for Text Classification	Text document	Rule induction, Naïve bays , decision tree,support vectr machine, clustering	text document data then then classifying these data by the help of naïve bays classifier, in these Bayesian text classification uses a parametric mixture model to model the generation of document. Furthermore, by adding the entropy of (the probability distribution induced by) the document, we account for varying document complexities.
20	Very Large Bayesian Networks in Text Classification	Language text	ETC algorithm, naïve bays classifier, e Chow/Liu algorithm	In these work they used ETC described in details , it constructs a tree-like Bayesian network but contrary to the Chow/Liu algorithm . they reduce the ETC complexity, the popular words should be removed from the dictionary. But in some cases this may deteriorate the accuracy of the classification.

2.2 Data Selection

In the paper title[1] 'Text categorization based on Concept indexing and principal component analysis.' By Thorsten Joachims we observed that With rapid development of Internet, more and more online texts are appearing on the network. While the users of Internet can access more information, they are being swamped by the voluminous information at the same time. Texts on the network are not managed orderly; they are just an unordered set of various data. To search specific information from Internet usually costs much time and energy. If the online texts are well indexed and summarized, people can access these texts more efficiently and effectively. Text categorization is such a solution: it assigns predefined categories to free text documents. Recently much research attention has been paid to text categorization. Most of the researchers based their work on the vector space model (VSM) of text document. It has observed that even for a specified classification method, classification performance of the classifiers based on different training text corpuses are different and in some cases such differences are quite substantias, This observation implies that:

a) Classifier performance relevant to its training corpus in some degree.

b) Good or high quality training corpus may derive classifiers of good performance.

Unfortunately, up to now little research work in the literature has been seen how to exploit training text corpus to improve classifiers performance.

In paper title[2] 'A Comparison of Word- and Sense-based Text Categorization Using Several classification Algorithms' By Janez Brank and Marko Grobelnik , in which we observe that We have compared the relative merit of word- and sense-features for purposes of text classification using the Brown Corpus semantic concordance as benchmark. This comparison has been effected by experimenting with all combinations of: four document representations, six different classification algorithms, various values of the parameters of each algorithm, and ten different data sets. The experimental results have been presented in the form of both tables and diagrams. Our experiments have demonstrated that although in some cases the words result in a slightly better classification than senses, in general there exists a marginal advantage of the senses over the words with respect to classification accuracy. The classification accuracies on the testing data/documents of six algorithms presented in this work is in the range 65-75 % or better. The lower classification accuracy of the two versions of the maximum a posteriori (MAP) algorithm, and the maximum likelihood (ML) algorithm can be attributed to the "Naive Bayes" assumption of the statistical independence of the words. The two variants of the k-nearest neighbour (KNN) algorithm gave better results than the previous algorithms. Nevertheless their performance is limited by the weighted summation they perform which may "smooth out" useful discriminatory features. The overall best results were obtained by the "¾-FLNMAP with Voting" classifier. The good generalization on the testing data/documents of the latter classifier is attributed to the calculation of the largest "uniform boxes" in the training data sets as explained in the text. Note that the computation of the largest uniform boxes in the training data using the technique of maximal expansions is known to improve classification accuracy [18]. Moreover the employment of several "voting ¾-FLNMAP modules" results in a stability and high classification accuracy; this can be attributed to "data noise cancellation" due to the different permutations of the training data used to train different ¾-FLNMAP modules.

We have performed the words/senses comparison assuming complete knowledge of the senses. Nevertheless, in a practical classification task the senses would have to be obtained by a disambiguation step which, in all probability, would introduce a significant error. It seems likely that the 1-2% classification accuracy advantage obtained in the experiments reported here would be more than offset by faulty disambiguation.

In paper Title [4] 'Integrating Feature and Instance Selection for Text Classification' by Dimitris Fragoudis,Dimitris Meretakis, Spiros Likothanasis we observed thatFIS (Feature and Instance Selection) targets both problems simultaneously in the context of text classification of Feature selection where it aims at the reduction of redundant features in a dataset whereas instance selection aims at the reduction of the number of instances. It sequentially selects features that have high precision in predicting the target class. FIS algorithm pre-processes the dataset by selecting the features and instances then dataset is provided to algorithm.

Naïve Bayes, TAN and LB classifier has produced better results with resultant dataset of FIS algorithm. Also provides best result compared to SVM. The results were compared between MI (Mutual Information) and FIS dataset classifier. It was observed that FIS had more promising results than MI.

FIS combines feature and instance selection. The FIS algorithm is easy to implement, very fast, while it greatly decreases the number of features and training instances required to train accurate text classifiers. Training and testing times for text classification are also decreased, in some cases to an order of magnitude.

In paper Title [6] 'Accuracy improvement of automatic text classification based on feature transformation and Multi-classifier combination' by Xuexian Han Guowei Zu, Wataru Ohyama we observed that The normalization to the relative word frequency, the principal component analysis (K-L transformation) and the power transformation were applied to the feature vectors, which were classified by the Euclidean distance, the linear discriminant function, the projection distance, the modified projection distance and the SVM. The classifiers alone are not efficient enough but the working together it overcomes each- others drawback. Based on the score of the document's feature, dataset can be reduced dimensionally if the feature doesn't have the needed count.

The step for dataset selection was feature vector selection, reduction, Feature subset selection and then applying the algorithm to each and every subsets. The algorithms are multi-classifiers ie Combinations of classifiers are used which overcomes individual drawbacks. Among the multi-classifiers used the SVM linear algorithm had the best results among all the algorithms used.

In paper title[7] 'Combining Multiple K-Nearest Neighbour Classifiers for Text Classification by Reducts' by Yongguang Bao and Naohiro Ishii we observed that The basic KNN is the simple method for classification. For text classification it is the best method but for accuracy of KNN many methods are there but not made fully satisfy. Unfortunately, many combining methods such as Bagging, Boosting, or Error Correcting Output Coding, do not improve the kNN classifier at all.

In this paper, we present RKNN, an attempt of combing multiple kNN classifiers using reducts instead of the random feature subsets or the test features. A reduct is the essential part of an information system that can discern all objects discernible by the original information system. Furthermore the multiple reducts can be formulated precisely and in a unified way within the framework of Rough Sets theory. This paper proposes a hybrid technique using Rough Set theory to generate multiple deducts. Then these multiple reducts are used to improve the performance of the k-nearest neighbour classifier

2.3 Classification Method

Jana Novovicova [7] find out 'Feature Selection using Improved Mutual Information for Text Classification' by that we observed A major characteristic of text document classification problem is extremely high dimensionality of text data. In this paper they present two algorithms for feature (word) selection for the purpose of text classification. They used sequential forward selection methods based on improved mutual information introduced by Battiti and Kwak and Choi for non-textual data. These feature evaluation functions take into consideration how features work together.

Filter methods in feature selection are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable.

Top reasons to use feature selection are:

- It enables the machine learning algorithm to train faster.
- It reduces the complexity of a model and makes it easier to interpret.
- It improves the accuracy of a model if the right subset is chosen.
- It reduces overfitting.

By literature paper [10] 'A comparative study on feature selection in text categorization'. This paper is a comparative study of feature selection methods in statistical learning of text categorization. The focus is on aggressive dimensionality reduction. Five methods were evaluated, including term selection based on document frequency (DF), information gain (IG), mutual information (MI), a chi(square) -test (CHI), and term strength (TS). We found IG and CHI most effective in our experiments. Using IG thresholding with a k-nearest neighbor classifier on the Reuters corpus, removal of up to 98% removal of unique terms actually yielded an improved classification accuracy (measured by average precision).

This becomes even more important when the number of features are very large. You need not use every feature at your disposal for creating an algorithm. You can assist your algorithm by feeding in only those features that are really important.

By the literature paper[15] topic ‘An evaluation of statistical approaches to text categorization’ by that paper We observe the comparative study of text categorization methods. Fourteen methods are investigated, based on previously published results and newly obtained results from additional experiments. Problems in previously published experiments are analysed and the results of awed experiments are excluded from the cross method evaluation.

A k-nearest neighbour kNN classifier was chosen for the performance baseline on several collections on each collection, the performance scores of other methods were normalized using the score of kNN. Widrow-Hoff , k nearest neighbour, neural networks and the Linear Least Squares Fit mapping are the top performing classifiers ,while the Rocchio approaches had relatively poor results compared to the other learning methods. KNN is the only learning method that has scaled to the full domain of MEDLINE categories, showing a graceful behaviour when the target space grows from the level of one hundred categories to a level of tens of thousands.

In research paper[17] ‘A Comparison of Word- and Sense-based Text Categorization Using Several classification Algorithms’ in that We have compared the relative merit of word- and sense-features for purposes of text classification using the Brown Corpus semantic concordance as benchmark. This comparison has been effected by experimenting with all combinations of: 1) four document representations, 2) six different classification algorithms, 3) various values of the parameters of each algorithm, and 4) ten different data sets. The experimental results have been presented in the form of both tables and diagrams. Our experiments have demonstrated that although in some cases the words result in a slightly better classification than senses, in general there exists a marginal advantage of the senses over the words with respect to classification accuracy. The classification accuracies on the testing data/documents of six algorithms presented in this work is in the range 65-75 % or better. The lower classification accuracy of 1) the two versions of the maximum a posteriori (MAP) algorithm, and 2) the maximum likelihood (ML) algorithm can be attributed to the “Naive Bayes” assumption of the statistical independence of the words. The two variants of the k-nearest neighbour (KNN) algorithm gave better results than the previous algorithms. Nevertheless their performance is limited by the weighted summation they perform which may “smooth out” useful discriminatory features. The overall best results were obtained by the “¾-FLNMAP with Voting” classifier. The good generalization on the testing data/documents of the latter classifier is attributed to the calculation of the largest “uniform boxes” in the training data sets as explained in the text. Note that the computation of the largest uniform boxes in the training data using the technique of maximal expansions is known to improve classification accuracy [18]. Moreover the employment of several “voting ¾-FLNMAP modules” results in a stability and high classification accuracy; this can be attributed to “data noise cancellation” due to the different permutations of the training data used to train different ¾-FLNMAP modules.

We have performed the words/senses comparison assuming complete knowledge of the senses. Nevertheless, in a practical classification task the senses would have to be obtained by a disambiguation step which, in all probability, would introduce a significant error. It seems likely that the 1-2% classification accuracy advantage obtained in the experiments reported here would be more than offset by faulty disambiguation.

In the literature paper [19] has topic ‘Automatic Detection of Text Genre’. In which the experiments indicate that categorization decisions can be made with reasonable accuracy on the basis of surface cues. All of the facet level assignments are significantly better than a baseline of always choosing the most frequent level (Table 1). and the performance appears even better when one considers that the machines do not actually know what the most frequent level is. When one takes a closer look at the performance of the component machines, it is clear that some facet levels are detected better than others. Table 2 shows that within the facet GENRE, our systems do a particularly good job on REPORTAGE and FICTION. trend correctly but not necessarily significantly for SCITECH and NONFICTION, but perform less well for EDITORIAL and LEGAL texts. We suspect that the indifferent performance in SCITECH and LEGAL texts may simply reflect the fact that these genre levels are fairly infrequent in the Brown corpus and hence in our training set. Table 3 sheds some light on the other cases. The lower performance on the EDITORIAL and NONFICTION tests stems mostly from misclassifying many NONFICTION texts as EDITORIAL. Such confusion suggests that these genre types are closely related to each other, as ill fact they are. Editorials might best be treated in future experiments as a subtype of NONFICTION, perhaps distinguished by separate facets such as OPINION and INSTITUTIONAL AUTHORSHIP.

Although shows that our methods predict BROW at above-baseline levels, further analysis indicates that most of this performance comes from accuracy in deciding whether or not a text is HIGH BROW. The other levels are identified at near baseline performance. This suggests problems with the labelling of the BRow feature in the training data. In particular, we had labelled journalistic texts on the basis of the overall brow of the host publication, a simplification that ignores variation among authors and the practice of printing features from other publications. We plan to improve those labelling in future experiments by classifying brow on an article-by-article basis.

CHAPTER 3

MACHINE LEARNING MODELS

In the following section we discuss some of the most popular classification models used in our system i.e. Multi-Nomial Naïve Bayes(MNB), Random Forest Tree Classifier (RF), Linear SVM, KNN, Logistic Regression (LR).

3.1 Naïve Bayes Classifier

The Naive Bayesian classifier is based on Bayes' theorem with the independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods.

Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c)$, $P(x)$, and $P(x|c)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence. Here the Equation3.1 depicts the probability calculation done through the conditional independencies.

$$\boldsymbol{P(c|x) = \frac{P(x|c)P(c)}{P(x)}} \quad \text{……….(I)}$$

$$\boldsymbol{P(c|X) = P(x_1|c) \times P(x_2|c) \times \ldots\ldots \times P(x_n|c)} \quad \text{……….(II)}$$

Equation3.1 and Equation3.2 Posterior Probability Calculation

- $P(c|x)$ is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*

With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by a multinomial where is the probability that event occurs (or K such multinominals in the multiclass case). A feature vector is then a histogram, with counting the number of times event i was observed in a particular instance. This is the event model typically used for document classification, with events representing the occurrence of a word in a single document (see bag of words assumption). The likelihood of observing a histogram x is given by the Equation3.3 below

$$\boldsymbol{P}(\boldsymbol{x} \mid \boldsymbol{C_k}) = \frac{(\sum_i^n x_i)!}{\prod_i x_i!} \prod_i Pk_i{}^{x_i} \qquad \text{………(III)}$$

If a given class and feature value never occurs together in the training data, then the frequency-based probability estimate will be zero. This is problematic because it will wipe out all information in the other probabilities when they are multiplied. Therefore, it is often desirable to incorporate a small-sample correction, called pseudo count, in all probability estimates such that no probability is ever set to be exactly zero. This way of regularizing naive Bayes is called Laplace smoothing when the pseudo count is one, and Lid stone smoothing in the general case.

3.2 Random Forest Trees

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set $X = x_1, \ldots, x_n$ with responses $Y = y_1, \ldots, y_n$, bagging repeatedly (B times) selects a random sample with replacement of the training set and fits trees to these samples:

For $b = 1, \ldots, B$:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification or regression tree f_b on X_b, Y_b. by taking the majority vote in the case of classification trees.

This bootstrapping procedure leads to better model performance because it decreases the variance of the model, without increasing the bias. This means that while the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated. Simply training many trees on a single training set would give strongly correlated trees (or even the same tree many times, if the training algorithm is deterministic); bootstrap sampling is a way of de-correlating the trees by showing them different training sets.

Additionally, an estimate of the uncertainty of the prediction can be made as the standard deviation of the predictions from all the individual regression trees on x'.

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample. The training and test error tend to level off after some number of trees have been fit.

Random Forest is a great algorithm to train early in the model development process, to see how it performs and it's hard to build a "bad" Random Forest, because of its simplicity. This algorithm is also a great choice, if you need to develop a model in a short period of time. On top of that, it provides a pretty good indicator of the importance it assigns to your features.

Random Forests are also very hard to beat in terms of performance. Of course you can probably always find a model that can perform better, like a neural network, but these usually take much more time in the development. And on top of that, they can handle a lot of different feature types, like binary, categorical and numerical. Overall, Random Forest is a (mostly) fast, simple and flexible tool, although it has its limitations.

3.3 Support Vector Machine

SVM defined as system which uses hypothesis space of linear function in a high dimensional space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory.

SVM can use for both classification and regression of the data. SVM does pretty good job than other classifiers.

For the SVM The length of a vector **x** is called its norm, which is written as ||**x**||. The Euclidean norm formula to calculate the norm of a vector **x** = (x1,x2,...,xnx1,x2,...,xn) is:

$$\| \boldsymbol{x} \| = \sqrt{\boldsymbol{x}_{1+}^2 \boldsymbol{x}_{2+}^2 \dots \dots \boldsymbol{x}_n^2} \qquad \dots\dots(IV)$$

The direction of a vector **x** = (x1, x2) is written as **w**, and is defined as:

$$w=(x1/\|x\|,x2/\|x\|) \qquad \dots\dots.(V)$$

that cos(θ)=x1/‖x‖ and cos(α)=x2/‖x‖. Thus, the direction vector **w** can also be written as:

$$w=(\cos(\theta),\cos(\alpha)) \qquad ....(VI)$$

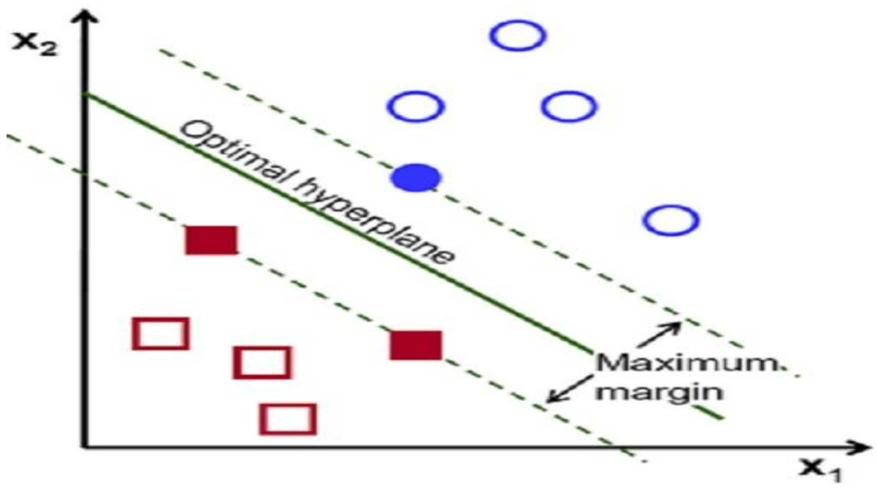

Fig3.1 SVM hyper-planes

by getting each dot product of two vectors we will get the scaler.

$$x\cdot y=\|x\|\ \|y\|\cos(\theta) \qquad \text{.....(VII)}$$

θ is angle between them.

By getting each vector we can classify the vectors in 2D plane but if there is classification needed in 3D then we have to go with hyperplanes.

For the hyper-plane, let's look at the two-dimensional case first. The two-dimensional linearly separable data can be separated by a line. The function of the line is y=ax+b. We rename x with x1 and y with x2 and we get:

$$x1-x2+b=0 \qquad \text{........(VIII)}$$

If we define **x** = (x1,x2) and **w** = (a,−1), we get:

$$w\cdot x + b=0 \qquad \text{..........(IX)}$$

This equation is derived from two-dimensional vectors. But in fact, it also works for any number of dimensions. This is the equation of the hyperplane.

For the classifier of the hyper-plane, once we have the hyperplane, we can then use the hyperplane to make predictions. We define the hypothesis function h as:

$$h(xi)=\{+1 \text{ if } w\cdot x+b\geq 0 \qquad \text{..........(X)}$$
$$-1 \text{ if } w.\ x+b\leq 0$$

The point above or on the hyperplane will be classified as class +1, and the point below the hyperplane will be classified as class -1. So basically, the goal of the SVM learning algorithm is to find a hyperplane which could separate the data accurately. There might be many such hyperplanes. And we need to find the best one, which is often referred as the optimal hyperplane.

3.4 K-Nearest Neighbour

KNN falls in the supervised learning family of algorithms. Informally, this means that we are given a labelled dataset consisting of training observations (x,y) and would like to capture the relationship between x and y. More formally, our goal is to learn a function h:X→Y so that given an unseen observation x, h(x) can confidently predict the corresponding output y.

How does KNN work?

In the classification setting, the K-nearest neighbour algorithm essentially boils down to forming a majority vote between the K most similar instances to a given "unseen" observation. Similarity is defined according to a distance metric between two data points. A popular choice is the Euclidean distance given by

$$(x,x')=\sqrt{}d(x,x')=(x1-x1')^2+(x2-x2')^2+\ldots+(xn-xn')^2 \quad \ldots\ldots\ldots.(XI)$$

but other measures can be more suitable for a given setting and include the Manhattan, Chebyshev and Hamming distance. Other distances are calculated by following formula where x is datapoint from dataset and y

$$\textit{Euclidean Distance} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \ldots\ldots\ldots(XII)$$

$$\textit{Manhattan Distance} = \sum_{i=1}^{n}|x_i - y_i| \quad \ldots\ldots..(XIII)$$

$$\textit{Minkowski Distance} = \left(\sum_{i=1}^{n}(|(x_i - y_i|)^q\right)^{1/q} \quad \ldots\ldots.(XIV)$$

n= Number of Dimensions; x= datapoint from dataset; y=new datapoint(to be predicted)

More formally, given a positive integer K, an unseen observation x and a similarity metric d, KNN classifier performs the following two steps:

- It runs through the whole dataset computing d between x and each training observation. We'll call the K points in the training data that are closest to x the set A. Note that K is usually odd to prevent tie situations.
- It then estimates the conditional probability for each class, that is, the fraction of points in A with that given class label. (Note I(x) is the indicator function which evaluates to 1 when the argument x is true and 0 otherwise)

Finally, our input x gets assigned to the class with the largest probability.

3.5 Logistic regression

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables logistic regression used when the dependent variable(target) is categorical.

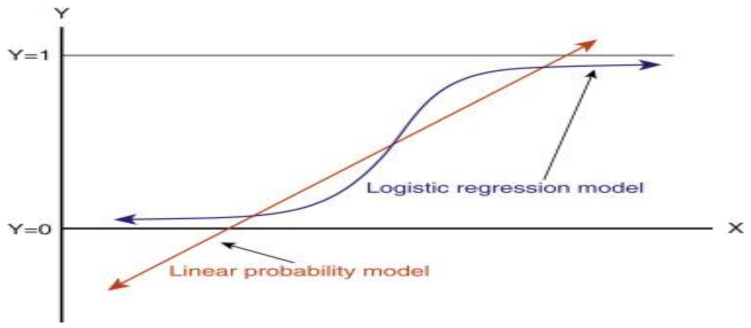

Fig3.2 Logistic curve

The logistic function also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It’s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits. Our dataset has many features and observation.so for each feature

$$\text{Value}= h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_3 x_1^2 + \theta_4 x_2^2) \qquad \text{………..(XV)}$$

We calculate for each feature from its no. of features and no. of observation we made. So here θ will give features and x as observation on text classification.

$$\text{Prior\_prob}=1 / (1 + e\text{^-value}) \qquad \text{……….(XVI)}$$

X1, X2 are the probabilities for values

CHAPTER 4

PROPOSED SYSTEM AND DESIGN

4.1 System Analysis and Design

System analysis and design mainly refers to “the process of studying a procedure or business in order to identify its goals and purposes, to create systems and procedures that will achieve them in an efficient way”.

After identifying the goals and various requirements for the system, we now perform a thorough analysis of these requirements to generate a high level model of the solution. For the proposed solution, we first identify different use cases and components of the system. The primary focus here is to identify and analyse the different use-cases in the system, in order to ensure that the solution meets the demands and constraints of the real – world. We then provide a detailed overview of the solution from the perspective of system design by analysing and formulating the different flows within the system.

4.1.1 Use Case Analysis

Use case analysis is a technique used to identify the requirements of a system (normally associated with software/process design) and the information used to both define processes used and classes (which are a collection of actors and processes) which will be used both in the use case diagram and the overall use case in the development or redesign of a software system or program. The use case analysis is the foundation upon which the system will be built. The Use case diagram for the whole system is as follows:

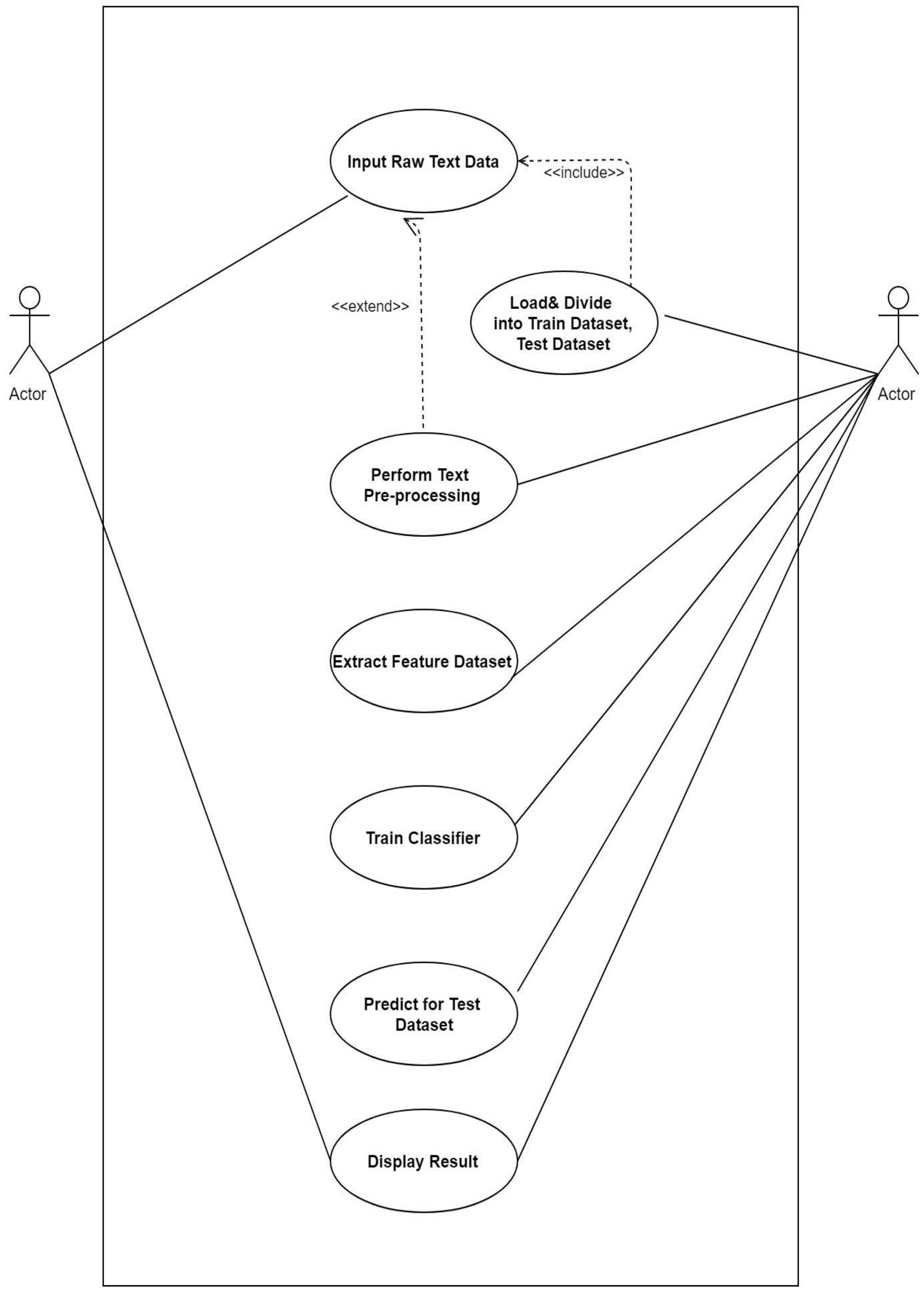

Fig4.1 Use case diagram

Use case 1: Input Raw dataset

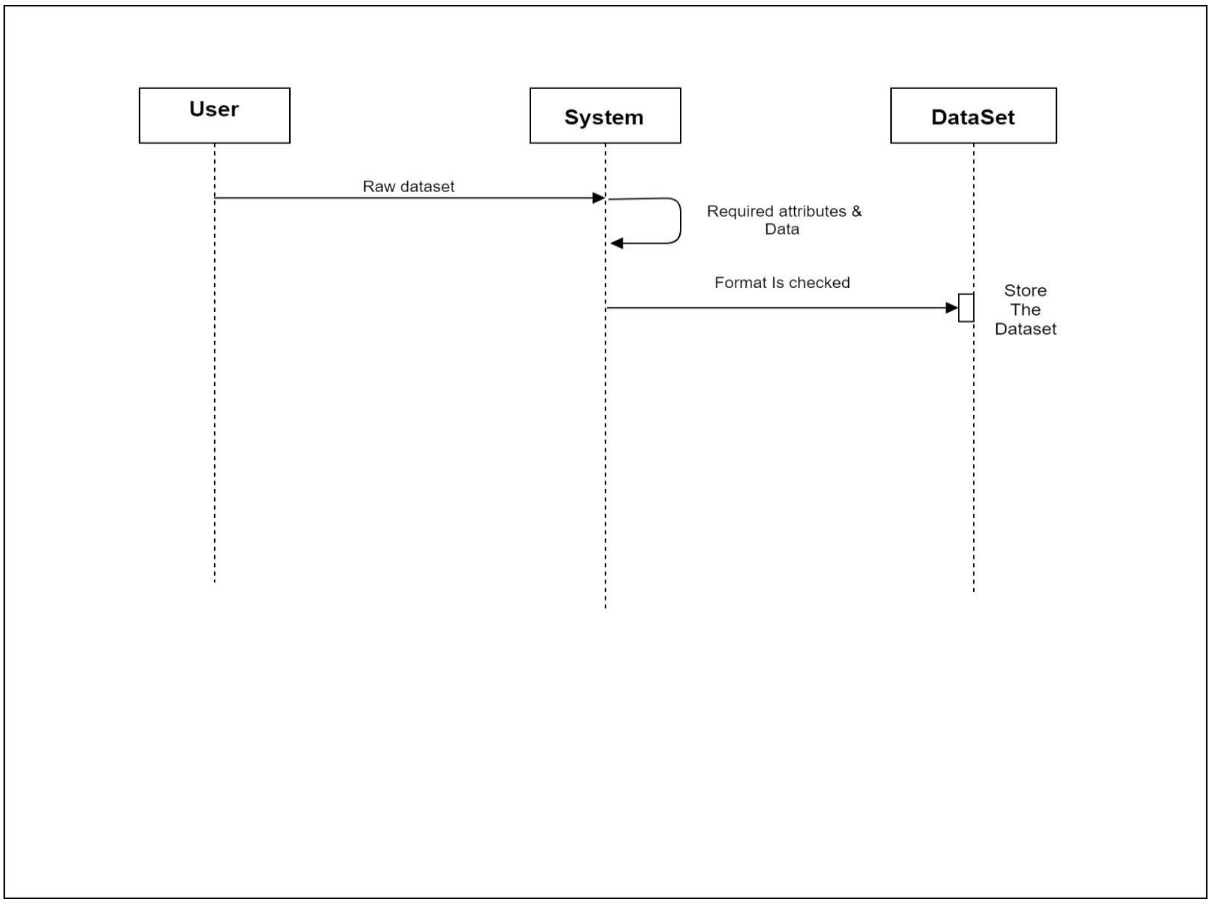

Fig4.2: Sequence diagram of Input Raw dataset

• Goal in context:

The goal of this use case is to take the raw dataset as an input containing mostly the textual data for classification purpose..

• Primary Actors:

User

• Pre-conditions:

1. User should be configured with dataset containing only textual data.
2. Check should be performed before entering data into static dataset.
3. Check should be performed to decide whether data is in required form or not.

• Post-conditions:

1. User must be able to give the raw dataset as an input to the System.

• Basic Flow of Events:

1. Raw Dataset is checked for the required attribute and data format
2. Raw Dataset is forwarded to the System from User
3. If Dataset passes step 1, step 2 then data is Store the dataset.

• Alternative Flows:

1a. Data is not validated.

3a. Dataset doesn’t pass step 1 and/or step 2.

3a1. Dataset exits the System.

Use Case 2: Splitting Dataset Into Train And Test Dataset

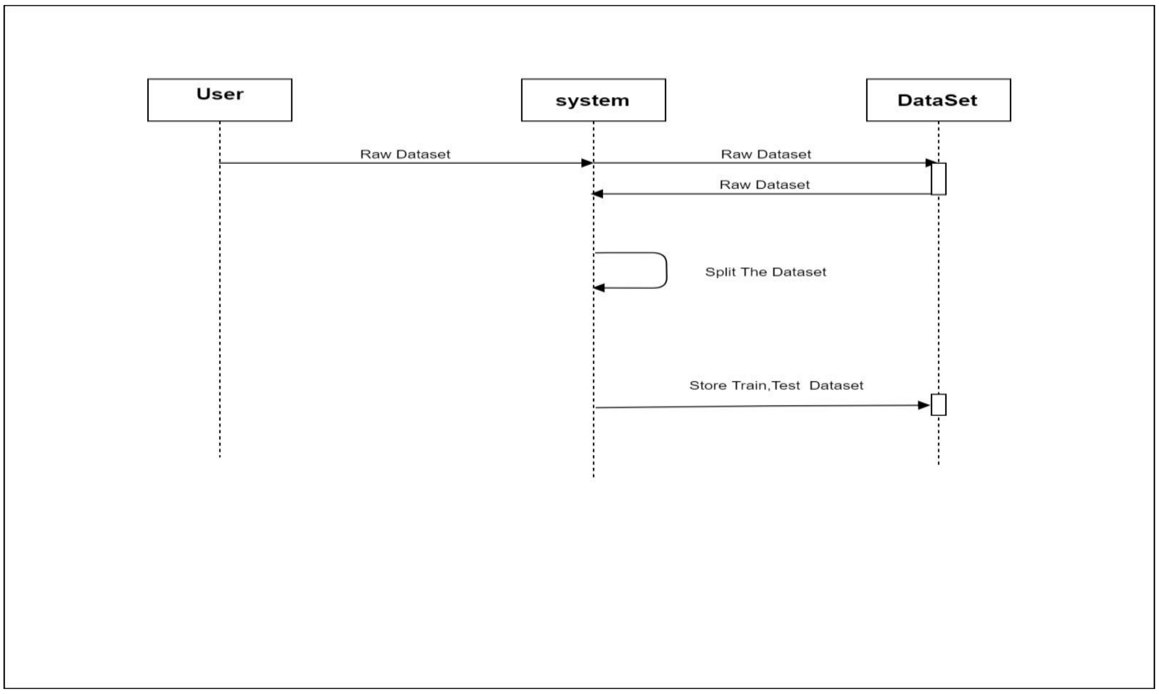

Fig 4.3 Sequence Diagram of Splitting Dataset Into Train And Test Dataset

• Goal in context:

The goal of this use case is to take the raw dataset as an input and split the dataset into the Training and Testing dataset.

• Primary Actors:

System

• Pre-conditions:

1. System should be able to load dataset containing only textual data.
2. Check should be performed before entering and splitting data into static dataset.
3. Check should be performed to decide whether data is in required form or not.
   (Use case: Input Raw Dataset)

• Post-conditions:

1. System will have the Training and Testing Dataset.

• Basic Flow of Events:

1. System will load the raw dataset
2. System will Split the Dataset into Training and Testing dataset.
3. If Dataset passes step1, step2 then store the Training and Testing dataset.

• Alternative Flows:

1a. System is unable to load the dataset.

3a. Dataset doesn’t pass step 1 and/or step 2.

3a1. Dataset exits the System.

Use case 3: Performing Text pre-processing

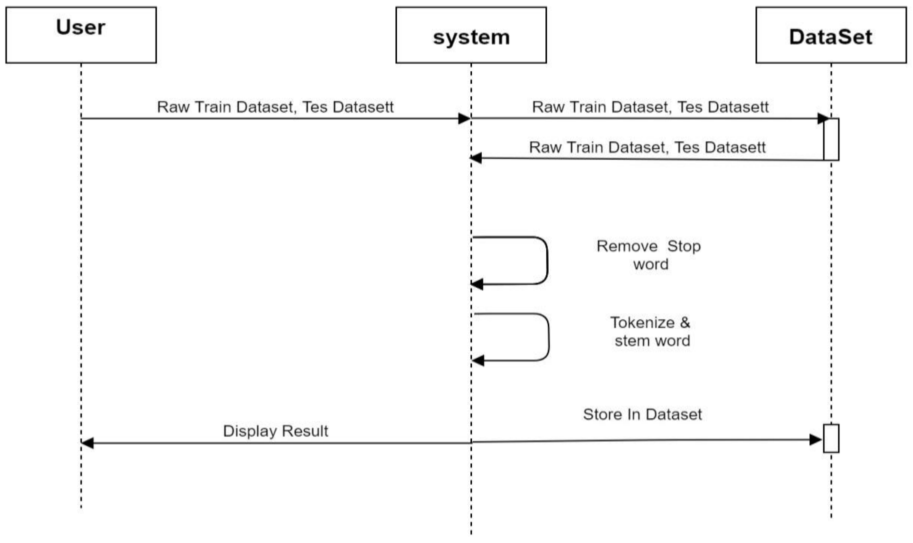
User
system
DataSet
Raw Train Dataset, Tes Datasett
Raw Train Dataset, Tes Datasett
Raw Train Dataset, Tes Datasett
Remove Stop word
Tokenize & stem word
Store In Dataset
Display Result

Fig 4.4 Sequence Diagram of Performing Text Processing

• Goal in context:

The goal of this use case is to validate, pre-process data and ensure that entered into the system is cleansed and validated according to the requirements of the system if the data can be converted into valid form by text pre-processing.

• Primary Actors:

System

• Pre-conditions:

1. System should be configured with datasets containing only textual data.

2. Check should be performed before entering data into static dataset.

3. Check should be performed to decide whether data is in required form or not.

(Use case: Load and Split the Dataset into Training and Testing dataset)

• Post-conditions:

1. System converts Data into valid format for further use.

• Basic Flow of Events:

1. Raw Train and Test datasets are forwarded to the System from User and check if data contains required attributes and data format.

2. Removal of stop-words from the dataset is done by the System.

3. Stemming and Tokenization of each and every word (feature set) is done by the System.

4 If Dataset passes step 2 and step 3 then data is displayed to User

• Alternative Flows:

1a. Data is not validated.

2a. Removal of stop words from raw dataset doesn’t occur.

3a. Tokens of words are not created during Tokenization process.

4a. Dataset doesn’t pass step 2 and/or step 3 and/or step4 and/or step 5.

4a1. Dataset exits the System.

Use case 4: Extract Feature Set using Tf-Idf values.

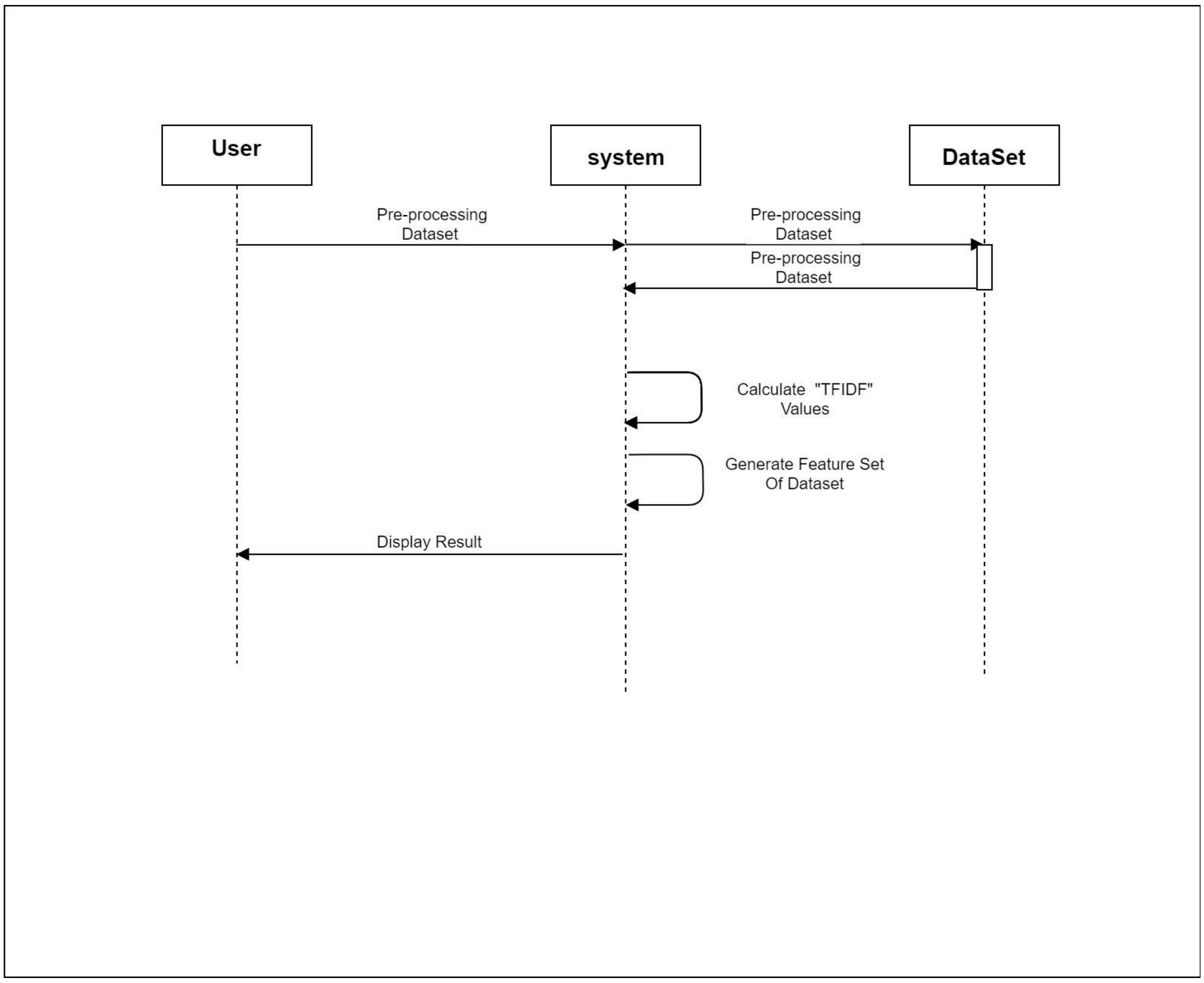

Fig 4.5 Sequence Diagram of Extract Feature Set using Tf-Idf values

• Goal in context:

The goal of this use case is to extract the features from the Train and Test Datasets for classification purpose.

• Primary Actors:

System

• Pre-conditions:

1. Check must be performed over the pre-processed text.

2. Removal of stop-words from the dataset is done by the System.

3. Tokenization of each and every word (feature set) is done by the System.

• Post-conditions:

1. User must have the features and their Tf-Idf values in the New Dataset containing word and its Tf-Idf values.

• Basic Flow of Events:

1. Compute the Tf-Idf values of pre-processed datasets. (**Use case 3**: Performing Text pre-processing)

2. Store the Tf-Idf values and words in feature set.

3. If Dataset passes step 1 and step2 then feature set is displayed to User

• Alternative Flows:

1a. Data is not validated

1a1. Not able to compute the Tf-Idf values.

3a. Dataset doesn't pass step 1 and/or step 2.

3a1. Dataset exits the System

Use case 5: Training Classifiers

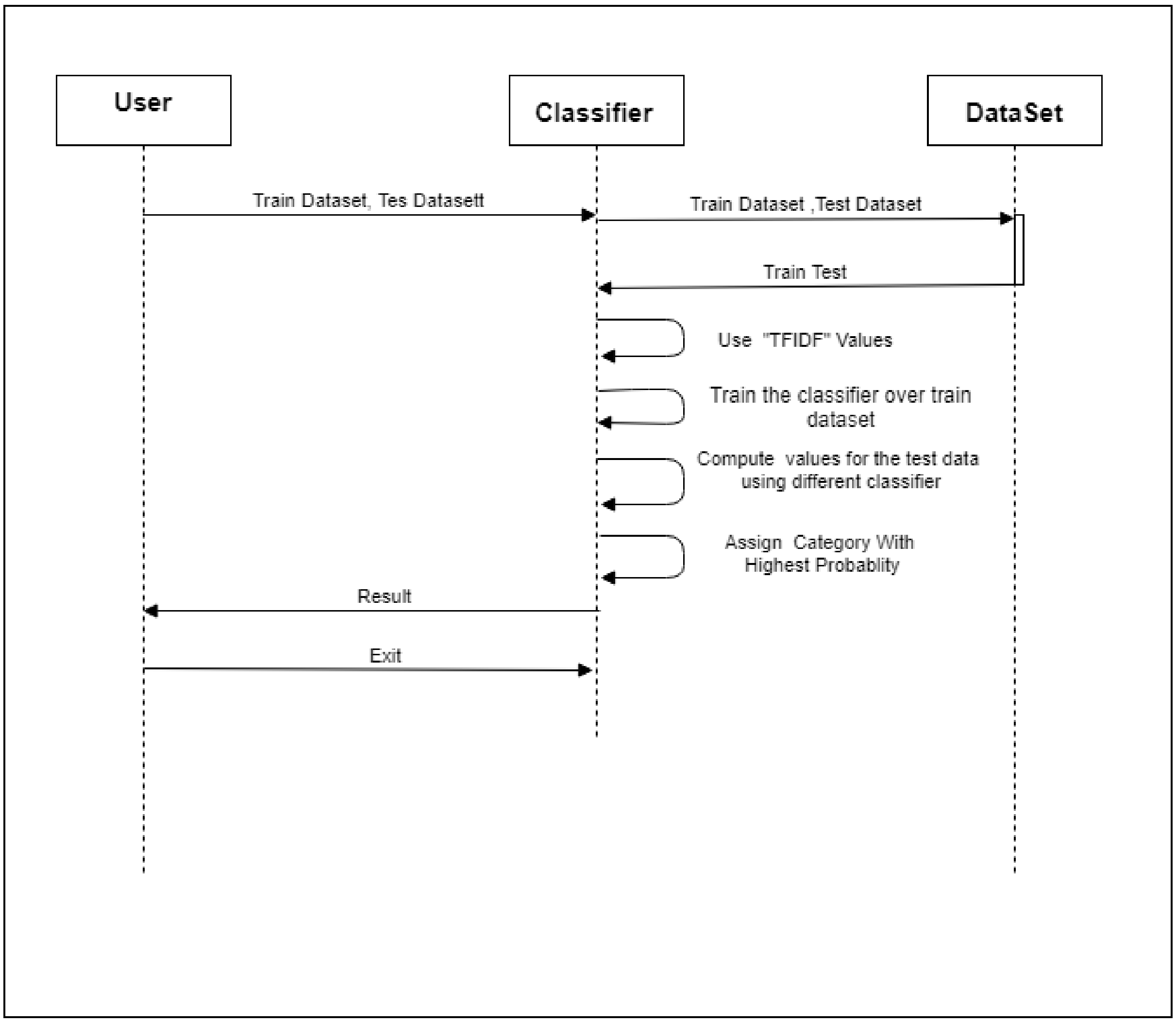
User
Classifier
DataSet
Train Dataset, Tes Datasett
Train Dataset ,Test Dataset
Train Test
Use "TFIDF" Values
Train the classifier over train dataset
Compute values for the test data using different classifier
Assign Category With Highest Probablity
Result
Exit

Fig4.6 Sequence Diagram of Classifier

• Goal in context:

The goal of this use case is to train the classifier using the Train Dataset generated during the Use case 2

• Primary Actors:

System

• Pre-conditions:

1. System should be configured with dataset generated During Use case 1: Text pre-processing of Dataset.
2. System should import the necessary libraries and packages required for the classifier.
3. System must have performed Use Case: Text pre-processing of Dataset

• Post-conditions:

1. System classifies the test dataset with a particular category out of five.

• Basic Flow of Events:

1. Pre-processed Train and Test datasets are forwarded to the System from User for training and testing of the classifier respectively.
2. Use of the Tf-Idf values is done for feature extraction.
3. Compute the values for the test dataset using different Techniques.
4. Assign the category with Highest Value.
5. If the classifier passes step 3, step 4 then Display result to User.

• Alternative Flows:

1a. Pre-processed Datasets are not validated.

3a. System is unable to compute the probability for the categories

4a. System is unable to assign category.

5a. Dataset doesn’t pass step 3and/or step 4.

5a1. Dataset exits the System.

Use case 6: Predict For Test Data

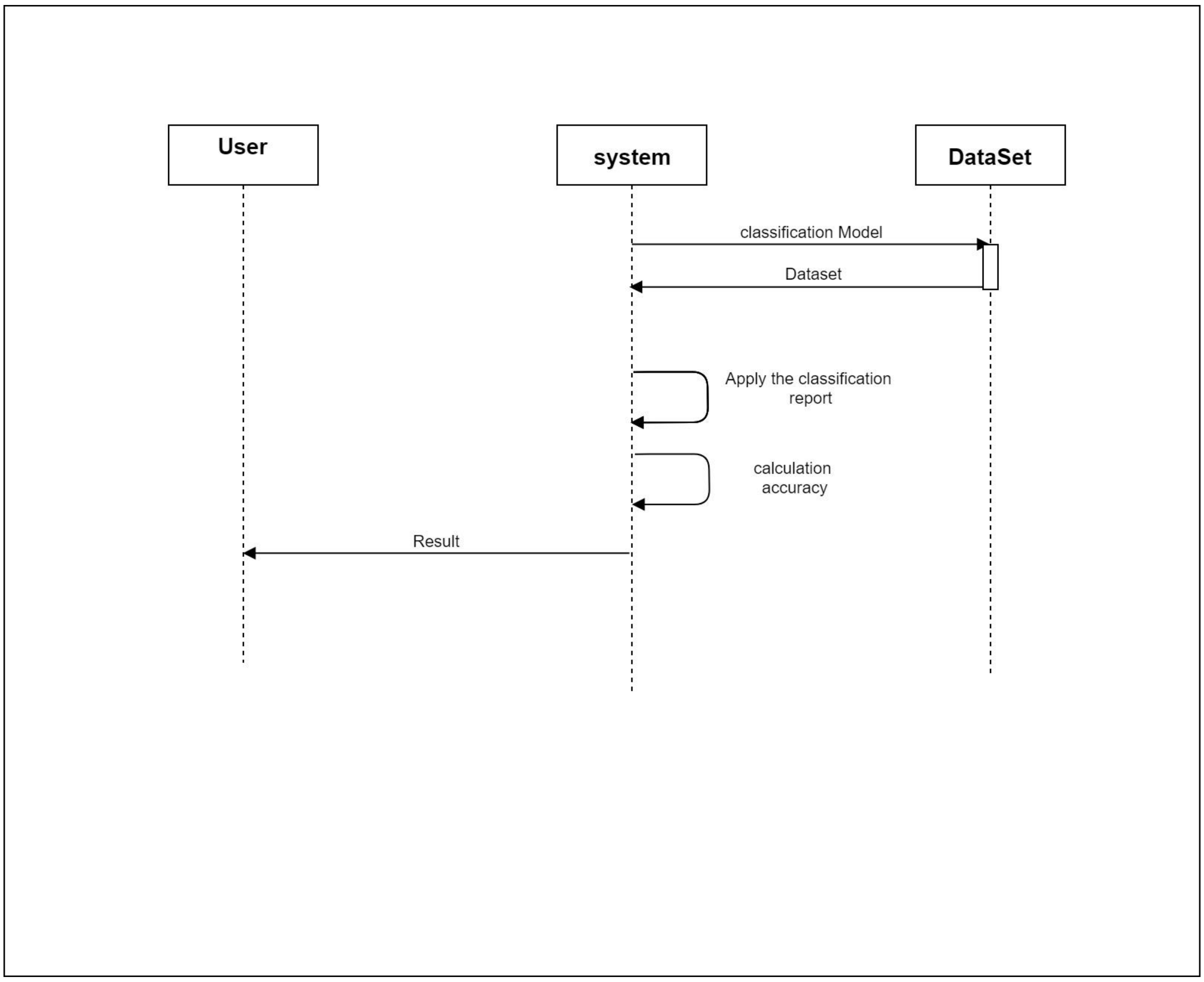

Fig 4.7 Sequence Diagram of predict for test data

• Goal in context:

The goal of this use case is to test the classifier's predicted output vs the actual output.

• Primary Actors:

System

• Pre-conditions:

1. System must be trained over the Train Dataset for classification purpose.
2. System must be done with the Training Phase (Use Case: Training Classifiers)
3. System must have Test Dataset's predicted and actual output.

• Post-conditions:

1. Classifiers performance is compared with every other classifier available.

• Basic Flow of Events:

1. Compute the Correct Number of classifications done. (Use Case: Training Classifiers)

2. Calculate the accuracy for each Classifier.

3. If Dataset passes step 1 and step 2 Display the accuracy to the User.

• Alternative Flows:

1a. Datasets are not validated.

2a. Error in Calculating accuracy values

3a. Dataset doesn't pass step 3and/or step 4.

3a1. Dataset exits the System

Use case 7: Display Result:

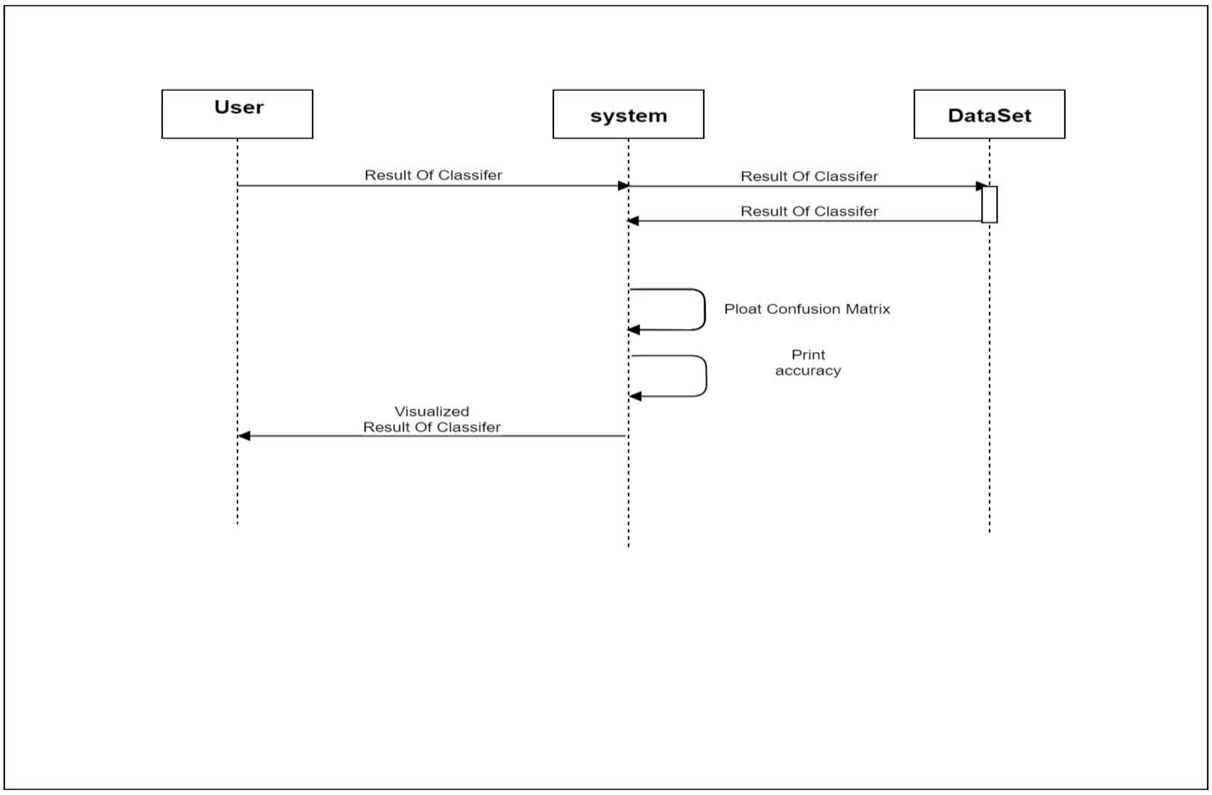

Fig 4.8 Sequence Diagram of Display the result

• Goal in context:

The goal of this use case is to provide a visualized comparison to the User between the classifier based on their accuracy, performance and other metrics using the classification report function and confusion matrix for each and every classifier.

• Primary Actors:

System

• Pre-conditions:

1. System must be done with the testing of Test Dataset. (Use Case: Predict for Test Data)

2. System must have Test Dataset's predicted and actual output for visualization.

• Post-conditions:

1. Confusion Matrix and Performance Measures must be displayed.

• Basic Flow of Events:

1. Plot a confusion matrix using the inbuilt function.

2. Print the accuracy along with the matrix.

3. If Dataset passes step 1 and step 2 then display the Matrix with Classification report.

• Alternative Flows:

1a.Inbuilt functions are not imported

1a1. Proper labelling of matrix is not done

3a. Dataset doesn't pass step 1and/or step 2.

3a1. Dataset exits the System.

4.2 System Design

System analysis and design mainly refers to "the process of studying a procedure or business in order to identify its goals and purposes, to create systems and procedures that will achieve them in an efficient way".

After identifying the goals and various requirements for the system, we now perform a thorough analysis of these requirements to generate a high level model of the solution. For the proposed solution, we will use different types of techniques for classifying the documents in a dataset. Thus taking a stable generated dataset as an input, applying text pre-processing, generating feature sets and feeding the train and test dataset for the classification models for training and testing purposes respectively. After that the results are generated and the visualization of classifiers results is what we opt for in this project.

Based on the system analysis and requirements of the problem statement, we thus formulate the data input data input has two types that are category and Text Document. So here when input will give to system it will goes to train data for its pre-processing. After that pre-processing will takes place various pre-processing actions performed on the text data. After that find out feature selection .Then calculate TF- IDF values. These data performed on the text classification models, to feed the pre-processed data to the classifiers to predict the labels/category to which the text belongs.

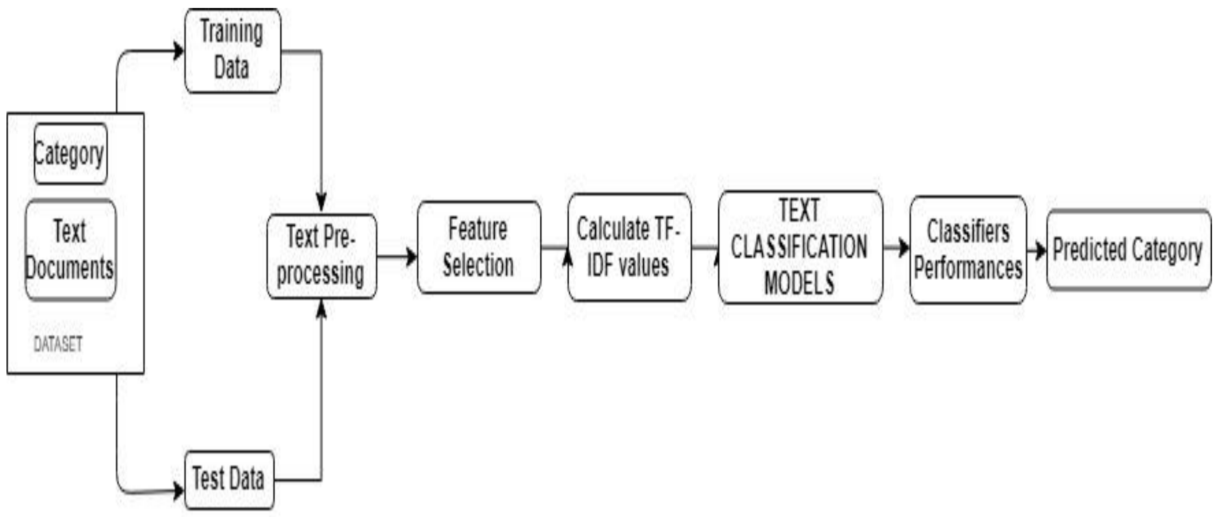

Fig 4.9 System Design

4.2.1 Data

BBC Datasets:

Two news article datasets, originating from BBCNews (https://storage.googleapis.com/dataset-uploader/bbc/bbc-text.csv), provided for use as benchmarks for machine learning research. These datasets are made available for non-commercial and research purposes only, and all data is provided in pre-processed matrix format. If you make use of these datasets please consider citing the publication: D. Greene and P. Cunningham. "Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering", Proc. ICML 2006.

The data was collected from Kaggle Dataset website under the dataset BBC Text Categorization which includes different types of text documents belongs to a certain category. The dataset contains

- Consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005.
- Class Labels: 5 (business, entertainment, politics, sport, tech)

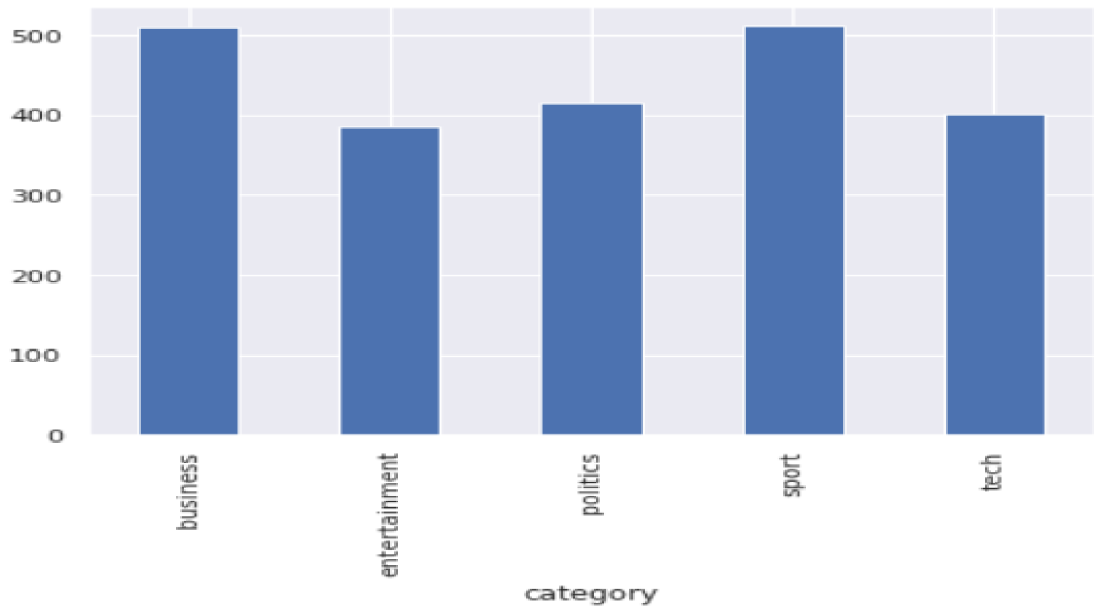

Fig 4.10 Class/Category Count of document used.

4.2.2 Classification

Considering the idea of predicting and classify of documents based on various features like the document text data and other metadata associated with the documents, along with generating labels for the list of documents. So we are going to classify the documents accordingly and generate a specified class for the dataset. For that we need to explore the fields of Machine Learning and provide an appropriate data classification model for the project. Most of the dataset is text based so many algorithms. We have used five classifiers for the project which are Multinomial Naïve Bayes (MNB), Random Forest Classifier (RF), K Nearest Neighbour (KNN), Linear Support Vector Machine (SVM) and Logistic Regression Classifiers (LR). would be the perfect algorithm to predict and classify the documents according to the expected and obtained output. Also the end performance can be observed through the confusion matrix and the accuracy with which the classification is done. Once the model is trained, the test data is given as input to the model which makes predictions. Predictions of the model are compared with the actual labels of the test data. Higher the number of correct predictions, higher is the accuracy.

There are various performance metrics like accuracy, precision, recall, F-score etc. The proposed system is going to be judged on its accuracy, precision and recall. Accuracy is a direct measure of the no. of correctly classified examples. Precision is the measure of result relevancy, while recall is a measure of how many truly relevant results were returned. Excessive false positives deteriorate precision while more of false negatives deteriorate recall.

In the next chapter, we will discuss the implementation of this design. Implementation covers everything from dataset acquisition to analysis of the predictions made by the "Text Categorization Using Tf-Idf Value For A Textual Dataset And Predicting The Class Label For The Documents".

CHAPTER 5

IMPLEMENTATION

In this chapter, we discuss about the implementation of machine learning techniques to distinguish normal connections from attacks using various datasets. First and foremost, we use BBC Text dataset. We use the exact same techniques across all the five classifiers so as to have common grounds for comparison. For dataset, we perform classification, first a multiclass classification for all classes present in the dataset. The data was collected from Kaggle Dataset website under the dataset BBC Text Categorization which includes different types of text documents belongs to a certain category. The dataset consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005.

5.1 DATA PRE-PROCESSING

The first step is the Dataset Preparation step which includes the process of loading a dataset and performing basic pre-processing. The dataset is then split into train and validation sets.

5.1.1 STEPS INVOLVED IN PRE-PROCESSING

Tokenize multi-line comments into single sentences: Make a single sentence of a document and then store it in new document

Remove stop words in the tokenized sentence for a selective language, remove all the stop words

Tokenize each sentence into words: Generates the token for each and every word.

Stemming of words in the tokenized sentence: Put the tokenized value instead of word in a sentence to reduce memory consumption

Calculate the TF-IDF values for the words in the processed data.

Generate the feature set: All the remaining words will be used for generating the feature set for the classifier model. Fig 5.1 depicts the steps involved and the feature extraction from text pre-processing.

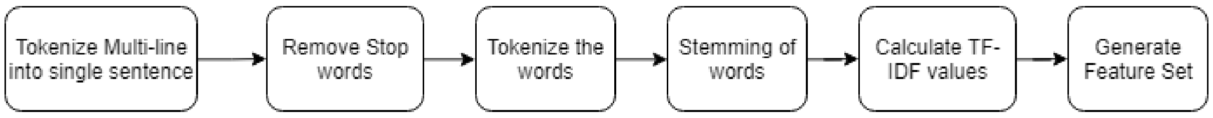

Fig5.1 Steps involved in Text Pre-processing

5.1.2 Feature Selection

Sentiment analysis is one of the most common use cases for classifiers. This kind of analysis is used detect positive or negative sentiment from a user or customer in their comments, tweets, reviews, etc. In language detection, an incoming piece of text will be analysed against a list of languages (English language has been preferred for the project) to programmatically detect the language of the given text. Working with a set of apparel products and you want to automatically classify them using their descriptions. Text classification is often used for organize text by topic. This is commonly used for emails, support tickets, reviews, articles, and other different categories.
We can train a topic classifier to categorize what incoming texts are about, provided that we have provided associations between texts and categories that a machine learning model can learn from.

TFIDF:

The use of TF-IDF stands for (term frequency-inverse document frequency) is discussed in examining the relevance of key-words to documents in corpus. TF-IDF is a combination of two different words i.e. Term Frequency (TF) and Inverse Document Frequency (IDF). TF is used to measure that how many times a term is present in a document. Consider a word "newton" occurs in a document "D1" containing 2000 words exactly 15 times the TF can be calculated as

TF= (No. of word count / No. of words in that document)

= (15/2000) = 0.0075

And considering these values generated through TF, we can't assign weights to the important words. Meaning the influence of particular unique words repeated shortly throughout the document can have the same TF values as the unimportant words like stop-words in the 'of', 'and' etc. which are repeated most of the time in any document. To tackle this IDF is used. The IDF assigns lower weight to frequent words and assigns greater weight for the words that are infrequent. For example a word “newton” occurs in 50 documents and total numbers of documents is 180 then the IDF can be calculated as

IDF= (No. of Document count/ No of documents word is present)
= log_e(180/50) = 0.556

Thus IDF ensures that the weights assign are higher for infrequent words and lower for frequent words. The greater or higher occurrence of a word in documents will give higher term frequency and the less occurrence of word in documents will yield higher importance (IDF) for that keyword searched in particular document.

The TF-IDF value is the multiplicative product of the TF and IDF values for a word. The TF-IDF value for word “newton” is calculated as TF-IDF= 0.0075*0.556 = 0.00417

5.2 Classifiers

Our aim is to feed the pre-processed data to the classifiers to predict the labels/category to which the text belongs.

We have used five classifiers for the project which are Multi-Nomial Naïve Bayes (MNB), Random Forest Tree Classifier (RF), Linear SVM, KNN, Logistic Regression (LR).

- The manipulation of parameters of data transformation and classification in the pipeline can improve the accuracy. To reach out the best accuracy I tried, these parameters are used
- Creates a “Pipeline” of data transformation and classification
- Uses Classifiers to classify the Movie data into Positive and Negative Sentiment

5.2.1 Naïve Bayes Classifier

Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c)$, $P(x)$, and $P(x|c)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence.

- We have a number of documents, each of which has a number of words.
- We would like to classify documents into categories.
- The feature vector consists of all possible words in all documents, and has values of number of counts in each document.

P(word|class)=
(word_count_in_class + 1) / (total_words_in_class + total_unique_words_in_all_classes)

word_count_in_class : sum of(tf-idf values of the word for all the documents belonging to that class)

//basically replacing the counts with the tfidf weights of the same word calculated for every document within that class.

Thus class with highest posterior probability will be assigned as the label to that text document.

5.2.2 Random Forest Tree

One of the main feature of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random, by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

Given the nature of **random forests** (a *bagging decision tree*), it is true that you may come up with a rather weak classifier, especially if only a couple of features are truly significant to determine the outcome. The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set $X = x_1, ..., x_n$ with responses $Y = y_1, ..., y_n$, bagging repeatedly (B times) selects a random sample with replacement of the training set and fits trees to these samples:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification or regression tree f_b on X_b, Y_b.

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x':

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample. The training and test error tend to level off after some numbers of trees have been fit.

5.2.3 Support Vector Machine

Support Vector Machines (SVM) is gaining popularity as an efficient data classification algorithm and is being widely used in many machine vision applications due to its good data generalization performance. The linear SVM does not benefit from feature selection but is much less sensitive to the reduction of the feature space. Here one can reduce the feature set to the point where training documents are represented by 10 terms on average, while still losing only a few percentage in terms of the F1 performance measure.

For the SVM The length of a vector **x** is called its norm, which is written as ||**x**||. The Euclidean norm formula to calculate the norm of a vector **x** done in equation IV.

In this method, for k class problem, k*(k-1)/2 binary classifiers are needed, which is much higher than one-vs-All method. For example, if we have 20 classes, we need to build {20*19}{2}=190 binary classifiers. hence, increasing the number of classes may lead to cumbersome. However, in this method the problem of unbalanced dataset is often smaller, hence the number of samples which are needed to train the classifier become smaller and actually the training and tuning of the classifier become faster and more accurate. To determine the final class for this approach, a simple voting scheme can be used.

For the classifier of the hyper-plane, once we have the hyperplane, we can then use the hyperplane to make predictions. We define the hypothesis function h as in equation X.

The point above or on the hyperplane will be classified as class +1, and the point below the hyperplane will be classified as class -1. So basically, the goal of the SVM learning algorithm is to find a hyperplane which could separate the data accurately.

5.2.4 K-Nearest Neighbours

The KNN based text classification approach is quite simple: given a test document, the system finds the k nearest neighbours among training documents in the training corpus, and uses the classes of the k nearest neighbours to weight class candidates. The similarity score of each nearest neighbour document to the test document is used as the weight of the classes of the neighbour document. If several of k nearest neighbours shares a class, then the per-neighbour weights of that class are added together, and the resulting weighted sum is used as the likelihood score of that class with respect to the test document. By sorting the scores of candidate classes, a ranked list is obtained for the test document. By thresholding on these scores, binary class assignments are obtained.

In the classification setting, the K-nearest neighbour algorithm essentially boils down to forming a majority vote between the K most similar instances to a given "unseen" observation. Similarity is defined according to a distance metric between two data points. A popular choice is the Euclidean distance given by

Finally, our input x gets assigned to the class with the largest probability.

5.2.5 Logistic Regression

Logistic regression is a simple and easy to understand classification algorithm, and Logistic regression can be easily generalized to multiple classes. Logistic Regression requires significantly more time to be trained comparing to Naive Bayes, because it uses an iterative algorithm to estimate the parameters of the model. Consider bag of words representation for the text and occurrence of a word is denoted by 1 and absence by 0. We are trying to estimate coefficient theta for every word and category combination.

The logistic function also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits. Our dataset has many features and observation.so for each feature

We calculate for each feature from its no. of features and no. of observation we made. So here θ will give features and x as observation on text classification.

The probability score is calculated for all the classes i.e. 1/(1+e^-value). New document is classified based on highest probability.

CHAPTER 6

RESULTS

In the experiment, we compared the performances of the various classifiers used for the text classification. We observed that each classifier takes up a different approach for classification as mentioned in the DATA and CLASSIFIERS. The main aim of the experiment is to compare the different performance of the classifier used. Multinomial Naïve Bayes uses the probabilities; Random forest uses entropy; SVM uses hyper-planes; KNN uses the similarity score for each class and the logistic regression uses the probability score.

Table 6.1: Accuracy Measurements of all the classifier

Classifiers	**Accuracy**
Multinomial Naïve Bayes (MNB)	0.9386
Random Forest Classifier (RF)	0.9446
Support Vector Machine (SVM)	0.9746
K Nearest Neighbour (KNN)	0.9281
Logistic Regression Classifiers (LR)	0.9686
Average Accuracy of Experiment	**0.9509**

In Table 6.1 Depicts the accuracy of the MNB is 0.9386, RF is 0.9446, SVM is 9746, KNN is 0.9281 , LR is 0.9686 and average accuracy of the experiment is 0.9509.

We have used five classifiers for the project which are Multinomial Naïve Bayes, Random Forest Classifier, K Nearest Neighbour (KNN), Support Vector Machine and Logistic Regression Classifiers.

The experiment was perform and the precision and recalls of all the classifiers and combined them for our all over result of the experiment. So result of each classifier can be seen one by one as below.

1. Naïve Bayes

Table no. 6.2: Accuracy using MNB is 0.9386

	Precision	Recall	F1-score	Support
Business	0.94	0.94	0.94	157
Entertainment	0.99	0.85	0.91	112
Politics	0.84	0.99	0.91	109
Sport	0.94	0.99	0.97	152
Tech	0.98	0.91	0.94	138
Micro avg	0.94	0.94	0.94	668
Macro avg	0.94	0.94	0.94	668
Weighted avg	0.94	0.94	0.94	668

Table 6.2 show the accuracy of the naïve bayes classifier is 0.9386. I n these table we got the micro average is 0.94 and macro average is also 0.94 with the support of 668 test documents. The weighted average of Naïve Bayes classifier is 0.94

2. Random Forest Algorithm

Table no.6.3: Accuracy using Random Forest is 0.9446.

	Precision	Recall	F1-score	Support
Business	0.91	0.96	0.93	157
Entertainment	0.97	0.90	0.94	112
Politics	0.93	0.94	0.94	109
Sport	0.96	0.99	0.97	152
Tech	0.97	0.91	0.94	138
Micro avg	0.94	0.94	0.94	668
Macro avg	0.94	0.94	0.94	668
Weighted avg	0.94	0.94	0.94	668

Table 6.3 show the accuracy of the Random forest algorithm is 0.9446. In these table we got the micro average is 0.94 and macro average is also 0.94 with the support of test documents. The weighted average of Random Forest classifier is 0.94 .

3. Support Vector Machine

Table no. 6.4: Accuracy using Linear SVM is 0.9746

	Precision	Recall	F1-score	Support
Business	0.96	0.97	0.97	157
Entertainment	0.98	0.96	0.97	112
Politics	0.96	0.97	0.97	109
Sport	0.98	0.99	0.99	152
Tech	0.99	0.96	0.98	138
Micro avg	0.97	0.97	0.97	668
Macro avg	0.97	0.97	0.97	668
Weighted avg	0.97	0.97	0.97	668

Table 6.4 show the accuracy of the support vector machine is 0.9746. I n these table we got the micro average is 0.97 and macro average is also 0.97 with the support of 668 test documents. The weighted average of SVM classifier is 0.97.

4. K Nearest Neighbour

Table no. 6.5: Accuracy using KNN is 0.9281

	Precision	Recall	F1-score	Support
Business	0.94	0.83	0.89	157
Entertainment	0.95	0.92	0.94	112
Politics	0.81	0.97	0.88	109
Sport	0.99	0.99	0.99	152
Tech	0.94	0.94	0.94	138
Micro avg	0.93	0.93	0.93	668
Macro avg	0.93	0.93	0.93	668
Weighted avg	0.93	0.93	0.93	668

Table 6.5 show the accuracy of the KNN is 0.9281. I n these table we got the micro average is 0.93 and macro average is also 0.93 with the support of 668 test documents. The weighted average of KNN classifier is 0.93

5. Logistic Regression

Table no. 6.6: Accuracy using Logistic Regression is 0.9686

	Precision	Recall	F1-score	Support
Business	0.95	0.96	0.95	157
Entertainment	0.99	0.96	0.97	112
Politics	0.95	0.97	0.96	109
Sport	0.99	0.99	0.99	152
Tech	0.98	0.97	0.97	138
Micro avg	0.97	0.97	0.97	668
Macro avg	0.97	0.97	0.97	668
Weighted avg	0.97	0.97	0.97	668

Table 6.6 show the accuracy of the Logistic Regression is 0.9686. I n these table we got the micro average is 0.97 and macro average is also 0.97 with support of 668 test documents. The weighted average of Logistic Regression classifier is 0.97.

Confusion Matrix:

The multi-label confusion matrix is an object that contains the prediction, the expected values and also a lot of pre-processed information related with these data. The below confusion matrixes are represented for better visualization purpose using the *Seaborn.* Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. It plots rectangular data as a colour-encoded matrix.

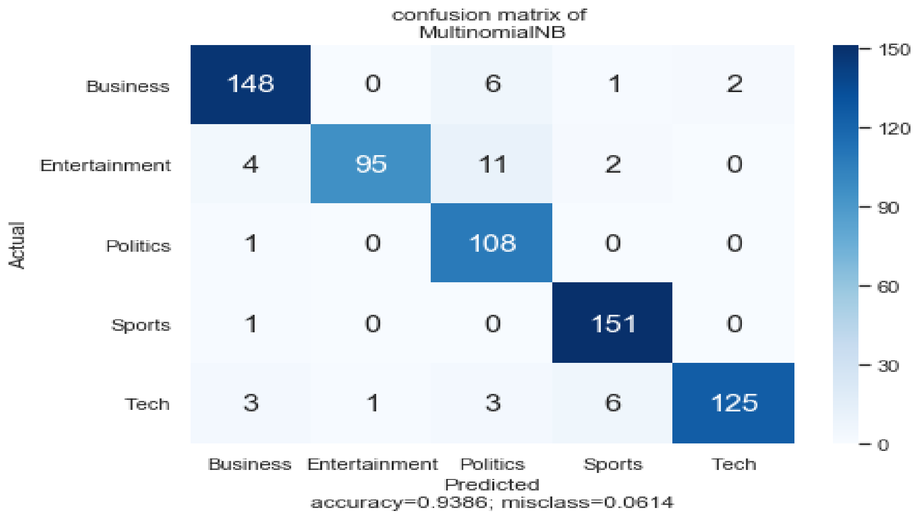

Fig6.1 Confusion Matrix of Multi-Nomial Naïve Bayes

In Fig 6.1 show that confusion matrix of Multi –normal Naïve Bayes classifier in that correctly classified the Business class is 148 , entertainment class is 95 , politics class is 108, sport class is 151 and tech class is 125 no of actual and correctly classified document correctly matched. . In that accuracy of correctly classified class is 0.9386 and misclass is 0.0614. In that sport class having highest no of actual and predicted no of document match and economics class having lowest no of matching document.

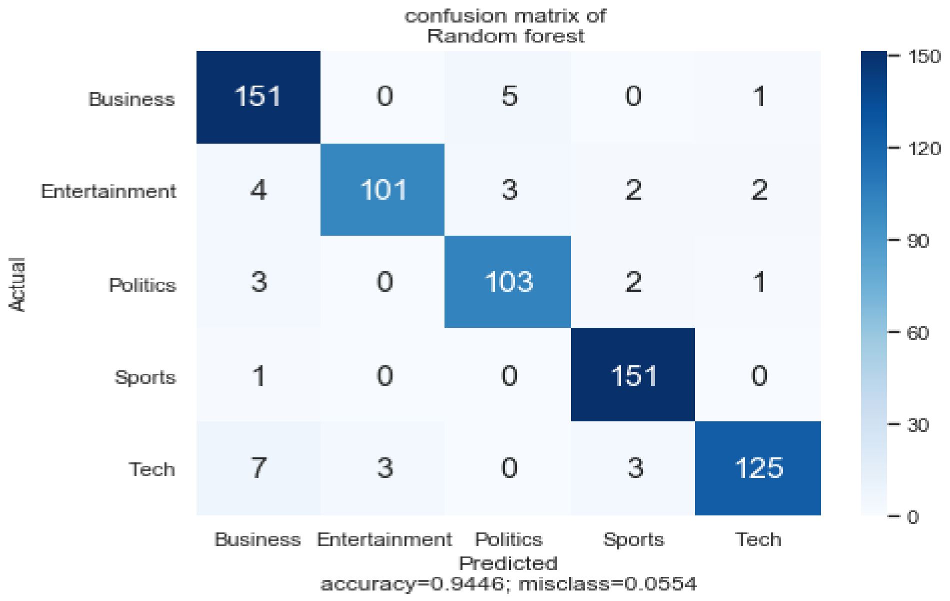

Fig.6.2 Confusion Matrix of Random Forest Tree

In Fig 6.2 show that confusion matrix of Random forest tree classifier in that correctly classified the business class is 151, entertainment class is 101 , politics class is 103, sport class is 151 and tech class is 125 no of actual and correctly classified document correctly matched. . In that accuracy of correctly classified class is 0.9446 and misclass is 0.0554. In that sport class and business class having highest no of actual and predicted no of document matched and economics class having lowest no of document matched.

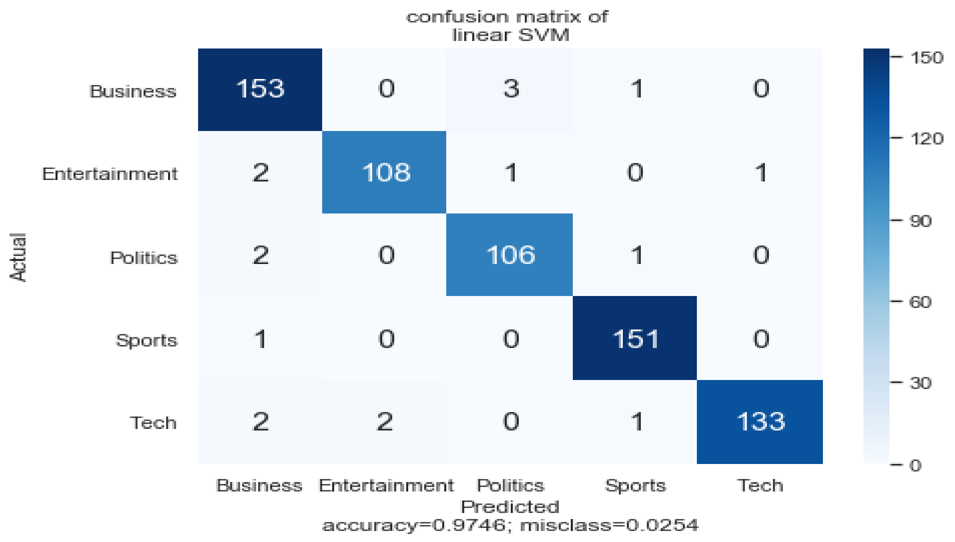

Fig6.3 Confusion Matrix of Linear SVM

In Fig 6.3 show that confusion matrix of linear SVM in that correctly classified the business class is 153 , entertainment class is 108 , politics class is 106, sport class is 151 and tech class is 133 no of actual and correctly classified document correctly matched. . In that accuracy of correctly classified class is 0.9746 and misclass is 0.0254. In that business class has the highest no of actual and predicted no of document match and politics class has the lowest no of matching document.

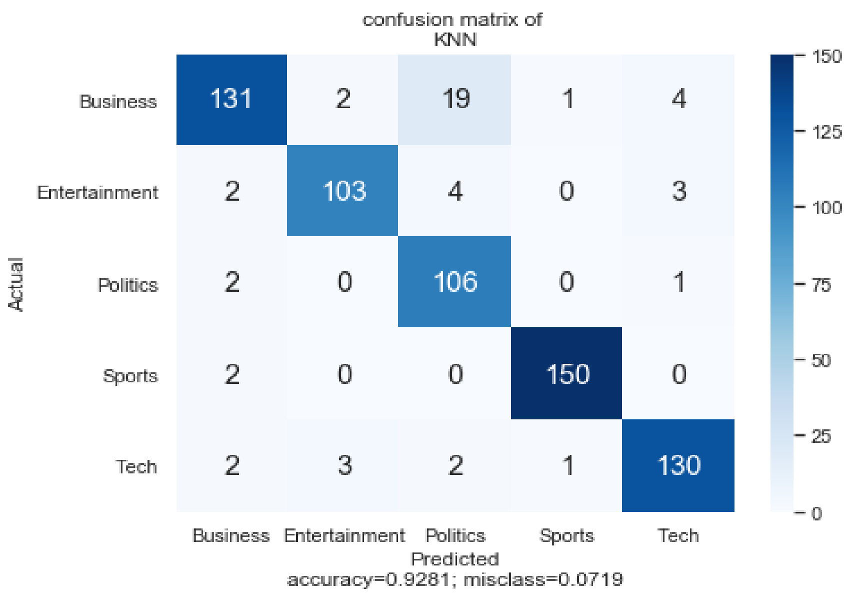

Fig6.4 Confusion Matrix of KNN

In Fig 6.4 show that confusion matrix of KNN classifier in that correctly classified the business class is 131 , entertainment class is 103 , politics class is 106, sport class is 150 and tech class is 130 no of actual and correctly classified document correctly matched. . In that accuracy of correctly classified class is 0.9281 and misclass is 0.0719. In that sport class has the highest no of actual and predicted no of document match and economics class has the lowest no of match document.

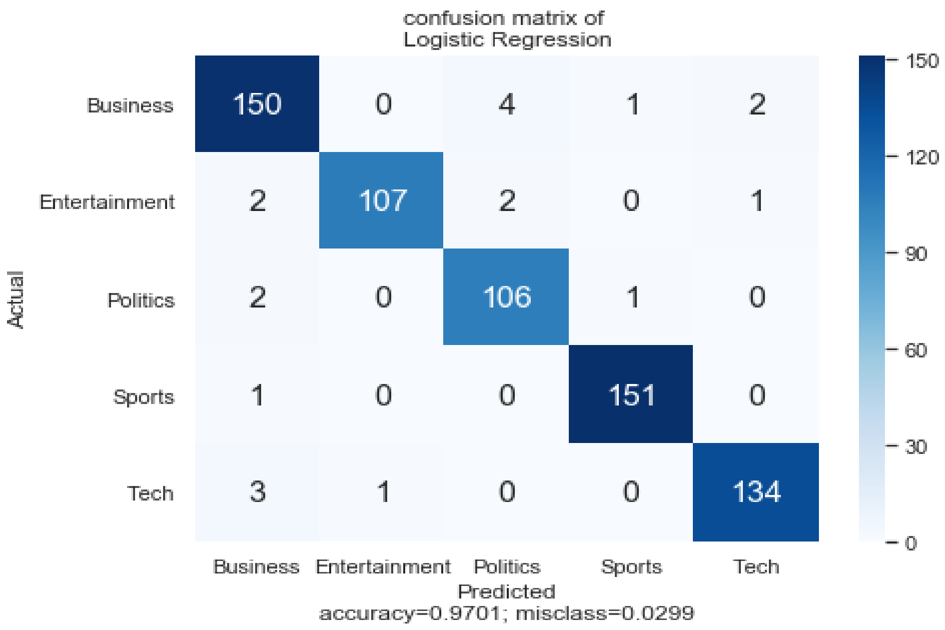

Fig6.5 Confusion Matrix of Logistic Regression

In Fig 6.5 show that confusion matrix of logistic regression in that correctly classified the business class is 150, entertainment class is 107, politics class is 105, sport class is 151 and tech class is 134 no of actual and correctly classified document correctly matched. In that accuracy of correctly classified class is 0.9686 and misclass is 0.0314. In that sport class has the highest no of actual and predicted no of document match and politics class has the lowest no of matching document.

CHAPTER 7

CONCLUSION

In this paper, we have conducted experiments on the BBC Text datasets. After performing sentiment analysis on the datasets using TF-IDF, bag of words model. Conducting experiment with our proposed model ie. using TF-IDF, we found out a good average accuracy of 95.09%. The accuracy percentages for BBC Text datasets came as 93.86% (Multinomial NB), 94.46% (Random forest), 92.81% (KNN), 96.86%(Logistic Regression) respectively [Fig7] with Linear SVM being the highest of all (97.46%). So, from our experiments, we have concluded that when TF-IDF model approach to work on 5 different multi-label BBC Text datasets works better**.**

With rising in the information available online we are going to build a classifier for the classifying the text documents so that the documents should not be in a biased state or label. Thus the document can be searched easily and minimum trust can be laid upon the biased system.

With this system, we are able to classify the documents easily without any bias. This also reduces the time and efforts required for classifying the text documents manually.

CHAPTER 8

FUTURE SCOPE

In future we plan to implement the project as mentioned and use an algorithm for classifying the documents. The work we want to see if the results can be generalized to other languages i.e. Spanish, Italian. Also regional languages like Hindi and other Indian languages can be generalized. If the results were positive, a generic algorithm would be found that worked well on nearly any language. This method put forwarded in the report is meaning-full to online text categorization. Thus it can be scaled up for other languages too. The text classification can be extended for web page classification, e-mail classification. In future, the hybrid networks must be designed and implemented to classify a huge set of documents to achieve improved performance to prove its effectiveness. The scope of future work can deal with Incremental learning, which stores the existing model and processes the new incoming data more efficiently. More specifically, the models with incremental learning can be used in categorization process to improve the document classification.

BIBLIOGRAPHY

1. Linear SVM

   https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html
2. Random Forest

   https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
3. BBC TEXT DATASET

   https://storage.googleapis.com/dataset-uploader/bbc/bbc-text.csv
4. Thorsten Joachims ,Text Categorization with Support Vector Machines: Learning with Many Relevant Features – 2003 Universiy at Dortmund Informatik LS8, Baroper Str. 301 44221 Dortmund, Germany
5. Janez Brank, Marko Grobelnik, Interaction of Feature Selection Methods and Linear Classification Models – 1999 Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
6. Ana CardosoCachopo and Arlindo Limede OliveiraAn Empirical Comparison of Text Categorization Methods-2006.
7. Dimitris Fragoudis,Dimitris Meretakis, Spiros LikothanaSsis, Integrating Feature and Instance Selection for Text Classification1996
8. Jihong Guan, Shuigeng Zhou- Pruning Training Corpus to Speedup Text Classification 2006
9. Xuexian Han Guowei Zu, Wataru Ohyama1-Accuracy improvement of automatic text classification based on feature transformation and Multi-classifier combination 2000
10. Yongguang Bao and Naohiro Ishii Combining Multiple K-Nearest Neighbour Classifiers for Text Classification by Reducts, 2005
11. Jana Novovicova , Antonon Malik and Pavel Pudil Feature Selection using Improved Mutual Information for Text Classification.1998
12. Pio Nardiello, Fabrizio Sebastiani, and Alessandro Sperduti, Discretizing Continuous Attributes in AdaBoost for Text Categorization 1997
13. Yaming Yang, Jan O Pederson, A comparative study on feature selection in text categorization 2004
14. Kluwer Academic Publishers ,Text Categorization with Support Vector Machines, 2005

15. Ke H., Shaoping M Text categorization based on Concept indexing and principal component analysis- 2006

16. Clairvoyance Corporation,Baum Boulevard, Suite ,Improving SVM Text Classification Performance through Threshold Adjustment-1997

17. Performance-Pedro A. C. Sousa, Paulo Pimentão, Bruno René D. Santos, Fernando Moura-Pires3, Feature Selection Algorithms to Improve Documents Classification- ,2005

18. Yiming Yang, An evaluation of statistical approaches to text categorization 2003

19. Ke H., Shaoping M ,Text categorization based on Concept indexing and principal component analysis,1999

20. Kehagias A., Petridis V., Kaburlasos V., Fragkou P -A Comparison of Word- and Sense-based Text Categorization Using Several Classification Algorithms, 2006

21. B. Kessler, G. Nunberg, and H. Schutze.,Automatic detection of text genre,2004

22. Karl-Michael Schneider ,Techniques for Improving the Performance of Naive Bayes for Text Classification-2001

23. Klopotek M. and Woch M.Very Large Bayesian Networks in Text Classification 2003