Tribhuvan University
Institute of Science and Technology

Performance Analysis of Nepali Text Classification using Back Propagation and Naive Bayes Algorithm

Dissertation
Submitted to

Central Department of Computer Science & Information Technology
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science & Information Technology

By
Jamuna Maharjan
Date: 25 June, 2014

Tribhuvan University
Institute of Science and Technology

Performance Analysis of Nepali Text Classification using Back Propagation and Naive Bayes Algorithm

Dissertation
Submitted to

Central Department of Computer Science & Information Technology
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science & Information Technology

By
Jamuna Maharjan
Date: 25 June, 2014

Supervisor
Prof. Dr. Shashidhar Ram Joshi

Tribhuvan University
Institute of Science and Technology
Central Department of Computer Science & Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

... ...
Jamuna Maharjan
Date: 25 June, 2014

Supervisor's Recommendation

I hereby recommend that this dissertation prepared under my supervision by **Ms. Jamuna Maharjan** entitled **"Performance Analysis of Nepali Text Classification using Back Propagation and Naive Bayes Algorithm"** in partial fulfilment of the requirements for the degree of M.Sc. in Computer Science and Information Technology be processed for the evaluation.

... ...
Prof. Dr. Shashidhar Ram Joshi
Department of Electronics & Computer Engineering,
Institute of Engineering,
Pulchowk, Nepal

Date: 25 June, 2014

Tribhuvan University
Institute of Science and Technology
Central Department of Computer Science & Information Technology

LETTER OF APPROVAL

We certify that we have read this dissertation and in our opinion it is satisfactory in the scope and quality as a dissertation in the partial fulfillment for the requirement of Master's Degree in Computer Science and Information Technology.

Evaluation Committee

… … …… … … … … …
Mr. Nawaraj Paudel
Central Department of Computer Science
& Information Technology,
Tribhuvan University, Kathmandu, Nepal
(Head)

… … …… … … … … … … … … … …
Prof. Dr. Shashidhar Ram Joshi
Department of Electronics & Computer
Engineering, Institute of Engineering,
Pulchowk, Kathmandu, Nepal
(Supervisor)

… … …… … … … … …
(External Examinar)

… … …… … … … … …
(Internal Examinar)

Date: 10 July, 2014

ACKNOWLEDGEMENTS

Above all, I thank God for his blessing and submit my graduate to my almighty for providing strength and confidence in me to complete this work.

Secondly, I would like to extent my, gratitude and sincerest thanks to my respected Supervisor **Prof. Dr. Shashidhar Ram Joshi**, Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk for his impressive tutelage, constructive criticism and intellectual support best owed for me sacrificing his invaluable time . His crucial role to make this report culminate is indescribable. I would like to thank my respected teacher **Asst. Prof. Mr. Nawaraj Poudel**, Head of Computer Science & IT Department Kirtipur, TU(Kathmandu, Nepal) for his guidance and encouragement.

I would like to express my gratitude to respected teachers Prof. Dr. Tank Nath Dhamala, Prof. Dr. Subarna Shakya, Prof. Sudarshan Karanjeet, Mr. Min Bahadur Khati, Mr.Bishnu Gautam, Mr. Jagdish Bhatt, Mr. Bikash Balami, Mr. Sarbin Syami, Mr. Dhiraj Pandey, Mr. Arjun Singh Saud and others staffs of CDCSIT for granting me broad knowledge and inspirations within the time of period of two years.

I am tremendously grateful to **Mr. Ashok Kumar Pant** for providing the supportive environment to conduct this research and for his sharp insights, constructive comments and tireless guidance during the research and for the whole study period. I am thankful to colleague **Mr. Harendra Raj Bist** for his through full reading and corrections of the document. Thanks to all my close friends for their supports.

From the beginning to end in all count, I would like to thank my family members for their love, support and encouragement.

As we know that, there won't be 100% accuracy and efficiency in any work done by both machine any human, so there may be some errors in my project. But, I have done my best to complete this dissertation, so any suggestion regarding the mistakes of this work will be always welcomed.

Jamuna Maharjan

ABSTRACT

Automated document classification is the task of assigning the given document into some class of interest. Text classification is the subset of document classification as document can be text, image, music, etc. Document classification has many applications in library science, information science, computer science and others. It can be used for intellectual categorization of documents, indexing of documents, filtering of spams, routing of emails, identification of language, classification of genre, etc.

The problem of automated document classification can be solved in supervised, unsupervised or semi-supervised way. Most of the learning and classification algorithms use document attributes and human inference to learn and classify given documents. In this dissertation work, many Natural Language Processing (NLP) techniques are used for document processing and attribute selection. And, two learning based classification techniques are used namely, Artificial Neural Network(ANN) and Naive Bayes Classifier. ANN is a microbiological model of leaning system and Naive Bayes Classifier is a probability based classification technique.

For the evaluation of the system, we have created Nepali text datasets for five class of documents: Business, Crime, Education, Health and Sports. There are two separate datasets for training and testing of the system. Training set contains total 1253 documents with 243 for Business, 147 for Crime, 250 for Education, 270 for Health, and 343 for Sports. Similarly, testing dataset contains total 89 documents with 19 for Business, 20 for Crime, 12 for Education, 19 for Health, and 19 for Sports. Training and testing is done by splitting training set into two sets while keeping the testing set unique. Experimentation results show, feed-forward multilayer perceptron based neural network classifier has lower classification error rate than Naive Bayes based classifier. MLP classification system has the average system accuracy rate of 87.55%, system error rate of 12.44%, precision rate of 80.29% recall rate of 93.41% and f-score rate of 86.55%. Similarly, Naive Bayes classification system has the average system accuracy rate of 87.09%, system error rate of 12.90%, precision rate of 79.37% recall rate of 93.87% and f-score rate of 86.05%.

Keywords:

Automated Document Categorization, Text Classification, Natural language processing, Nepali language, Preprocessing, Feature extraction, Artificial Neural Networks, Multilayer Perceptron, Naive Bayes Classifier

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI Artificial Intelligence

ANN Artificial Neural Network

API Application Programming Interface

BPN Back Propagation Network

CDT C/C++ Development Components

CM Confusion Matrix

DTM Document Term Matrix

IDE Integrated Development Environment

KNN K-Nearest Neighbor

MT Machine Translation

MLP Multilayer Perceptron

MAP Maximum A Posteriori

NB Naive Bayes

NN Neural Network

NLP Natural Language Processing

PCA Principal Component Analysis

SVM Support Vector Machine

SDK Software Development Kit

TC Text Classification

TF-IDF Term Frequency - Inverse Document Frequency

Chapter 1

INTRODUCTION

1.1 Introduction

1.1.1 Document Classification

Document classification has become a very important issue in the past few years as unstructured, congested, unordered documents in terms of both the amount of time spent on and the resources needed to automatically classify the documents. Text classification [1] is big challenging problems due to the increased availability of documents in digital form and the ensuring need to organize them that are paramount of disordered, congested and unstructured documents as new documents are emerges or lost in different fields. It may be hard to find archived documents search for the previous documents with specified contents or features when the documents are not well structured or organized.

To make the documents manageable and structured, it can be classified into predefined category based on properties of content or feature vector that represents the document in which it belongs to. There are many research work had done for the text classification based on machine learning approaches and rule- based approaches. Paper [1] introduces many learning approaches like Naïve Bayesian, Riccho Method, Neural Network, Decision tree, KNN method and so on for text classification. But we analyze the concept of new classification model which classifies self created database of Nepali collected documents to predefined classes such as education, business, sport, health and crime etc. The task of Nepali document classification is unique challenge in terms of accuracy.

1.1.2 Definition of Text Classification

Let d_1, d_2, $\ldots$, d_n be the number of documents, D is a domain of documents. Let C= $\{c_1$, c_2, $\ldots$,$c_n\}$ be the predefined categories of classes. A value T assigned to (d_j, c_i) indicates a decision to document d_j to c_i, while a value F indicates document d_j not belongs to class c_i.

More formally, the task is approximate the unknown target function $f : D * C \in (T, F)$which describes how the documents ought to be classified. Hence, (T, F) is called classifier.

1.1.3 Principles of Text Classification

The two main principle that are widely accepted are as follows:

- **Content based classification:** In this classification, the weight given to particular subjects in a document determines the class to which the document is belongs to. In automatic classification, it could be the number of times given words appears in a document.
- **Request oriented classification:** In this classification, the anticipated request from users is influencing how documents are being classified.

1.1.4 Methods of Text classification

Text classification includes two main methods: topic based text classification and text genre-based classification.

- **Topic-based text categorization:** classifies documents according to their topics.
- **Texts genre- based classification:** classifies document based on the genre which can be defined as the way a text is created, the way it is edited, the register of language it uses, and the kind of audience to whom it is addressed.

As considering the human interference, text classification can be categorized into manual or automated.

1.1.4.1 Manual text classification

Text classification can be done manually which is very accurate when job is done by experts and is consistent when the problem size and team is small. But, it is difficult and expensive to scale.

1.1.4.2 Automatic text classification:

Automatic Text Classification [2] which automatically involves assigning a text document to a set of pre-defined classes, using a machine learning technique. The classification is usually done on the basis of significant words or features vectors extracted from the text document.

1.1.5 Nepali Text Classification

Nepali Text Classification is the act of dividing a set of input Nepali documents into two or more classes where each document can be said to belong to one or multiple classes. We classified the collected Nepali textual document into five different categories namely Business, Crime, Sport, Health and Education. It is one of the challenging problems in the field of artificial intelligence and machine learning. Today huge amount of information are being associated with the web technology and the internet which are unstructured, unordered and congested. To gather useful information from it these texts have to be classified [1]. Since, huge growth of information flows and especially the explosive growth of Internet promoted the growth of automated text classification. The development of computer hardware provided enough computing power to allow automated text classification to be used in practical applications. Text classification is commonly used to handle spam emails, classify large text collections into topical categories and manage knowledge and also to help Internet search engines.

1.2 Applications of Document Classification

The motivation behind developing text classification systems is inspired by its wide range of applications.

1. **Spam Filtering:** A spam filter is a program that is used to detect unsolicited and unwanted email and prevent those messages from getting to a user's inbox. A text classification system could, in the ideal case, categorize incoming messages into genuine and spam categories, rejecting these that it found to be spam.

2. **Document Organization:** A news or media company will typically get hundreds and thousands of submissions every day. In order to efficiently handle such vast flow of information, there is a need of an automatic text classification system, which would categorize each document by topics so that they could be sent to the relevant recipient maintaining the Integrity of the Specification.

3. **Web page prediction:** Text classification can be used to predict web page the user is likely to click on. Each hyperlink text description is treated as a miniature document. Also a text categorization system could be used to naively predict the next page for a fast look-ahead caching system.

4. **Pornography classification:** The exponential increase of information in internet has raised the issue of information security. Pornography web content is one of the biggest harmful resources that pollute the mind of children and teenagers. Several web content classification approaches have been proposed to avoiding these illicit web contents which are accessing by the children.

5. **Automatic summary evaluation:** Text classification could be applicable to evaluate automatic summarization of text on the basis of feature vector of document.

1.3 Motivation

Nepali Text Classification is a special problem in the domain of Data mining and machine intelligence. The field of Text Classification is split into two different categories: Automatic classification and Manual classification. Due to information overload, efficient classification and retrieval of relevant content has gained significant importance. The problem of classification increases when we operate it in the automatic mode. There are lots of work has been done in this area in the past few years. There are lots of research work have been done for English as well as other language too. But there is no any research work done for the Nepali language, so I was motivated to do this research work for Nepali language. Nepali Text Classification system can also help in automatic organizing of web content, filtering, prediction of web pages and any other. Automatic processing benefits into availability for their contents. Although, a lot of approaches have been proposed for other languages, so automated text Classification is still a major area of research.

1.4 Problem Definition

The high-level task of text classification is to classify the text into predefined classes such as Education, Sports, Health, Business, Crime etc. The problem of Nepali text classification is determined the class of input document according to its content. In this research work, the problem of Nepali text classification is addressed. The classification task is carried out with Naïve Bayesian and Back Propagation approach. In this thesis work, to present models based on Naïve Bayesian and Back Propagation for classification of the text written in which is used for Nepali language. Accuracy is one of the main concerns of the thesis. In order to classify documents, a data sets are prepared by collecting documents from web pages, newspaper, article, notices etc. The system performs document classification by searching the collections of important words in document corpus using TF-IDF properties of text and principle component analysis (PCA) is used to reduce high dimensionality of feature vector of text.

There are many sub-problems in the domain of document classification such as stop-words removal, symbols and punctuation removal, white spaces removal, word stemming, feature extraction, term weighting etc. These are also addressed with the most suitable solutions in the literature for this research work.

1.5 Objectives

The objective of this research work is to investigate various feature extraction techniques and to compare Neural Network based text classification techniques namely Back propagation and Naïve Bayesian probabilistic techniques, to analyse the accuracy of Nepali text classification. Comparative Performance matrices are analyzed. The sub-problem is also addressed such as stop-words removal, symbol and punctuation mark removal, digit and Non-Nepali character removal, stemming, feature extraction, term weighting etc. Main objective is given below:

1. To compare performance accuracy of Multilayer Perceptron and Naïve Bayesian technique on Nepali text classification Problem.

1.6 Contribution of this Dissertation

The main contribution of this thesis to the field of automatic Nepali Text Classification can be seen in its extensive experimental work. A more detailed list of the various contributions is provided below:

- Use of Naive Bayes Classifier and Artificial Neural Network to analysis the Nepali Text classification.
- Built a text classification model for Nepali language.

1.7 Outline of the Document

The remaining part of the document is organized as follows

Chapter 2 describes necessary background information and related work of document classification research on single document as well as in multi document.

Chapter 3 describes in detail the system model and the theoretical approaches for automated Nepali document classification problem. It includes document preprocessing, feature extraction and classification methods.

Chapter 4 describes the implementation details of the system. All the methods described in the Chapter 3 are implemented for system evaluation.

Chapter 5 includes experimentation results and analysis of the systems.

Chapter 6 concludes the system performance and future directions.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Related Work

Text classification, which dates back to the beginning of the 1960 but only in 1990 did it became major principle in the information systems discipline. Text categorization is now being applied in many context based on a vocabulary to document indexing based on controlled vocabulary, to document filtering , word sense disambiguation, etc. “Knowledge engineering” [1] which is more popular approach used in late 1980 which consisting of set of rules encoding expert knowledge on how to classify document under given categories. Furthermore, machine learning approach was introduced due to the increasing popularity of classification introduces an automatic text classifier by learning. If we survey previous works relevant to this research, there exist other kinds of approaches to text categorization i.e heuristic and rule based approaches. Heuristic approaches were already applied to early commercial text categorization systems [3]. However, rule based approaches have poor recall and require a time consuming job of building rules manually. Nowadays, the extensive growth of the Internet and on-line available digital documents, the task of organizing text data becomes one of the critical issues. In these days, the best TC systems use the machine learning approach: the classifier learns rules from examples, and evaluates them on a set of test documents.

There are lots of Machine learning algorithms were introduced among them, four approaches to text categorization KNN (K-Nearest Neighbor), NB (Naïve Bayes), SVM (Support Vector Machine), and BP (Back propagation) have been used more popularly than any other traditional approaches. KNN is evaluated as a simple and competitive classification algorithm where objects are classified by voting several labeled training examples with their smallest distance from each object [1]. Sebastiani mentioned that SVM is also recommendable approach to text categorization. SVMs can handle with exponentially or even infinitely many features, because it does not have to represent examples in that transformed space, the only thing that needs to be computed efficiently is the similarity of two examples. Similarly, NB learns training examples in advance before given unseen examples. It classifies documents based on prior probabilities of categories and probabilities that attribute values belong to categories. The Attributes are considered as independent of each other; its performance is feasible, since its learning is fast and simple [1, 4]. Another popular machine learning approach is BP. It classifies objects by defining a set of input layer, hidden layer and output layer, since output layer defines class label, it is applicable to only linearly separable distribution of training examples. In 1995, BP was initially applied to text categorization by Wiener in his master thesis [5]. The evaluation approach to text categorization shown that BP is better than KNN in the context of classification performance. There are lots of research work conducted based on machine learning by applying above algorithms up to date. Researchers of paper [6] describes the concept of assignment of natural language documents to predefined categories based on the semantic content using neural networks initialization with decision tree found effective for improving text categorization accuracy. Research work [7] introduces the novel combination of support vector machine with word-cluster representation which is compared with SVM based categorization using the bag of words representation which simply outperforms in terms of categorization accuracy and efficiency. An evaluation measure for TC involving either primary or secondary categories and the results are obtained by reformulating well established classification problems such as single or multilevel multiclass classification using Support Vector Machine and kernel based methods, found in paper [8]. The dramatic increase in email creates complexity, hence researchers developed tools using multilayer neural network to implement Back propagation technique for managing unstructured, congested, overloaded, prioritized email mentioned in paper [9]. Moreover, a mobile SMS classification and document classification using Back propagation algorithm and document frequency threshold are discussed in [10, 11, 12]. Researchers in paper [2] survey how to deal with unstructured text, handling large number of attributes and selecting a machine learning techniques to text classification.

Paper [13] describes best performance of Naive Bayes classifier which is measured by cross validation experiments for five predefined categories for classifying about 300 non-vocalized Arabic web documents per category thus accuracy achieved to 92.8%. Furthermore, tests carried out on a manually collected evaluation set which consists of 10 documents from each of the 5 categories, show that the overall classification accuracy achieved over all categories is 62%, and that the best result by category reaches 90%. TC system based on naive Bayes algorithm that integrates strong independence assumptions in categorizing articles showed that the accuracy obtained for training is 81.82% whereas the accuracy for testing is 47.62% [14]. The authors of [4] mentioned classification using Association Rule and Naïve Bayes Classifier; instead of using words word relation i.e. association rule from these words is used to derive feature set from pre-classified text document instead of word to word relation.

Identification in Asian language such as Chinese and Japanese is a difficult task. Researchers used n-gram creating feature vector for a traditional feature selection process during 1994 to 1995. In 2001, significant notable achievement was obtained by calculating feature vector similarities on Chinese and Japanese text classification by avoiding word segmentation [15]. A fast Back propagation neural network is developed which assumes a three-layer structure as fast learning algorithm [16]. The learning efficiency is very high because of the information contained in the vector selected and for the output; Shannon entropy is used to tune the threshold of the binary classifier. Hence, the output of the classifier is approximately accurate and efficient. There are many research works for text classification found based on rule and machine learning approaches. In conclusion, there is no comparison of various classification techniques are available in the literature of Nepali text is made. Since, classification of Nepali text is challenging problem. There is a large corpus of research on the application of text classification in different domains, but no system to date has achieved the goal of system acceptability for Nepali text classification.

2.2 Overview of Data Mining Concepts

Data Mining is the process of extracting knowledge or discovering of new information from large volumes of raw data [17]. The knowledge or information should be new and one must be able to use it. It discovers patterns and relationship using data analysis tools and techniques to build models.

There are two main kinds of models in data mining which are as follow:

- **Predictive model:** In this model, known data results are used to develop a model and that can be used to explicitly predict values.
- **Descriptive model:** In this model, patterns are described from existing data and models are abstract representation of reality which can be reflected to understand business and suggest actions.

2.3 Data Mining

Data mining was introduced in the 1990s and it is traced back along three categories i.e classical statistics, artificial intelligence and machine learning. Data mining is the process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. It is also known as knowledge discovery i.e detecting something new from large–scale or information processing [17]. Its objective is to extract information from a data set and transform it into an understandable structure for further use. It is mainly related to database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating. The data mining step might identify multiple groups in the data, which can then be used to obtain more accurate prediction results by a decision support system.

A Decision Support System [17] is a computer-based information system that supports business or organizational decision making activities. It serves the management, operations, and planning levels of an organization and help to make management decisions, which may be rapidly changing and not easily specified in advance .Hence, the actual data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records (cluster analysis i.e grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups), unusual records (anomaly detection i.e detection of outliers, noise, deviations or exceptions in large data sets) and dependencies (association rule mining i.e detecting interesting relations between the variables in large databases).

2.3.1 Purpose of Data Mining

Data mining can automate the process of finding relationships and patterns in raw data. Thus, results can be either utilized in an automated decision support system or assessed by a human analyst. There are three main reasons to use data mining: especially in science and business areas which need to analyze large amounts of data to discover trends which they could not otherwise find.

- Too much data and too little information.
- There is a need to extract useful information from the data and to interpret the data.
- Predict outcomes of future situations.

2.3.2 Techniques in Data Mining

Data mining is the automated extraction of patterns representing knowledge implicitly stored in large databases, data warehouses and other massive information repositories. Some of the techniques adopted in data mining as given below:

2.3.2.1 Association rule

In this technique, interesting association between attributes that are contained in a database are discovered which are based on the frequency counts of the number of items occur in the event (i.e. a combination of items), association rule tells if item X is a part of the event, then what is the percentage of item Y is also the part of event.

2.3.2.2 Clustering

Clustering is a technique used to discover appropriate groupings of the elements for a set of data. It is undirected knowledge discovery or unsupervised learning i.e, there is no target field and relationship among the data is identified by bottom-up approach.

2.3.2.3 Decision Trees

In this technique, classification is performed by constructing a tree based training instance with leaves having class labels. The tree is traversed for each test instance to find a leaf, and the class of the leaf is predicted class. This is a directed knowledge discovery in the sense that there is a specific field whose value we want to predict.

2.3.2.4 Neural Networks

It is often represented as a layered set of interconnected processors. These processor nodes are frequently referred as neurons so as to indicate a relationship with the neurons of the brain. Each node has a weight connection to several other nodes in adjacent layers, each individual nodes take the received from connected nodes and use the weights together to compute output values.

2.3.2.5 Classification and Prediction

Classification is the technique in which set of documents are classified in the predefined category. Prediction is the process of predicting categorical class labels, constructing a model based on the training set and class labels in a classifying attribute.

2.4 Automatic Text Classification

Automatic text Classification has an important application and research topic since the inception of digital documents. The TC task assigns category label to new documents based on the knowledge gained in a classification system. A wide variety of supervised machine learning algorithms has been applied to this area using a training data set of categorized documents. TC can play an important role in wide range of more flexible, dynamic and personalized task. In general, it can be applied in many applications requiring document organization or selective and adaptive document dispatching. Automatic document classification tasks can be divided into three sorts:

2.4.1 Supervised Text Classification

In Supervised Learning, incorporates an external teacher, the set of possible classes is known in advance so that each output unit is told what its desired response to input signals ought to be. It may require global information during the learning process. Supervised learning include error-correction learning, reinforcement learning and stochastic learning. An important concerning issue of supervised learning is the problem of error convergence, i.e. the minimization of error between the desired and computed unit values. The aim is to determine a set of weights which minimizes the error. A paradigm of supervised learning is least mean square convergence which is known method among many paradigms.

2.4.2 Unsupervised Text Classification

In unsupervised classification, the set of possible classes is not known. After classification, we can try to assign a name to that class. It is called clustering, where the classification is done entirely without reference to external information. It uses no external teacher and is based upon only local information. It is also referred to as self-organization, in the sense that data are organized by itself presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian learning and competitive learning.

2.4.3 Semi-supervised Document Classification

In this classification, parts of the documents are labeled by the external mechanism. It learns with a small set of labeled examples and a large set of unlabeled examples i.e learning with positive and unlabeled examples.

2.5 Automatic Text Classification Techniques

Dealing with unstructured text, handling large number of attributes, examining success of pre-processing techniques , dealing with missing meta data and choice of a suitable machine learning technique for training a text classifier are major concerns of automatic text classification. Since, no single method is found to be superior to all others for all types of classification. Some of widely accepted techniques due to extensive increase in digital documents are as follows:

2.5.1 Support Vector Machines

Support Vector Machines(SVMs) are a generally applicable tool for machine learning. Let training examples be x_i, and the target values $y_i \in \{-1{,}1\}$. SVM searches for a separating hyper plane, which separates positive and negative data samples from each other with maximal margin, in other words, the distance of the decision surface and the closest example is maximal [7, 18].

2.5.2 Artificial neural networks

A neural network is a powerful data-modeling tool that is able to capture and represent complex input/output relationships [19]. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform ”intelligent” tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways:

- a neural network acquires knowledge through learning.
- a neural network's knowledge is stored within inter-neuron connection strengths known as synaptic weights.

Physical nervous system is highly parallel, distributed information processing system having high degree of connectivity with capability of self learning. Human nervous system contains about 10 billion neurons with 60 trillions of interconnections. These connections are modified based on experience.

Artificial neural networks are composed of interconnecting artificial neurons that mimic the properties of biological neurons which can be either be used to gain an understanding of biological neural networks, or for solving artificial intelligence problems without necessarily creating a model of a real biological system. The real biological nervous system is highly complex. Artificial neural network algorithms attempt to abstract this complexity and focus on what may hypothetically matter most from an information processing point of view. Another incentive view is to reduce the amount of computation required to simulate artificial neural networks, so as to allow one to experiment with larger networks and train them on larger data sets.

2.5.3 K-Nearest Neighbor

K-NN is a simplest type of machine learning algorithms. In the learning, function is approximated locally and all computation is deferred until classification. It is well known as lazy learning algorithm or instance based learning. It is non parametric method used for classification and regression in which input consists of k closest training set in the feature space. The output depends on whether K-NN is used for classification or regression [20].

In K-NN classification: the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (where k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. In k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors.

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point.

In this learning, Euclidean distance is commonly used for continuous variables and Hamming distance is used for discrete variables for text classification problems. A common weighting scheme consists in giving each neighbor a weight of $\frac{1}{d}$, where d is the distance to the neighbor.

2.5.4 Decision Tree

This learning algorithm constructs the decision tree with a divide and conquers strategy. Each node in a tree is associated with a set of cases. At the beginning, only the root is present, with associated the whole training set and with all case weights equal to 1. At each node the following, divide and conquer algorithm is executed, trying to exploit the locally best choice, with no backtracking allowed [6].

Let T be the set of cases associated at the node. The weighted frequency freq (C_i,T) is computed of cases in T whose class in C_i, $i \in$ [1,N]. If all cases in T belong to a same class C_j (or the number of cases in T is less than a certain value) then the node is a leaf, with associated class C_j.

If T_1, T_2, $\ldots T_s$ are the subsets of T and T contains cases belonging to two or more classes, then the information gain of each attribute is calculated.

$$I = H(T) - \sum_{i=1}^{s} \frac{|T_i|}{|T|} H(T_i) \tag{2.1}$$

Where,

$$H(T) = -\sum_{j=1}^{n} \frac{freq(c_j, T)}{|T|} * \log(\frac{freq(c_j, T)}{|T|}) \tag{2.2}$$

is the entropy function.

2.5.5 Naive Bayes Probabilistic Model

The most widely used method for text categorization is Naive Bayes Classifier, each word position in a document is defined as an attribute and the value of that attribute to be the word found in that position. Naive Bayes categorization is given by,

$$V_{NB} = argmaxP(V_j)P(a_i|V_j) \tag{2.3}$$

To conclude, the Naive Bayes Categorization V_{NB} is the categorization that maximizes the probability the words that were actually found in the training documents. Naive Bayes is very popular among spam filters, because it is very fast and simple for both training and testing. Hence, it is simplicity to learn from new examples and the ability to modify an existing model.

2.6 Machine Learning

Machine learning is the development of algorithms and techniques, which allow computers to learn. It is a wide area of artificial intelligence. Machine learning has a broad spectrum of applications including search engines, medical diagnosis, and bio-informatics, detecting credit card fraud, stock market analysis, classifying DNA sequences, speech recognition, computer games, robot locomotion and spam filtering.

2.6.1 Learning

Learning is a process by which weights are determined, the free parameters of a neural network are adapted through a process of stimulation by the environment in which the network is embedded. Since, every neural network possesses knowledge, contained in the values of the connections weights. The knowledge is modified in the network as a function of experience implies a learning rule for changing the values of the weights. All learning methods used for neural networks can be classified into two major categories: Supervised Learning and Unsupervised Learning.

2.6.2 Text Categorization as a Supervised Machine Learning Problem

Text classification is a well-established area of research within the field of machine learning. A machine learner acquires or learns a general concept from specific training sets; it uses available examples of data to build a model that best generalizes to all possible sets. One of the issues facing a machine learning text classification is which examples of data set should be used as training data for different class. The proportion of text may represents the document to fall into different category.

Supervised machine learning method prescribes the input and output format. Machine Learning (ML) algorithms typically use a vector-space (attribute-value) representation of examples; mostly the attributes correspond to words. However, word-pairs or the position of a word in the text may have considerable information and practically infinitely many features can be constructed to enhance classification accuracy. Machine learning focuses on prediction based on known properties learned from the training data. Since, no Machine Learning classification is likely to be perfect. Classification errors are inevitable but the cost of misclassifying text document is the extra challenge facing any text classification techniques. Misclassified text document are known as false positives which are unacceptable and inconvenience.

The classifier uses the training set to learn how to associate labels with documents. The learning mechanism may be statistical, geometrical, rule-based, neural and something else. The inputs to a text classification problem can be viewed from three labels which are as follows:

- A set of labeled training documents (the ”training set”)
- A set of labeled text documents (the ”testing set”) and
- A classification algorithm (the ”classifier”).

The trained classifier is used to predict the label of each document in the test set. Since, the correct labels are already known, the classifier can be scored based on its accuracy.

Chapter 3

RESEARCH METHODOLOGY

3.1 Automatic Text Classification System Overview

The top level document classification system is divided into five sub-systems, data acquisition, preprocessing, feature extraction, dimensionality reduction and classification. Each stages of this theoretical model are briefly described in this section.

Detail of each subsystem is given in later sections. The top level of data flow diagram of the proposed system is given in Figure 3.1. Various stages have to be performed to achieve automatic text classification for the Nepali documents. Detailed sub-system flow is given in Figure 3.2.

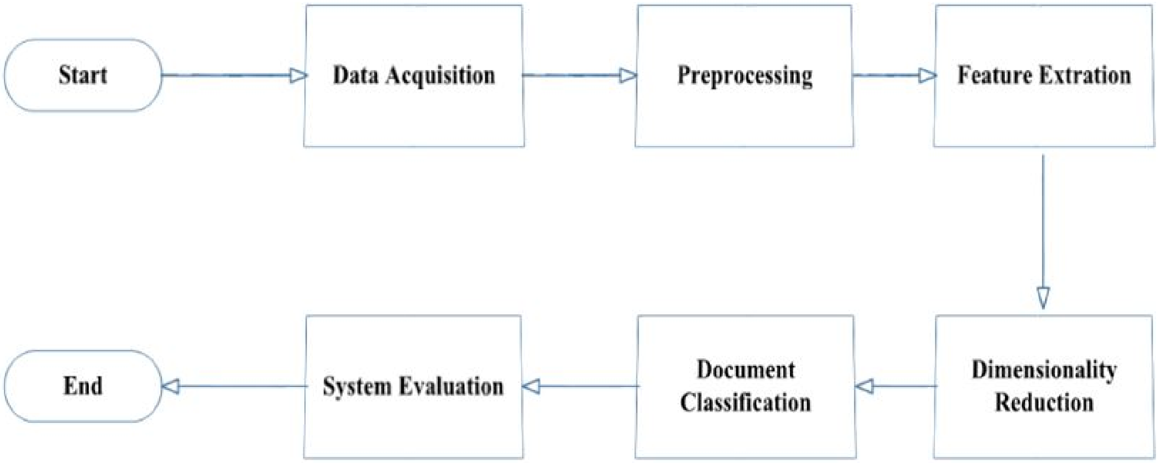


Figure 3.1: Flow-chart of Automated Text Classification System.

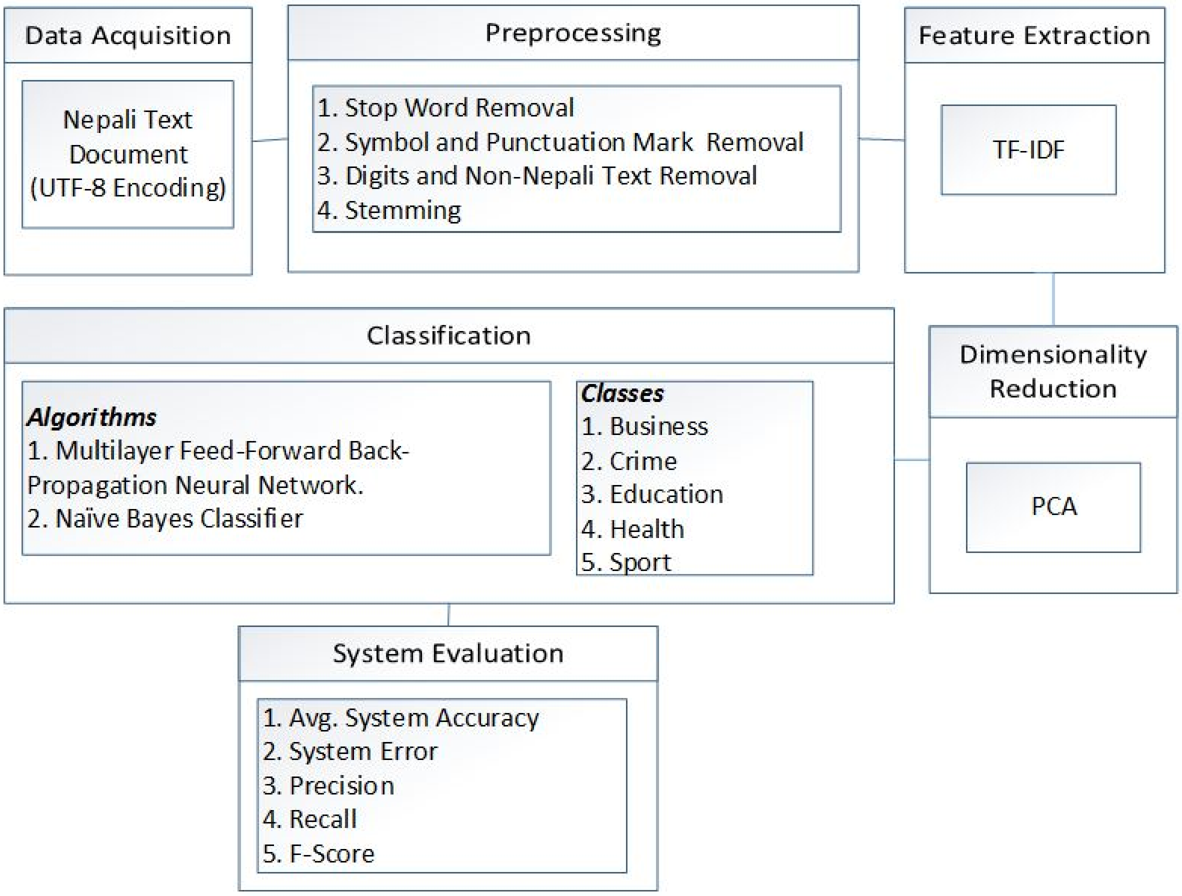


Figure 3.2: The Detail Architecture of Automatic Text Classification System for Nepali Language.

3.2 Data Acquisition

Data are acquired from different sources like Nepali newspaper, articles, books, magazines. Data which are collected in huge amount are stored in UTF-8. UTF-8 is a variable-width encoding that can represent every character in the Unicode character set. The collected documents are transformed into a uniform format which is understandable by machine learning algorithm as input.

3.3 Preprocessing

An approach applied to remove the set of non-content-bearing functional words from the set of words produced by word extraction is known as stop words removal. The next step of text mining process is text preprocessing in which collected documents are analyzed syntactically or semantically. Since, the collected text document is considered as a bag of words because the words and its occurrences are used to represent the document. The algorithm applied in this stage is stemming, digit and non-Nepali text removal, stop word removal, number removal and strip whitespaces.

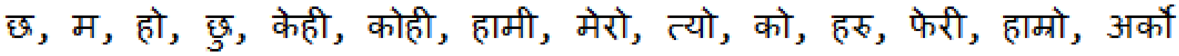
Another task is tokenization which is the task of chopping it up into pieces, called tokens. We have created dictionary for those common words that are useless and has a less discriminative value that do not add meaningful content to the document (auxiliary verbs, conjunctions and articles).

3.3.1 Stop Word Removal

Stop words are high-frequency words of a language which rarely contribute to useful information in terms of document relevance and appear frequently in the text but provide less meaning in identifying the important content of the document [9, 10]. Those common words that are consider as stop words , useless and has a less discriminative value that do not add meaningful content to the documents are auxiliary verbs , conjunctions and articles. Words are pruned at the processing phase to reduce the number of features vector. The stop word lists for English and other languages are freely available on the Web and often utilized in classification. But, we cannot find easily the stop word list for Nepali language. We have prepared the list of stop words for Nepali language manually for this dissertation. During the removal procedure all the words that appear in a list of stop words are removed by matching from the source documents. Some of the Nepali stop words are given in Figure 3.3. The details of dictionary used for stop word removal is explained in Section 5.2.

Algorithm 3.1 Stop Word Removal

```
Read text document.
Match the token of document with token in the stop word dictionary.
Remove matched token from document.
Repeat until all stop words are not removed.
```



छ, म, हो, छु, केही, कोही, हामी, मेरो, त्यो, को, हरु, फेरी, हाम्रो, अर्को

Figure 3.3: Sample Stop Words.

3.3.2 Symbols Removal

Document may contain some symbols to represent some information. They are not so informative, like the symbols $, #, , %, etc are used to denote some information in the document. So, we need to remove such symbols from the document before feature extraction. As in any language, punctuations are used to organize the text and give the sentence a powerful meaning.

The punctuation in the text summary does not have any value, so we remove all punctuation which are not full stop. The data dictionary used for symbol removal is explained in section 5.2.

Algorithm 3.2 Symbol and Punctuation mark Removal

```
1: Read text document.
2: Match the token of document with token in the symbol and punctuation mark dictionary.
3: Remove matched token from document.
4: Repeat until all stop words are not removed.
```

3.3.3 Stemming

In a text document, a word may exist in different morphological variants, stemming reduces such different morphological variant words into the number of unique root words. In text categorization and many other similar tasks, the root word may have different forms. So, it is desirable to combine these morphological variants of the same word into one canonical form. The different morphological variant of the same word which is combined into a single canonical form is called stemming or base word transformation [8].

Many NLP applications which use words as basic elements employ stemmers to extract the stems of words. Mainly, it is used in information retrieval systems to improve performance. Actually, this operation reduces the number of terms in the information retrieval system, thus decreasing the size of the index files. Stemming helps to obtain the stem or root of each word, which ultimately helps in semantic analysis and faster processing. There need a specific language dependent stemmer, and is requires some significant linguistic expertise in the language. A typical simple stemmer algorithm involves removing suffixes/prefixes using a list of frequent suffixes/prefixes, while a more complex one would use morphological knowledge to derive a stem from the words. The stemmer which simply prunes the suffixes/prefixes using the list of frequent suffixes/prefixed is very efficient and lightweight approach compared to morphologic parsing. Even though there are some advanced stemmers for languages such as English, the algorithms which they employ do not work well for highly inflected languages such as Nepali.

Since, Nepali is a highly inflected language so there are many word forms to denote a single concept. This situation is highly effected for the frequency of a term and therefore words have to be stemmed before getting their frequencies. There is light weight stemmer and morphological Analyzer that were developed under Madan Puraskar Pustakalaya, Nepal [21]. Stemming algorithm for Nepali language is given in Algorithm 3.3.

Algorithm 3.3 Stemming

```
Read text document.
Do the following for the string sequence in the input word
Strip off ◌े◌ी appeared at the end of the word, the very last letter of the input word. Record
    as the suffix associated.
Strip off ◌े◌ीय/◌े◌ीया appeared at the end of the word from the end of the input word.
    Record इय/इया as the suffix associated.
Strip off ◌े◌ाइ from the input word which is appeared at the end of the word from the end.
    Add ◌े to the end of the resulting word if the last letter of the word formed is a consonant.
    Record आइ as the suffix associated.
Exception holds the letter व. If the last character of the resulting word is व, strip it off and
    add ◌े◌ा Record आइ as the suffix associated.
If the initial letter is a vowel, stripe off ◌े◌ाइ from the word and insert ◌े◌ा in front of the
    character which is followed by ◌े◌ाइ.
Stripe off ◌े. from the end of the word. Look for the resulting word in the free morpheme
    list. If found record ◌े as the suffix.
The remaining part of the input word is root word.
```

3.4 Document Representation

Each document is typically represented by the feature vector or the bag of words. The most common text representation is bag of word approach (BoW). Here, text is represented as a vector. The BoW vectors are then refined by feature extraction, where vectors are removed from the representation using computationally less discrimination value. The set of feature vectors is of very high dimension in the vector space and each vector represents a unique term. In order to improve the scalability of the text categorization system, dimensionality reduction techniques should be employed to reduce the dimensionality of the feature vectors before they are fed as input to the text classifier.

3.4.1 Document Term Matrix

If we have a large collection of documents and hence a large number of document vectors then it is more convenient to organize into a matrix. The row vectors of the matrix correspond to terms (words) and the column vectors correspond to documents. Hence, describes the frequency of terms that occur in a collection of documents known as Document–Term matrix. Document frequency of a feature is the number of documents in which the frequency occurs and is class independent because of its simple computation and good performance [22].

3.5 Feature Extraction

Feature extraction plays major role in the classification system and it is heart of the classification system. A good feature sets should represent characteristic of a class that helps to distinguish it from other classes, while remaining invariant to characteristic differences within the class. Hence, to improve the accuracy of the classifier, it is necessary to identify a set of “good” features for object representation. To create improved features, measurement of various object properties are carried out to identify good features from a set of raw features [7].

3.5.1 Term Frequency-Inverse Document Frequency (TF-IDF)

For machine learning method, TF-IDf is widely accepted technique.Tf-idf stands for term frequency-inverse document frequency, often used in information retrieval and text mining. It is used to evaluate how important a word to a document in a collection or corpus. The importance increases proportionally to the number of times as a word appears in the document but is offset by the frequency of the word in the corpus. Every term are represented as a vector in a vector space model. Therefore, most vectors represented for document are sparse. This is more efficient method to extract the feature, since TF-IDF is constructed based on the word occurs many times in a document. TF-IDF is often used as a weighting factor in text classification. It is a central tool widely used in scoring and ranking a document’s relevance given a user query and also used for stop-words filtering classification [7, 4].

Mathematically, it can be calculated as,

$$W_{ik} = \frac{tf_{ik} log(\frac{N}{nk})}{\sum_{k=1}^{t} (tf_{ik})^2 [\log(\frac{N}{nk})]^2} \tag{3.1}$$

Where,
tf = Term frequency.
idf = Inverse document frequency.
T_k = Term k in document D_i.
tf_{ik} =frequency of term T_k in document D_i.
idf_k =Inverse document frequency of term T_k in document C.
N = Total number of document in the collection C.
n_k = The number of document in C that contain T_k.
$idf_k = \log(\frac{n_k}{N})$

3.6 Dimensionality Reduction

3.6.1 Principal Component Analysis

Principal Component Analysis (PCA) is one of a family of techniques for taking high-dimensional data, and using the dependencies between the variables to represent it in a more tractable, lower-dimensional form, without losing too much information. PCA can be done by eigen value decomposition of a data covariance or correlation matrix or singular value decomposition of a data matrix, usually after mean centering and normalizing or using Z-scores, the data matrix for each attribute. Using PCA, the dimension reduction process will reduce the original data vector into small number of relevant features [23].

Let M to be the matrix of document terms weights as follows.

$$M = \left\{ \begin{matrix} a11 & a12 & ... & a1m \\ a21 & a22 & .. & a2m \\ ... & ... & ... & ... \\ an1 & an2 & ... & anm \end{matrix} \right\} \tag{3.2}$$

Where, aij refers to the terms in the collection of documents, n is the number of terms and m is the number of documents. Then we calculate the mean a and subtract it from each data points $a - \bar{a}$. After variance-covariance matrix M can be calculated, where the new value of $aij = (aj - \bar{a})(ai - \bar{a})$. Then we determine eigenvalues and eigenvectors of the matrix M where C is a real symmetric matrix so a positive real number λ and a nonzero vector α can be found such that, $C\alpha = \lambda\alpha$ where λ is called an eigenvalue and α is an eigenvector of C. In order to find a nonzero vector α the characteristic equation $|C - \lambda I|$ must be solved. If C is an $n * n$ matrix of full rank, n eigenvalues can be found such that $(\lambda 1, \lambda 2, .., \lambda n)$. By using $(C - \lambda I)\alpha = 0$, all corresponding eigenvectors can be found. The eigenvalues and corresponding eigenvectors will be sorted so that $\lambda 1 \geq \lambda 2 \geq \cdots \geq \lambda n$. Then we select the first $d = n$ eigenvectors where d is the desired value.

3.7 Bayes Theorem

Bayes' theorem was named after the Reverend Thomas Bayes during 1702–1761, who studied how to compute a distribution for the probability parameter of a binomial distribution [24]. Bayes Theorem is defined as

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \tag{3.3}$$

Where,
$P(H|X)$is the posterior probability of H conditioned on X.
$P(H)$ is the prior probability of hypothesis H.
$P(X|H)$is the posterior probability of X conditioned on H.

$P(X)$is the prior probability of hypothesis X.

3.7.1 Naive Bayesian Classifier

A Bayes classifier is a simple probabilistic classifier based on applying Bayes theorem (from Bayesian statistics) with strong (naive) independence assumptions. In simple terms, a naïve Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4th diameter [25]. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple. Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

In spite of their naive design and apparently over-simplified assumptions, Naive Bayes classifiers have worked quite well in many complex real world situations. In 2004, analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficiency of Naive Bayes classifiers [26]. Still, a comprehensive comparison with other classification methods in 2006 showed that Bayes classification is outperformed by more current approaches, such as boosted trees or random forests [27].

An advantage of the Naive Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined not the entire covariance matrix.

3.7.2 The Naive Bayes probabilistic model

The probability model for a classifier is a conditional model

$$P(C|F_1, F_2, \ldots, F_n) \tag{3.4}$$

Over a dependent class variable C with a small number of outcomes or classes, conditional on several feature variables F_1 through F_n. The problem is that if the number of features is large or when a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable.Using Bayes' theorem, we write

$$P(C|F_1, F_2, \ldots, F_n) = \frac{P(C)P((F_1, F_2, \ldots, F_n)|C)}{P(F_1, F_2, \ldots, F_n)} \tag{3.5}$$

In plain English the above equation can be written as,

$$posterior = \frac{(prior * likelihood)}{evidence} \tag{3.6}$$

In practice we are only interested in the numerator of that fraction, since the denominator does not depend on and the values of the features F_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model,

$$P(C, F_1, F_2, \ldots, F_n)$$

Which can be rewritten as follows, using repeated applications of the definition of conditional probability:

$$P(C, F_1, F_2, \ldots, F_n) = P(C)P(F_1|C)P(F_2|C, F_1)P(F_3|C, F_1, F_2), \ldots, P(F_n|C, F_1, F_2, \ldots, F_{n-1}) \tag{3.7}$$

Now the "Naive" conditional independence assumptions come into play: assume that each feature F_i is conditionally independent of every other feature F_j for $j \neq i$. This means that $P(F_i|C, F_j) = P(F_i|C)$ For $i \neq j$, and so the joint model can be expressed as

$$P(C|F_1, F_2, \ldots, F_n) = P(C) \prod_{i=1}^{n} P(F_i|C) \tag{3.8}$$

This means that under the above independence assumptions, the conditional distribution over the class variable can be expressed like this:

$$P(C|F_1, F_2, \ldots, F_n) = \frac{1}{z} P(C) \prod_{i=1}^{n} P(F_i|C) \tag{3.9}$$

Where z (the evidence) is a scaling factor dependent only on, i.e. a constant if the values of the feature variables are known.

3.7.3 Parameter Estimation

All model parameters (i.e., class priors and feature probability distributions) can be approximated with relative frequencies from the training set. These are maximum likelihood estimates of the probabilities. A class' prior may be calculated by assuming equi-probable classes (i.e., priors = 1 / (number of classes)), or by calculating an estimate for the class probability from the training set (i.e., (prior for a given class) = (number of samples in the class) / (total number of samples)). To estimate the parameters for a feature's distribution, one must assume a distribution or generate nonparametric models for the features from the training set [28]. If one is dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution.

For example, suppose the training data contains a continuous attribute, x. we first segment the data by the class, and then compute the mean and variance of x in each class. Let μ_c be the mean of the values x in associated with class c, and let σ_c^2 be the variance of the values in x associated with class c. Then, the probability of some value given a class, $P(x = v|c)$, can be computed by plugging v into the equation for a Normal distribution parameterized by μ_c and σ_c^2.

That is,

$$P(x = v|c) = \frac{1}{\sqrt{(2\pi\sigma_c^2)}} e^{\frac{-(v-\mu_c)^2}{2\sigma_c^2}} \tag{3.10}$$

Another common technique for handling continuous values is to use binning to discretize the values. In general, the distribution method is a better choice if there is a small amount of training data, or if the precise distribution of the data is known. The discretization method tends to do better if there is a large amount of training data because it will learn to fit the distribution of the data. Since Naive Bayes is typically used when a large amount of data is available (as more computationally expensive models can generally achieve better accuracy), the discretization method is generally preferred over the distribution method.

3.7.4 Sample Correction

If a given class and feature value never occurs together in the training set then the frequency-based probability estimate will be zero. This is problematic since it will wipe out all information in the other probabilities when they are multiplied. It is therefore often desirable to incorporate a small-sample correction in all probability estimates such that no probability is ever set to be exactly zero.

3.7.5 Constructing a classifier from the probability model

The discussion so far has derived the independent feature model, that is, the Naive Bayes probability model. The Naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the maximum a posteriori or MAP decision rule. The corresponding classifier is the function classify defined as follows:

$$Classify(f_1, f_2, \ldots, f_n) = argmax_c P(C = c) \prod_{i=1}^{n} P(F_i = f_i | C = c) \tag{3.11}$$

3.7.6 Method Applied for Nepali Document Classification

Here is worked example of Naive Bayesian classification to the document classification problem. Imagine that documents are drawn from a number of classes of documents which can be modeled as sets of words where the (independent probability that the i^{th} word of a given document occurs in a document from class C can be written as $P(W_i|C)$. Then the probability that a given document D contains all of the words W_i, given a class C, is

$$P(D|C) = \prod_{i} P(W_i|C) \tag{3.12}$$

Now by definition,

$$P(D|C) = \frac{P(D \cap C)}{P(C)} \tag{3.13}$$

And

$$P(C|D) = \frac{P(C \cap D)}{P(D)} \tag{3.14}$$

Bayes theorem manipulates these into a statement of probability in terms of likelihood.

$$P(C|D) = P(D|C)\frac{P(C)}{P(D)} \tag{3.15}$$

Since we have five classes such that every element is in either one or another.

$$P(D|B) = \prod_i P(W_i|B) \tag{3.16}$$

$$P(D|C) = \prod_i P(W_i|C) \tag{3.17}$$

$$P(D|E) = \prod_i P(W_i|E) \tag{3.18}$$

$$P(D|H) = \prod_i P(W_i|H) \tag{3.19}$$

$$P(D|S) = \prod_i P(W_i|S) \tag{3.20}$$

And

$$P(D|\neg B) = \prod_i P(W_i|\neg B) \tag{3.21}$$

$$P(D|\neg C) = \prod_i P(W_i|\neg C) \tag{3.22}$$

$$P(D|\neg E) = \prod_i P(W_i|\neg E) \tag{3.23}$$

$$P(D|\neg H) = \prod_i P(W_i|\neg H) \tag{3.24}$$

$$P(D|\neg S) = \prod_i P(W_i|\neg S) \tag{3.25}$$

Using the Bayesian result above, we can write:

$$P(B|D) = \frac{P(B)}{P(D)} \prod_i P(W_i|B) \tag{3.26}$$

$$P(C|D) = \frac{P(C)}{P(D)} \prod_i P(W_i|C) \tag{3.27}$$

$$P(E|D) = \frac{P(E)}{P(D)} \prod_i P(W_i|E) \tag{3.28}$$

$$P(H|D) = \frac{P(H)}{P(D)} \prod_i P(W_i|H) \tag{3.29}$$

$$P(S|D) = \frac{P(S)}{P(D)} \prod_i P(W_i|S) \tag{3.30}$$

And

$$P(\neg B|D) = \frac{P(\neg B)}{P(D)} \prod_i P(W_i|\neg B) \tag{3.31}$$

$$P(\neg C|D) = \frac{P(\neg C)}{P(D)} \prod_i P(W_i|\neg C) \tag{3.32}$$

$$P(\neg E|D) = \frac{P(\neg E)}{P(D)} \prod_i P(W_i|\neg E) \tag{3.33}$$

$$P(\neg H|D) = \frac{P(\neg H)}{P(D)} \prod_i P(W_i|\neg H) \tag{3.34}$$

$$P(\neg S|D) = \frac{P(\neg S)}{P(D)} \prod_i P(W_i|\neg S) \tag{3.35}$$

3.8 Back Propagation Learning Algorithm

Back propagation is a form of supervised learning for multi-layer nets, also known as the generalized delta rule. It is a multilayer feed forward supervised network. It provides an effective means of allowing a computer to examine data patterns that may be incomplete or noisy.

In this learning algorithm, error data at the output layer is ”back propagated” to earlier ones, allowing incoming weights to these layers to be updated. It is most often used as training algorithm in current neural network applications. The back propagation algorithm was developed by Paul Werbos in 1974 and rediscovered independently by Rumelhart and Parker [29]. Since its rediscovery, the back propagation algorithm has been widely used as a learning algorithm in feed forward multilayer neural networks. In general, the difficulty with multilayer Perceptrons is calculating the weights of the hidden layers in an efficient way that resulting the least (or zero) output error; it becomes more difficult if there are more hidden layers. To update the weights, one must calculate an error. At the output layer this error is measured; since error is the difference between the actual and desired (target) outputs. At the hidden layers, however, there is no direct observation of the error; hence, some other technique must be used. To calculate an error at the hidden layers that will cause minimization of the output error, as this is the ultimate goal [19].

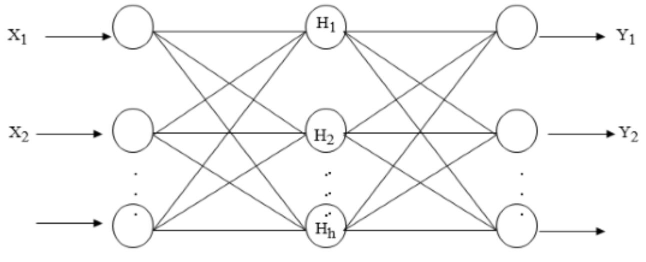


Figure 3.4: Multilayer Back propagation Network.

Let us assume, $[X_1, X_2, \ldots, X_n]$ be the input layer which can have more than one hidden layer. Let net1, net2, . . . , neth derive unit for each neuron and target output H_1, H_2,. . . ,H_h, to be used as the input to derive the result for output layer and $[Y_1, Y_2, \ldots, Y_j]$ be the output layer and W_{ij}. be weights. The nodes in the hidden layers organize themselves in a way that different nodes learn to recognize different features of the total input space. Initially, set up the network based on the problem domain and randomly generate weights W_{ij}. Then feed a training set, $[X_1, X_2, \ldots, X_n]$, into BPN in order to compute the weighted sum and apply the transfer function on each node in each layer. Feeding the transferred data to the next layer until the output layer is reached. The output pattern is compared to the desired output and an error is computed for each unit. Feedback error is back to each node in the hidden layer. Each unit in hidden layer receives only a portion of total errors and these errors then feedback to the input layer, until the error is very small.

Back propagation algorithm

1. Back propagation (training examples, η, n_{in}, n_{hidden}, n_{out})
2. Each training examples is a pair of the form (x, t) where x is the vector of network input values and t is the vector of target output values.
3. η is the learning rate. n_{in} is the number of network input, n_{hidden} is the number of units in hidden layer and n_{out} the number of output units.
4. The input unit from unit I into unit j is denoted by x_{ij} and weight from unit I into unit j is denoted by w_{ij}
5. Create a feed-forward network with n_{in} inputs, n_{hidden} hidden units and n_{out} output units.
6. Initialize all network weight to a small random numbers (e.g.,Between -0.5 and 0.5).
7. Until the termination condition is met do
8. For each (x, t) in training examples, do
   (a) **Propagate the input forward through the network**
   - Input the instance x to the network and the compute the output o_u of every unit in the network

   (b) **Propagate the error backward through the network**
   - For each network output unit k, calculate its error term δ_k.

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k) \tag{3.36}$$

   - For each hidden unit h, calculate its error term δ_h

$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} W_{kh}\delta_k \tag{3.37}$$

   - Update each network weight

$$W_{ji} \leftarrow w_{ji} + \Delta w_{ji}\Delta \tag{3.38}$$

$$W_{ji} = \eta \delta_j X_{ji} \tag{3.39}$$

3.9 Categories of classification for Nepali documents

For the empirical evaluation of the hypothesis, Nepali document database is created by collecting documents from the web pages, books, newspaper etc. There are five classes of documents:

1. Business

2. Crime
3. Education
4. Health
5. Sport

More detail about the collected datasets is given in the Section 5.1.

3.10 System Evaluation Measures

The correctness of a classification can be evaluated by computing the number of correctly recognized class examples (true positives), the number of correctly recognized examples that do not belong to the class (true negatives), examples that either were incorrectly assigned to the class (false positives) and examples that were not recognized as class examples (false negatives). These four counts constitute a confusion matrix [30].

Measures for multi-class classification based on a generalization of the measures of binary classification for many classes C_i are given below. Where, tp_i represent true positive for class C_i, fp_i represent false positive for class C_i, fn_i represent false negative for class C_i, tn_i represent true negative for class C_i, and μ represent micro averaging.

3.10.1 Average System Accuracy

Average system accuracy evaluates the average per-class effectiveness of a classification system.

$$Average\ Accuracy \;=\; \frac{\sum_{i=1}^{l} \frac{tp_i+tn_i}{tp_i+fn_i+fp_i+tn_i}}{l} \tag{3.40}$$

3.10.2 System Error

System error is the average per-class classification error of the system.

$$Error\ Rate \;=\; \frac{\sum_{i=1}^{l} \frac{fp_i+fn_i}{tp_i+fn_i+fp_i+tn_i}}{l} \tag{3.41}$$

3.10.3 Precision

Precision (also called positive predictive value) is the number of correctly classified positive examples divided by the number of examples labeled by the system as positive.

Micro precision is the agreement of the data class labels with those of a classifiers if calculated from sums of per-test decisions.

$$Precision_{\mu} \;=\; \frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l} tp_i + fp_i} \tag{3.42}$$

3.10.4 Recall

Recall(also called sensitivity)is the number of correctly classified positive examples divided by the number of positive examples in the test dataset.

Micro recall is the effectiveness of a classifier to identify class labels if calculated from sums of per-test decisions.

$$Recall\mu \;=\; \frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l} tp_i + fn_i} \tag{3.43}$$

3.10.5 F-Score

F-Score is the combination of the precision and recall.

Micro F-Score is the relation between data's positive labels and those given by a classifier based on sums of per-test decisions.

$$Fscore\mu \;=\; \frac{(\beta^2+1)Precision_\mu Recall_\mu}{\beta^2 Precision_\mu + Recall_\mu} \tag{3.44}$$

where β is the measure of effectiveness of classification with respect to class β times as much importance to recall as precision.

Chapter 4

IMPLEMENTATION TOOLS AND TECHNIQUES

All the algorithms of purposed classification system are implemented in Eclipse platform 4.3 version. Eclipse is installed on a Intel(R) Core(TM) i5 CPU M 520 @ 2.40 GHz, 2.40 GHz processor. The Computer has total main memory of 4 Gigabyte and 64-bit Operating system, x64-based processor and Microsoft Windows8 Enterprise operating system installed in it.

4.1 Programming Language and IDE

4.1.1 Java

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities than either C or C++. Java applications are typically compiled to byte-code (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general-purpose, concurrent, class-based, object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers ”write once, run anywhere” (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java is as of 2012 one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users. The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1995. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java and GNU Class path.

4.1.2 Eclipse IDE

The Eclipse Platform is specially designed for building integrated development environments (IDEs), and arbitrary tools. The Eclipse Platform's principal role is to provide tool providers with mechanisms to use, and rules to follow, that lead to seamlessly-integrated tools. These mechanisms are exposed via well-defined API interfaces, classes, and methods. The Platform also provides useful building blocks and frameworks that facilitate developing new tools. It contains large set of functionality required to build an IDE. It supports both GUI and non-GUI-based application development environments and runs on a wide range of operating systems, including Windows®, LinuxTM, Mac OS X, Solaris AIX and HP-UX.

Eclipse Platform enables the tool or application to integrate with other tools and applications also written using the Eclipse Platform. The Eclipse Platform is turned in a Java IDE by adding Java development components (e.g. the JDT) and it is turned into a C/C++ IDE by adding C/C++ development components (e.g. the CDT). It becomes both a Java and C/C++ development environment by adding both sets of components. However, the Eclipse Platform is itself a composition of components; by using a subset of these components, it is possible to build arbitrary applications. Hence, Eclipse SDK, Eclipse Rich Client Platform (RCP) and Eclipse IDE are popular framework which are widely used.

- The Eclipse SDK includes Java Development Tools and Plug-in Development Environment.
- It includes building applications that work in conjunction with application servers, databases and other backend resources to deliver product providing a rich and consistent experience for it's users.
- Eclipse IDE is designed to Support the construction of a variety of tools for application development. It support tools to manipulate arbitrary content types (e.g., HTML, Java, C, JSP, EJB, XML, and GIF).

4.2 Machine Learning Library and Plug-ins

A piece of program or application that is use to create, debug, maintain and support other applications is called Programming tool. It is a simple program which is integrated together to accomplish a task or support other program, to fix application. They make easier to do some specific tasks such as IDE combine the features of many tools in one package to develop applications.

Another term is Plug-in, which is a software component that is used to support a specific feature to an existing software application, to enable customization. For this research work, Weka is used as additional tool.

4.2.1 Weka

Weka (Waikato Environment for Knowledge Analysis) is a comprehensive and free available suite of Java class libraries that support the implementation of machine learning algorithms for data mining tasks. Weka contains tools for: data pre-processing, classification, regression, clustering, association rules, and visualization. It allows users to apply Weka class libraries of machine learning techniques to their own data regardless of computer platform. It is developed in Java platforms to support data mining tools, a suite of Java packages to provide facilities for developers.

The core package contains classes that are accessed from almost every other class in Weka. The most important classes in it are Attribute, Instance, and Instances. An object of class ”Attribute” represents an attribute-it contains the attribute’s name, its type, and, in case of a nominal attribute, it’s possible values. An object of class ”Instance” contains the attribute values of a particular instance; and an object of class Instances contains an ordered set of instances-in other words, a dataset.

Weka is used for Neural Network and Naive Bayes Classifier Training and Testing.

Chapter 5

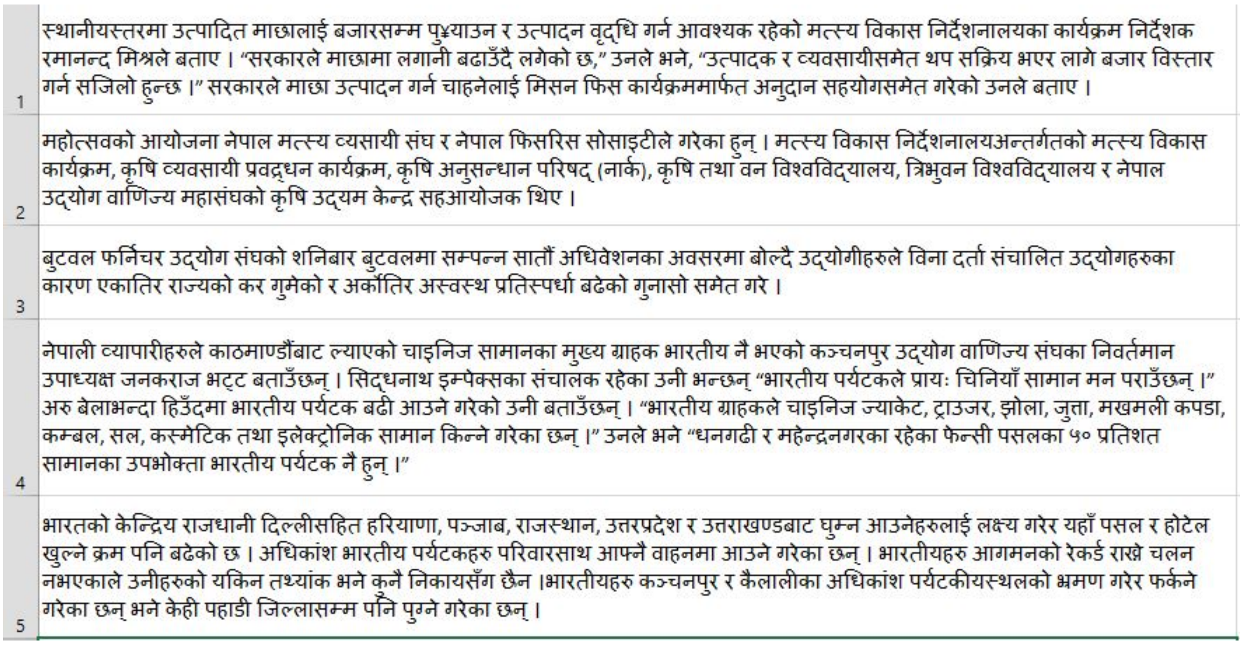
EXPERIMENTATIONS AND RESULTS

Multi class Nepali text classification system is experimented by creating two training dataset and one test dataset of Nepali text documents of five classes. This chapter describes the datasets and data dictionaries used in the experiments and corresponding empirical results. Training and testing datasets are described the Section 5.1 and all other NLP data dictionaries used in the system are described in the Section 5.2. Experimentation results and graphical analysis are described in the Section 5.3.

5.1 Training and Testing Datasets

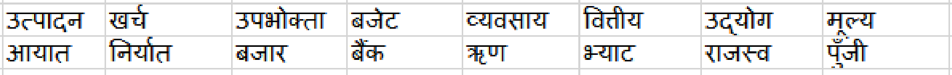
We have collected five classes of Nepali text documents for system evaluation. Documents are collected from various online sources, such as www.karobardaily.com, www.nagariknews.com, www.onlinekhaber.com, www.swasthyakhabar.com, and www.nepalhealthnews.com; and various offline sources, such as books, manuscripts, and articles.

1. **Business :** Business class of the text documents contains information about the trade of goods and/or services. Sample documents of business class are given in Figure 5.1. Sample Business dictionary which is used to evaluate the effectiveness of the business class is of the input document is given in the Figure 5.2.



1	स्थानीयस्तरमा उत्पादित माछालाई बजारसम्म पु¥याउन र उत्पादन वृद्धि गर्न आवश्यक रहेको मत्स्य विकास निर्देशनालयका कार्यक्रम निर्देशक रमानन्द मिश्रले बताए । "सरकारले माछामा लगानी बढाउँदै लगेको छ," उनले भने, "उत्पादक र व्यवसायीसमेत थप सक्रिय भएर लागे बजार विस्तार गर्न सजिलो हुन्छ ।" सरकारले माछा उत्पादन गर्न चाहनेलाई मिसन फिस कार्यक्रममार्फत अनुदान सहयोगसमेत गरेको उनले बताए ।
2	महोत्सवको आयोजना नेपाल मत्स्य व्यसायी संघ र नेपाल फिसरिस सोसाइटीले गरेका हुन् । मत्स्य विकास निर्देशनालयअन्तर्गतको मत्स्य विकास कार्यक्रम, कृषि व्यवसायी प्रवद्र्धन कार्यक्रम, कृषि अनुसन्धान परिषद् (नार्क), कृषि तथा वन विश्वविद्यालय, त्रिभुवन विश्वविद्यालय र नेपाल उद्योग वाणिज्य महासंघको कृषि उद्यम केन्द्र सहआयोजक थिए ।
3	बुटवल फर्निचर उद्योग संघको शनिबार बुटवलमा सम्पन्न सातौं अधिवेशनका अवसरमा बोल्दै उद्योगीहरुले विना दर्ता संचालित उद्योगहरुका कारण एकातिर राज्यको कर गुमेको र अर्कोतिर अस्वस्थ प्रतिस्पर्धा बढेको गुनासो समेत गरे ।
4	नेपाली व्यापारीहरुले काठमाण्डौंबाट ल्याएको चाइनिज सामानका मुख्य ग्राहक भारतीय नै भएको कञ्चनपुर उद्योग वाणिज्य संघका निवर्तमान उपाध्यक्ष जनकराज भट्ट बताउँछन् । सिद्धनाथ इम्पेक्सका संचालक रहेका उनी भन्छन् "भारतीय पर्यटकले प्रायः चिनियाँ सामान मन पराउँछन् ।" अरु बेलाभन्दा हिउँदमा भारतीय पर्यटक बढी आउने गरेको उनी बताउँछन् । "भारतीय ग्राहकले चाइनिज ज्याकेट, ट्राउजर, झोला, जुत्ता, मखमली कपडा, कम्बल, सल, कस्मेटिक तथा इलेक्ट्रोनिक सामान किन्ने गरेका छन् ।" उनले भने "धनगढी र महेन्द्रनगरका रहेका फेन्सी पसलका ५० प्रतिशत सामानका उपभोक्ता भारतीय पर्यटक नै हुन् ।"
5	भारतको केन्द्रिय राजधानी दिल्लीसहित हरियाणा, पञ्जाब, राजस्थान, उत्तरप्रदेश र उत्तराखण्डबाट घुम्न आउनेहरुलाई लक्ष्य गरेर यहाँ पसल र होटेल खुल्ने क्रम पनि बढेको छ । अधिकांश भारतीय पर्यटकहरु परिवारसाथ आफ्नै वाहनमा आउने गरेका छन् । भारतीयहरु आगमनको रेकर्ड राख्ने चलन नभएकाले उनीहरुको यकिन तथ्यांक भने कुनै निकायसँग छैन ।भारतीयहरु कञ्चनपुर र कैलालीका अधिकांश पर्यटकीयस्थलको भ्रमण गरेर फर्कने गरेका छन् भने केही पहाडी जिल्लासम्म पनि पुग्ने गरेका छन् ।

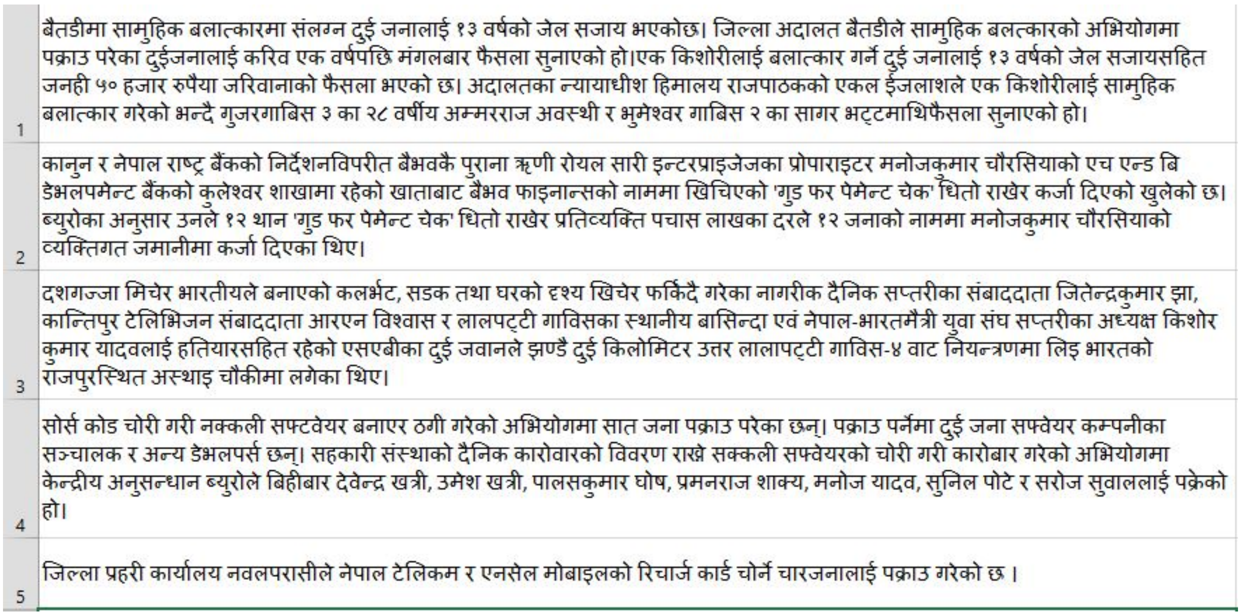
Figure 5.1: Sample Business Documents.



उत्पादन	खर्च	उपभोक्ता	बजेट	व्यवसाय	वितीय	उद्योग	मूल्य
आयात	निर्यात	बजार	बैंक	ऋण	भ्याट	राजस्व	पूँजी

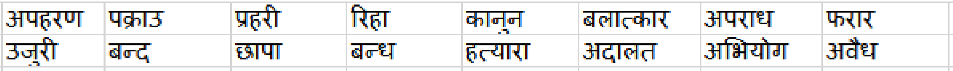
Figure 5.2: Sample Business Dictionary.

2. **Crime :** Crime class of documents contains information about unlawful acts or punishable acts. Criminal acts may include murder, rape, theft etc. Sample documents of crime class are given in Figure 5.3. Sample Crime dictionary which is used to evaluate the effectiveness of the crime class is of the input document is given in the Figure 5.4.



1	बैतडीमा सामुहिक बलात्कारमा संलग्न दुई जनालाई १३ वर्षको जेल सजाय भएकोछ। जिल्ला अदालत बैतडीले सामुहिक बलत्कारको अभियोगमा पक्राउ परेका दुईजनालाई करिव एक वर्षपछि मंगलबार फैसला सुनाएको हो।एक किशोरीलाई बलात्कार गर्ने दुई जनालाई १३ वर्षको जेल सजायसहित जनही ५० हजार रुपैया जरिवानाको फैसला भएको छ। अदालतका न्यायाधीश हिमालय राजपाठकको एकल ईजलाशले एक किशोरीलाई सामुहिक बलात्कार गरेको भन्दै गुजरगाबिस ३ का २८ वर्षीय अम्मरराज अवस्थी र भुमेश्वर गाबिस २ का सागर भट्टमाथिफैसला सुनाएको हो।
2	कानुन र नेपाल राष्ट्र बैंकको निर्देशनविपरीत बैभवकै पुराना ऋणी रोयल सारी इन्टरप्राइजेजका प्रोपाराइटर मनोजकुमार चौरसियाको एच एन्ड बि डेभलपमेन्ट बैंकको कुलेश्वर शाखामा रहेको खाताबाट बैभव फाइनान्सको नाममा खिचिएको 'गुड फर पेमेन्ट चेक' धितो राखेर कर्जा दिएको खुलेको छ। ब्युरोका अनुसार उनले १२ थान 'गुड फर पेमेन्ट चेक' धितो राखेर प्रतिव्यक्ति पचास लाखका दरले १२ जनाको नाममा मनोजकुमार चौरसियाको व्यक्तिगत जमानीमा कर्जा दिएका थिए।
3	दशगज्जा मिचेर भारतीयले बनाएको कलर्भट, सडक तथा घरको दृश्य खिचेर फर्किदै गरेका नागरीक दैनिक सप्तरीका संबाददाता जितेन्द्रकुमार झा, कान्तिपुर टेलिभिजन संबाददाता आरएन विश्वास र लालपट्टी गाविसका स्थानीय बासिन्दा एवं नेपाल-भारतमैत्री युवा संघ सप्तरीका अध्यक्ष किशोर कुमार यादवलाई हतियारसहित रहेको एसएबीका दुई जवानले झण्डै दुई किलोमिटर उत्तर लालपट्टी गाविस-४ वाट नियन्त्रणमा लिइ भारतको राजपुरस्थित अस्थाइ चौकीमा लगेका थिए।
4	सोर्स कोड चोरी गरी नक्कली सफ्टवेयर बनाएर ठगी गरेको अभियोगमा सात जना पक्राउ परेका छन्। पक्राउ पर्नेमा दुई जना सफ्वेयर कम्पनीका सञ्चालक र अन्य डेभलपर्स छन्। सहकारी संस्थाको दैनिक कारोवारको विवरण राख्ने सक्कली सफ्वेयरको चोरी गरी कारोबार गरेको अभियोगमा केन्द्रीय अनुसन्धान ब्युरोले बिहीबार देवेन्द्र खत्री, उमेश खत्री, पालसकुमार घोष, प्रमनराज शाक्य, मनोज यादव, सुनिल पोटे र सरोज सुवाललाई पक्रेको हो।
5	जिल्ला प्रहरी कार्यालय नवलपरासीले नेपाल टेलिकम र एनसेल मोबाइलको रिचार्ज कार्ड चोर्ने चारजनालाई पक्राउ गरेको छ ।

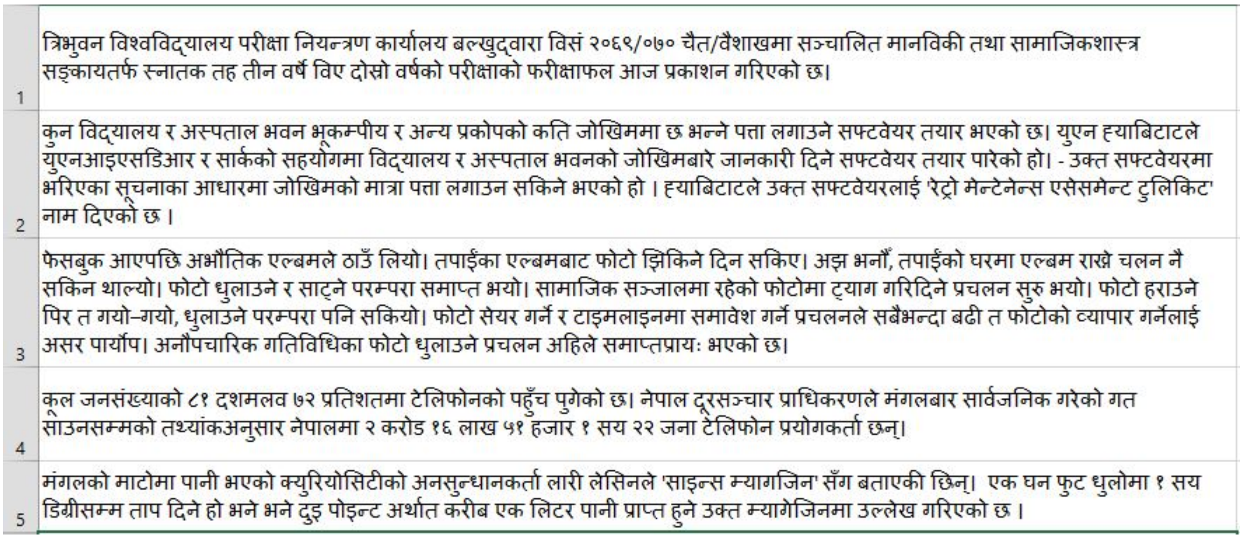
Figure 5.3: Sample Crime Documents.



अपहरण	पक्राउ	प्रहरी	रिहा	कानून	बलात्कार	अपराध	फरार
उजुरी	बन्द	छापा	बन्ध	हत्यारा	अदालत	अभियोग	अवैध

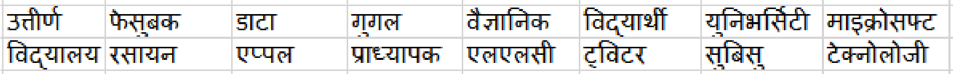
Figure 5.4: Sample Crime Dictionary.

3. **Education :** Education class contains all the documents related to academic stuffs which contains knowledge, skills, and habits and teaching, training, or research strategies. Sample documents of education class are given in Figure 5.5. Sample Education dictionary which is used to evaluate the effectiveness of the education class is of the input document is given in the Figure 5.6.



1	त्रिभुवन विश्वविद्यालय परीक्षा नियन्त्रण कार्यालय बल्खुद्वारा विसं २०६९/०७० चैत/वैशाखमा सञ्चालित मानविकी तथा सामाजिकशास्त्र सङ्कायतर्फ स्नातक तह तीन वर्षे विए दोस्रो वर्षको परीक्षाको फरीक्षाफल आज प्रकाशन गरिएको छ।
2	कुन विद्यालय र अस्पताल भवन भूकम्पीय र अन्य प्रकोपको कति जोखिममा छ भन्ने पत्ता लगाउने सफ्टवेयर तयार भएको छ। युएन ह्याबिटाटले युएनआइएसडिआर र सार्कको सहयोगमा विद्यालय र अस्पताल भवनको जोखिमबारे जानकारी दिने सफ्टवेयर तयार पारेको हो। - उक्त सफ्टवेयरमा भरिएका सूचनाका आधारमा जोखिमको मात्रा पत्ता लगाउन सकिने भएको हो । ह्याबिटाटले उक्त सफ्टवेयरलाई 'रेट्रो मेन्टेनेन्स एसेसमेन्ट टुलिकिट' नाम दिएको छ ।
3	फेसबुक आएपछि अभौतिक एल्बमले ठाउँ लियो। तपाईंका एल्बमबाट फोटो झिकिने दिन सकिए। अझ भनौं, तपाईंको घरमा एल्बम राख्ने चलन नै सकिन थाल्यो। फोटो धुलाउने र साट्ने परम्परा समाप्त भयो। सामाजिक सञ्जालमा रहेको फोटोमा ट्याग गरिदिने प्रचलन सुरु भयो। फोटो हराउने पिर त गयो–गयो, धुलाउने परम्परा पनि सकियो। फोटो सेयर गर्ने र टाइमलाइनमा समावेश गर्ने प्रचलनले सबैभन्दा बढी त फोटोको व्यापार गर्नेलाई असर पार्योप। अनौपचारिक गतिविधिका फोटो धुलाउने प्रचलन अहिले समाप्तप्राय: भएको छ।
4	कूल जनसंख्याको ८१ दशमलव ७२ प्रतिशतमा टेलिफोनको पहुँच पुगेको छ। नेपाल दूरसञ्चार प्राधिकरणले मंगलबार सार्वजनिक गरेको गत साउनसम्मको तथ्यांकअनुसार नेपालमा २ करोड १६ लाख ५१ हजार १ सय २२ जना टेलिफोन प्रयोगकर्ता छन्।
5	मंगलको माटोमा पानी भएको क्युरियोसिटीको अनसुन्धानकर्ता लारी लेसिनले 'साइन्स म्यागजिन' सँग बताएकी छिन्। एक घन फुट धुलोमा १ सय डिग्रीसम्म ताप दिने हो भने भने दुइ पोइन्ट अर्थात करीब एक लिटर पानी प्राप्त हुने उक्त म्यागेजिनमा उल्लेख गरिएको छ ।

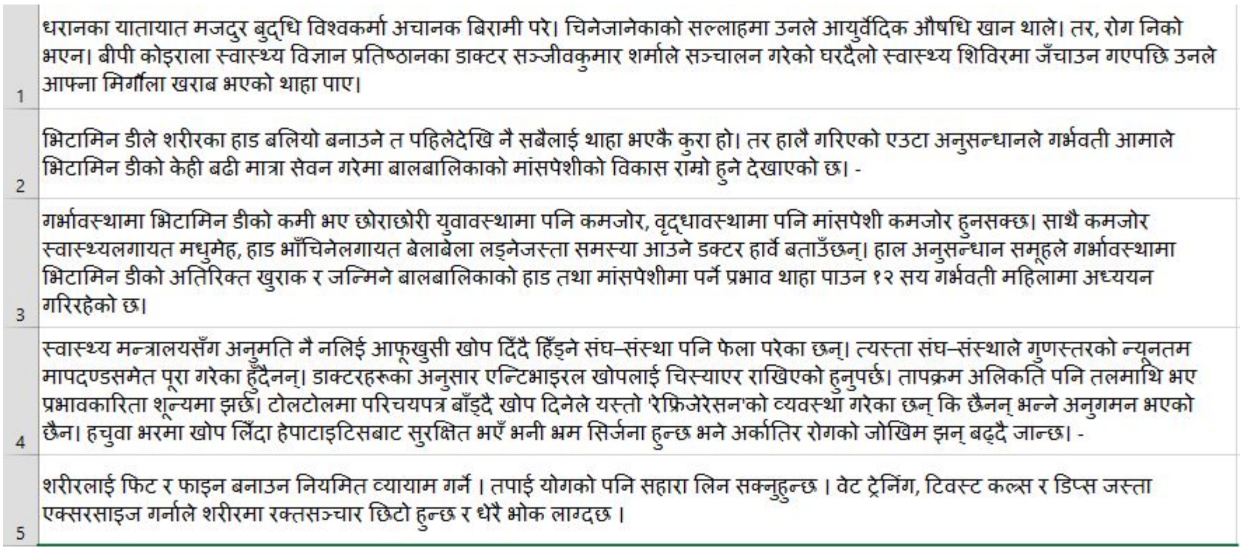
Figure 5.5: Sample Education Documents.



उत्तीर्ण	फेसुबक	डाटा	गुगल	वैज्ञानिक	विद्यार्थी	युनिभर्सिटी	माइक्रोसफ्ट
विद्यालय	रसायन	एप्पल	प्राध्यापक	एलएलसी	ट्विटर	सुबिस्	टेक्नोलोजी

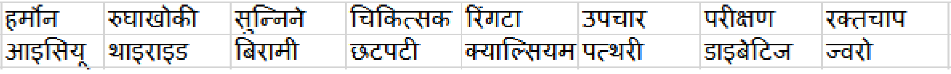
Figure 5.6: Sample Education Dictionary.

4. **Health :** Health class of dataset contains all the documents that are related to illness, injury, pain and diagnostics. It also contains documents related to nutrition, health care and health educations. Sample documents of health class are given in Figure 5.7. Sample health dictionary which is used to evaluate the effectiveness of the health class is of the input document is given in the Figure 5.8.



1	धरानका यातायात मजदुर बुद्धि विश्वकर्मा अचानक बिरामी परे। चिनेजानेकाको सल्लाहमा उनले आयुर्वेदिक औषधि खान थाले। तर, रोग निको भएन। बीपी कोइराला स्वास्थ्य विज्ञान प्रतिष्ठानका डाक्टर सञ्जीवकुमार शर्माले सञ्चालन गरेको घरदैलो स्वास्थ्य शिविरमा जँचाउन गएपछि उनले आफ्ना मिर्गौला खराब भएको थाहा पाए।
2	भिटामिन डीले शरीरका हाड बलियो बनाउने त पहिलेदेखि नै सबैलाई थाहा भएकै कुरा हो। तर हालै गरिएको एउटा अनुसन्धानले गर्भवती आमाले भिटामिन डीको केही बढी मात्रा सेवन गरेमा बालबालिकाको मांसपेशीको विकास राम्रो हुने देखाएको छ। -
3	गर्भावस्थामा भिटामिन डीको कमी भए छोराछोरी युवावस्थामा पनि कमजोर, वृद्धावस्थामा पनि मांसपेशी कमजोर हुनसक्छ। साथै कमजोर स्वास्थ्यलगायत मधुमेह, हाड भाँचिनेलगायत बेलाबेला लड्नेजस्ता समस्या आउने डक्टर हार्वे बताउँछन्। हाल अनुसन्धान समूहले गर्भावस्थामा भिटामिन डीको अतिरिक्त खुराक र जन्मिने बालबालिकाको हाड तथा मांसपेशीमा पर्ने प्रभाव थाहा पाउन १२ सय गर्भवती महिलामा अध्ययन गरिरहेको छ।
4	स्वास्थ्य मन्त्रालयसँग अनुमति नै नलिई आफूखुसी खोप दिँदै हिँड्ने संघ–संस्था पनि फेला परेका छन्। त्यस्ता संघ–संस्थाले गुणस्तरको न्यूनतम मापदण्डसमेत पूरा गरेका हुँदैनन्। डाक्टरहरूका अनुसार एन्टिभाइरल खोपलाई चिस्याएर राखिएको हुनुपर्छ। तापक्रम अलिकति पनि तलमाथि भए प्रभावकारिता शून्यमा झर्छ। टोलटोलमा परिचयपत्र बाँड्दै खोप दिनेले यस्तो 'रेफ्रिजेरेसन'को व्यवस्था गरेका छन् कि छैनन् भन्ने अनुगमन भएको छैन। हचुवा भरमा खोप लिँदा हेपाटाइटिसबाट सुरक्षित भएँ भनी भ्रम सिर्जना हुन्छ भने अर्कातिर रोगको जोखिम झन् बढ्दै जान्छ। -
5	शरीरलाई फिट र फाइन बनाउन नियमित व्यायाम गर्ने । तपाई योगको पनि सहारा लिन सक्नुहुन्छ । वेट ट्रेनिंग, टिवस्ट कल्स र डिप्स जस्ता एक्सरसाइज गर्नाले शरीरमा रक्तसञ्चार छिटो हुन्छ र धेरै भोक लाग्दछ ।

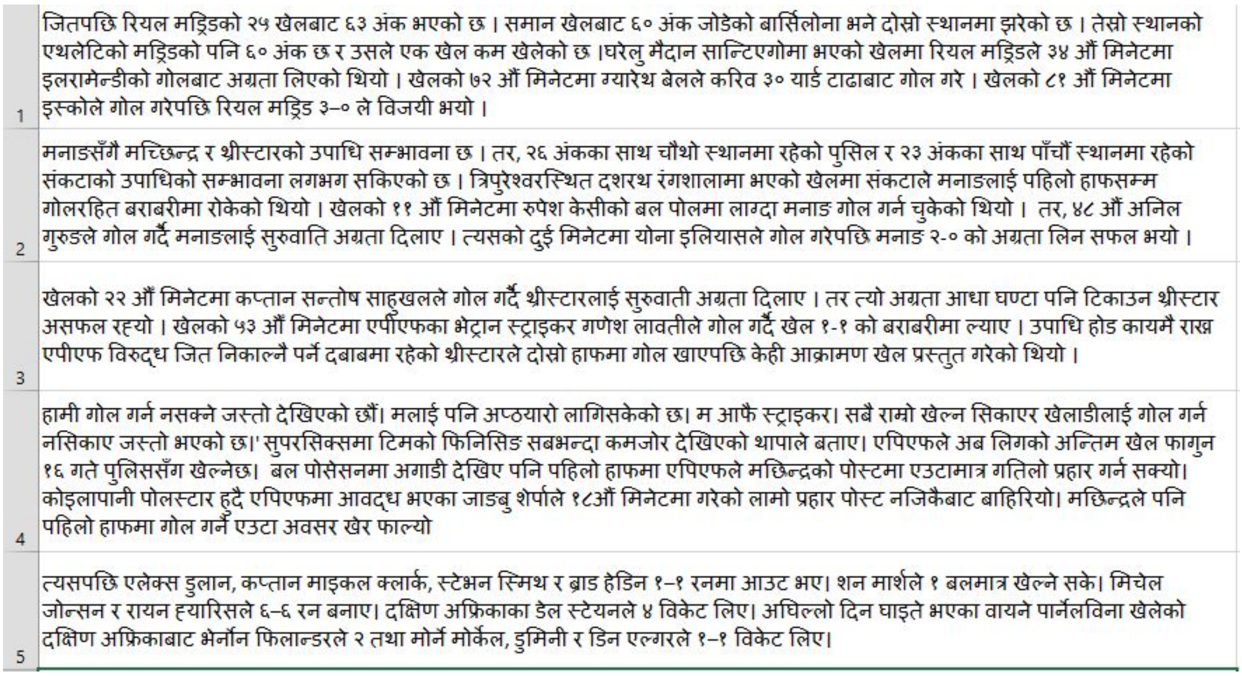
Figure 5.7: Sample Health Documents.



हर्मोन	रुघाखोकी	सुन्निने	चिकित्सक	रिंगटा	उपचार	परीक्षण	रक्तचाप
आइसियू	थाइराइड	बिरामी	छटपटी	क्याल्सियम	पत्थरी	डाइबेटिज	ज्वरो

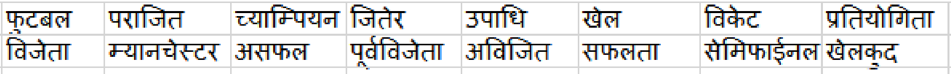
Figure 5.8: Sample Health Dictionary.

5. **Sports :** Sports class of dataset contains the documents related to sports. It includes sport competitions, tournaments, and other related news and events. Sample documents of sports class are given in Figure 5.9. Sample Sport dictionary which is used to evaluate the effectiveness of the sport class is of the input document is given in the Figure 5.10.



1	जितपछि रियल मड्रिडको २५ खेलबाट ६३ अंक भएको छ । समान खेलबाट ६० अंक जोडेको बार्सिलोना भने दोस्रो स्थानमा झरेको छ । तेस्रो स्थानको एथलेटिको मड्रिडको पनि ६० अंक छ र उसले एक खेल कम खेलेको छ ।घरेलु मैदान सान्टिएगोमा भएको खेलमा रियल मड्रिडले ३४ औं मिनेटमा इलरामेन्डीको गोलबाट अग्रता लिएको थियो । खेलको ७२ औं मिनेटमा ग्यारेथ बेलले करिव ३० यार्ड टाढाबाट गोल गरे । खेलको ८१ औं मिनेटमा इस्कोले गोल गरेपछि रियल मड्रिड ३–० ले विजयी भयो ।
2	मनाङसँगै मच्छिन्द्र र थ्रीस्टारको उपाधि सम्भावना छ । तर, २६ अंकका साथ चौथो स्थानमा रहेको पुसिल र २३ अंकका साथ पाँचौं स्थानमा रहेको संकटाको उपाधिको सम्भावना लगभग सकिएको छ । त्रिपुरेश्वरस्थित दशरथ रंगशालामा भएको खेलमा संकटाले मनाङलाई पहिलो हाफसम्म गोलरहित बराबरीमा रोकेको थियो । खेलको ११ औं मिनेटमा रुपेश केसीको बल पोलमा लाग्दा मनाङ गोल गर्न चुकेको थियो । तर, ४८ औं अनिल गुरुङले गोल गर्दै मनाङलाई सुरुवाति अग्रता दिलाए । त्यसको दुई मिनेटमा योना इलियासले गोल गरेपछि मनाङ २-० को अग्रता लिन सफल भयो ।
3	खेलको २२ औं मिनेटमा कप्तान सन्तोष साहुखलले गोल गर्दै थ्रीस्टारलाई सुरुवाती अग्रता दिलाए । तर त्यो अग्रता आधा घण्टा पनि टिकाउन थ्रीस्टार असफल रहयो । खेलको ५३ औं मिनेटमा एपीएफका भेट्रान स्ट्राइकर गणेश लावतीले गोल गर्दै खेल १-१ को बराबरीमा ल्याए । उपाधि होड कायमै राख्न एपीएफ विरुद्ध जित निकाल्नै पर्ने दबाबमा रहेको थ्रीस्टारले दोस्रो हाफमा गोल खाएपछि केही आक्रामण खेल प्रस्तुत गरेको थियो ।
4	हामी गोल गर्न नसक्ने जस्तो देखिएको छौं। मलाई पनि अप्ठयारो लागिसकेको छ। म आफै स्ट्राइकर। सबै राम्रो खेल्न सिकाएर खेलाडीलाई गोल गर्न नसिकाए जस्तो भएको छ।' सुपरसिक्समा टिमको फिनिसिङ सबभन्दा कमजोर देखिएको थापाले बताए। एपिएफले अब लिगको अन्तिम खेल फागुन १६ गते पुलिससँग खेल्नेछ। बल पोसेसनमा अगाडी देखिए पनि पहिलो हाफमा एपिएफले मछिन्द्रको पोस्टमा एउटामात्र गतिलो प्रहार गर्न सक्यो। कोइलापानी पोलस्टार हुदै एपिएफमा आवद्ध भएका जाङबु शेर्पाले १८औं मिनेटमा गरेको लामो प्रहार पोस्ट नजिकैबाट बाहिरियो। मछिन्द्रले पनि पहिलो हाफमा गोल गर्ने एउटा अवसर खेर फाल्यो
5	त्यसपछि एलेक्स डुलान, कप्तान माइकल क्लार्क, स्टेभन स्मिथ र ब्राड हेडिन १–१ रनमा आउट भए। शन मार्शले १ बलमात्र खेल्ने सके। मिचेल जोन्सन र रायन ह्यारिसले ६–६ रन बनाए। दक्षिण अफ्रिकाका डेल स्टेयनले ४ विकेट लिए। अघिल्लो दिन घाइते भएका वायने पार्नेलविना खेलेको दक्षिण अफ्रिकाबाट भेर्नोन फिलान्डरले २ तथा मोर्ने मोर्केल, डुमिनी र डिन एल्गरले १–१ विकेट लिए।

Figure 5.9: Sample Sports Documents.



फुटबल	पराजित	च्याम्पियन	जितेर	उपाधि	खेल	विकेट	प्रतियोगिता
विजेता	म्यानचेस्टर	असफल	पूर्वविजेता	अविजित	सफलता	सेमिफाईनल	खेलकुद

Figure 5.10: Sample Sports Dictionary.

5.1.1 Training Datasets

5.1.1.1 Dataset I

Statistics about dataset I are given in the Table 5.1

Table 5.1: Dataset I

Classes	No. of samples
Business	121
Crime	73
Education	125
Health	135
Sports	171
Total	**625**

5.1.1.2 Dataset II

Statistics about dataset II are given in the Table 5.2

Table 5.2: Dataset II

Classes	No. of samples
Business	122
Crime	74
Education	125
Health	135
Sports	172
Total	**628**

5.1.2 Testing Dataset

Statistics about test dataset is given in the Table 5.3

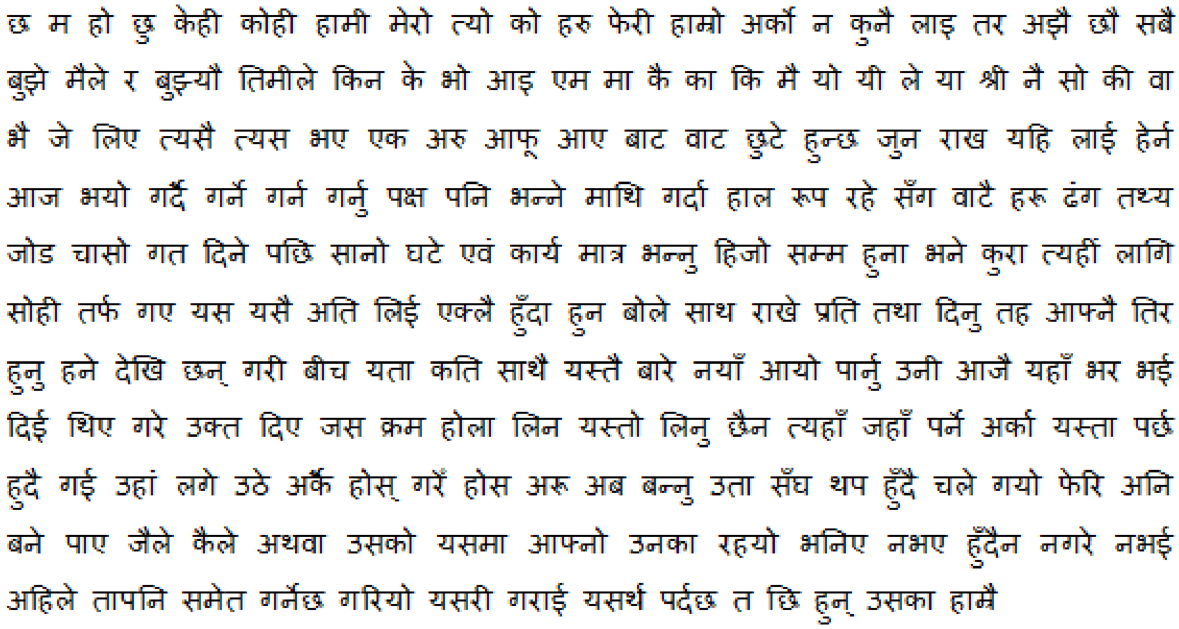
Table 5.3: Test Dataset

Classes	No. of samples
Business	19
Crime	20
Education	12
Health	19
Sports	19
Total	**89**

5.2 Data Dictionaries

5.2.1 Stop Word Dictionary

Stop word dictionary contains all the common and less informative words. Un-useful words are removed from the document in pre-processing stage to make feature space smaller. The words in the document that matched with the words listed in the stop word dictionary are excluded. Some of the stop words from the stop word dictionary are given in the Figure 5.11.

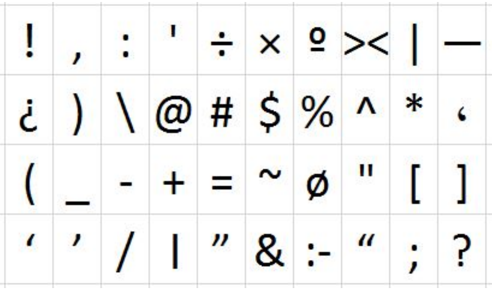


छ म हो छु केही कोही हामी मेरो त्यो को हरु फेरी हाम्रो अर्को न कुनै लाइ तर अझै छौ सबै
बुझे मैले र बुझ्यौ तिमीले किन के भो आइ एम मा कै का कि मै यो यी ले या श्री नै सो की वा
भै जे लिए त्यसै त्यस भए एक अरु आफू आए बाट वाट छुटे हुन्छ जुन राख यहि लाई हेर्न
आज भयो गर्दै गर्ने गर्न गर्नु पक्ष पनि भन्ने माथि गर्दा हाल रूप रहे सँग वाटै हरू ढंग तथ्य
जोड चासो गत दिने पछि सानो घटे एवं कार्य मात्र भन्नु हिजो सम्म हुना भने कुरा त्यहीं लागि
सोही तर्फ गए यस यसै अति लिई एक्लै हुँदा हुन बोले साथ राखे प्रति तथा दिनु तह आफ्नै तिर
हुनु हने देखि छन् गरी बीच यता कति साथै यस्तै बारे नयाँ आयो पार्नु उनी आजै यहाँ भर भई
दिई थिए गरे उक्त दिए जस क्रम होला लिन यस्तो लिनु छैन त्यहाँ जहाँ पर्ने अर्को यस्ता पर्छ
हुदै गई उहां लगे उठे अर्कै होस् गरें होस अरू अब बन्नु उता सँघ थप हुँदै चले गयो फेरि अनि
बने पाए जैले कैले अथवा उसको यसमा आफ्नो उनका रहयो भनिए नभए हुँदैन नगरे नभई
अहिले तापनि समेत गर्नेछ गरियो यसरी गराई यसर्थ पर्दछ त छि हुन् उसका हाम्रै

Figure 5.11: Stop Word Dictionary.

5.2.2 Symbol Dictionary

The symbols that don't carry the special meanings are removed in the pre-processing stage. Some of the special symbols are given in Figure 5.12.



!	,	:	'	÷	×	º	><	|	—
¿	)	\	@	#	$	%	^	*	‘
(	_	-	+	=	~	ø	"	[	]
‘	’	/	|	”	&	:-	“	;	?

Figure 5.12: Symbol Dictionary.

5.3 Experimentation Results

System is trained and tested against collected datasets described in the Section 5.1. Various performance matrices (Section 3.10) are evaluated. This section describes all the empirical results and analysis of the outcomes.

5.3.1 Experiment 1

First experiment is carried in Training Dataset I (Section 5.1.1.1) and Testing Dataset (Section 5.1.2). Experimentation results shows Naive Bayes classifier performs better than Neural Network based classifier. Table 5.4 shows results of the experiment 1. Figure 5.13 shows graphical representation of the results. Table 5.5 show confusion matrix corresponding to Neural Network classifier and Table 5.6 show confusion matrix corresponding to Naive Bayes classifier.

Table 5.4: Experimentation Results (Experiment 1)

Algorithm	Avg. Sys. Acc. (%)	Err (%)	Precision (%)	Recall (%)	F-Score (%)
MLP	85.92	14.08	77.56	92.12	84.63
Naïve Bayes	86.17	13.83	77.75	93.51	84.90

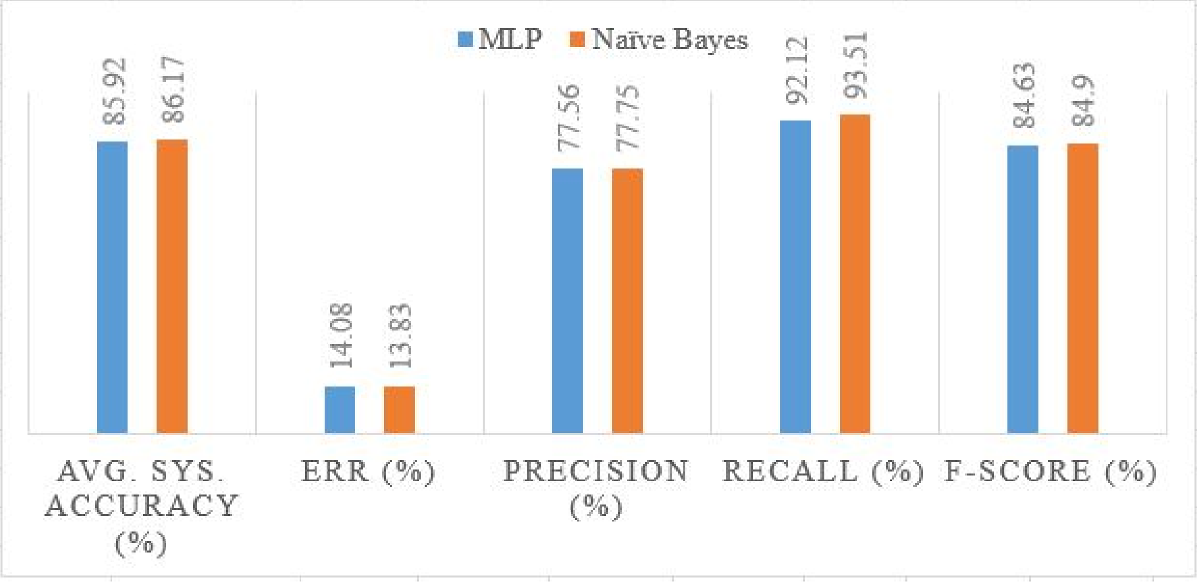


Figure 5.13: Graph of Experiment 1.

Table 5.5: Confusion Matrix (Experiment 1 - Neural Network)

Class	Business	Crime	Education	Health	Sport
Business	13	0	5	1	0
Crime	6	10	1	3	0
Education	1	1	9	1	0
Health	1	0	0	18	0
Sport	0	0	0	1	18

Table 5.6: Confusion Matrix (Experiment 1 - Naive Bayes)

Class	Business	Crime	Education	Health	Sport
Business	9	1	8	0	1
Crime	3	13	2	2	0
Education	1	1	10	0	0
Health	0	0	0	19	0
Sport	0	0	1	0	18

5.3.2 Experiment 2

First experiment is carried in Training Dataset II (Section 5.1.1.2) and Testing Dataset (Section 5.1.2). Experimentation results shows Neural Network based classifier performs better than Naive Bayes classifier. Table 5.7 shows results of the experiment 2. Figure 5.14 shows graphical representation of the results. Table 5.8 show confusion matrix corresponding to Neural Network classifier and Table 5.9 show confusion matrix corresponding to Naive Bayes classifier.

Table 5.7: Experimentation Results (Experiment 2)

Algorithm	Avg. Sys. Acc. (%)	Err (%)	Precision (%)	Recall (%)	F-Score (%)
MLP	89.19	10.81	83.02	94.70	88.48
Naïve Bayes	88.02	11.98	80.99	94.24	87.11

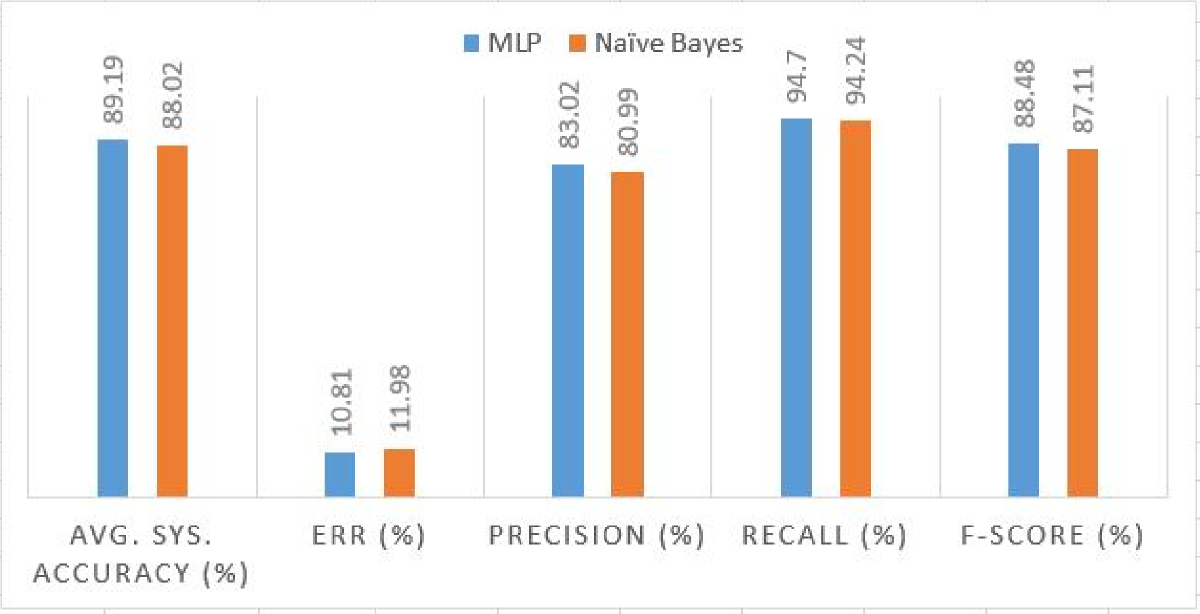


Figure 5.14: Graph of Experiment 2.

Table 5.8: Confusion Matrix (Experiment 2 - Neural Network)

Class	Business	Crime	Education	Health	Sport
Business	13	0	2	3	1
Crime	0	14	1	5	0
Education	1	0	9	2	0
Health	0	0	2	17	0
Sport	0	0	0	0	19

Table 5.9: Confusion Matrix (Experiment 2 - Naive Bayes)

Class	Business	Crime	Education	Health	Sport
Business	14	3	1	0	1
Crime	1	14	4	1	0
Education	1	0	11	0	0
Health	1	0	5	13	0
Sport	0	0	0	0	19

5.4 Result Analysis

Aggregate results of both the experiments are shown in Table 5.10.

Table 5.10: Aggregate System Results

Algorithm	Avg. Sys. Acc. (%)	Err (%)	Precision (%)	Recall (%)	F-Score (%)
MLP	87.55	12.44	80.29	93.41	86.55
Naïve Bayes	87.09	12.90	79.37	93.87	86.05

Results shows Neural Network based classifier has slight less error rates than Naive Bayes based classifier.

System results greatly influenced by number of training and testing data and extracted features. Classifier parameters also play important roles for better learning of the system. Computational and efficiency effectiveness can be enhanced by code optimization and distributed computing.

Chapter 6

CONCLUSION

6.1 Conclusion

An automatic multi class text classification problem for Nepali language is addressed in this dissertation work. As the solution of the stated problem, two machine learning based classification techniques are experimented and performance is measured for both the cases.

Classification systems take input a unknown text document and assign to a known class among five classes (”Business”, ”Crime”, ”Education”, ”Health”,”Sport”). Input text document is passed through various pre-processing steps like stop-word removal, symbol removal and stemming. Then, fine grained document is passed into feature extractor, where term frequency based features are extracted. Feature vector is than fed to classification systems-which are previously trained with given datasets and given classes in supervised manner.

Empirical results shows, Neural Network based classifier (MLP) performs better than Naive Bayes based classifier. MLP classification system has the average system accuracy rate of 87.55%, system error rate of 12.44%, precision rate of 80.29% recall rate of 93.41% and f-score rate of 86.55%. Similarly, Naive Bayes classification system has the average system accuracy rate of 87.09%, system error rate of 12.90%, precision rate of 79.37% recall rate of 93.87% and f-score rate of 86.05%.

6.2 Limitations and Future Scope

The performance of the proposed system may further be improved by improving pre-processing techniques. Exploring more features and enhancing data dictionaries can improve classification accuracy.

System performance is greatly influenced by training and testing corpus. Classifier parameters also play important roles for better learning of the system. Hence,accuracy can be enhanced by code optimization and distributed computing.

Due to the unavailability of standard training and test datasets, system performance can not be generalized well. To conclude, results are promising and can be enhanced further.

References

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[2] M. K. Dalal and M. A. Zaveri, "Automatic text classification: A technical review," no. 2, 2011.

[3] P. Jackson and I. Moulinier, *Natural Language Processing for Online Applications: Text Retrieval, Extraction & Categorization*. John Benjamins, 2002.

[4] S. M. Kamruzzaman and C. M. Rahman, "Text categorization using association rule and naive bayes classifier," *CoRR*, vol. abs/1009.4994, 2010.

[5] E. D. Wiener, "A neural network approach to topic spotting in text," Master's thesis, University of Colorado, 1995.

[6] N. Remeikis, I. Skucas, and V. Melninkaite, "Text categorization using neural networks initialized with decision trees," *Informatica, Lith. Acad. Sci*, vol. 15, no. 4, pp. 551–564, 2004.

[7] R. Bekkerman, R. El-Yaniv, N. Tishby, , and Y. Winter, "Distributional word clusters vs. words for text categorization," Mar. 2003.

[8] F. Aiolli, R. Cardin, F. S. 0001, and A. Sperduti, "Preferential text classification: Learning algorithms and evaluation measures," *ERCIM News*, vol. 2009, no. 76, 2009.

[9] T. Ayodele, S. Zhou, and R. Khusainov, "Email classification: Solution with back propagation technique," in *ICITST*, pp. 1–6, IEEE, 2009.

[10] S.Ramasundaram and S.P.Victor, "Text categorization by backpropagation network," *International Journal of Computer Applications*, vol. 8, pp. 1–5, October 2010. Published By Foundation of Computer Science.

[11] T. Jo, "Neural network for text categorization," *International Journal of Information Studies*, vol. 2, April 2010.

[12] R. N. R.Parimala, "A study on analysis of sms classification using document frequency threshold," vol. 4, no. 1, 2012.

[13] M. El, K. Amine, and B. T. eddine Rachidi, "Automatic arabic document categorization based on the naive bayes algorithm," aug 14 2008.

[14] W. Hazimah and B. W. Ismail, "Text categorization using naïve bayes algorithm," 2005.

[15] F. Peng, X. Huang, D. Schuurmans, and S. Wang, "Text classification in asian languages without word segmentation," in *IRAL* (J. Adachi, ed.), pp. 41–48, ACL, 2003.

[16] H. Li, L. Paull, Y. Biletskiy, and S. X. Yang, "Document classification using information theory and a fast back-propagation neural network," *Intelligent Automation and Soft Computing*, vol. 16, no. 1, pp. 25–38, 2010.

[17] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. University of Illinois at Urbana-Champaign, second ed., 2011.

[18] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," in *Knowledge Discovery and Data Mining*, vol. 2, pp. 121–167, 1998.

[19] S. Haykin, *Neural Networks: A Comprehensive Introduction*. Prentice Hall, 1999.

[20] G. Guo, H. Wang, D. Bell, and Y. Bi, "Knn model-based approach in classification," jul 08 2003.

[21] Bal Krishna Bal and Prajol Shrestha, "A morphological analyzer and a stemmer for nepali," Madan Puraskar Pustakalaya, Nepal, 2006.

[22] F. Sebastiani, "Text categorization," in *Text Mining and its Applications* (A. Zanasi, ed.), Southampton, UK: WIT Press, 2003. Invited chapter. Forthcoming.

[23] R. A. Calvo, M. Partridge, and M. A. Jabri, "A comparative study of principal component analysis techniques," 1998.

[24] Vikramkumar, V. B, and Trilochan, "Bayes and naive bayes classifier," apr 2014.

[25] "Wikipedia (www.wikipedia.com)."

[26] H. Zhang, "The optimality of naive bayes," in *FLAIRS Conference* (V. Barr and Z. Markov, eds.), pp. 562–567, AAAI Press, 2004.

[27] R. Caruana, A. Munson, and A. Niculescu-Mizil, "Getting the most out of ensemble selection," in *ICDM*, pp. 828–833, IEEE Computer Society, 2006.

[28] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," *CoRR*, vol. abs/1302.4964, 2013.

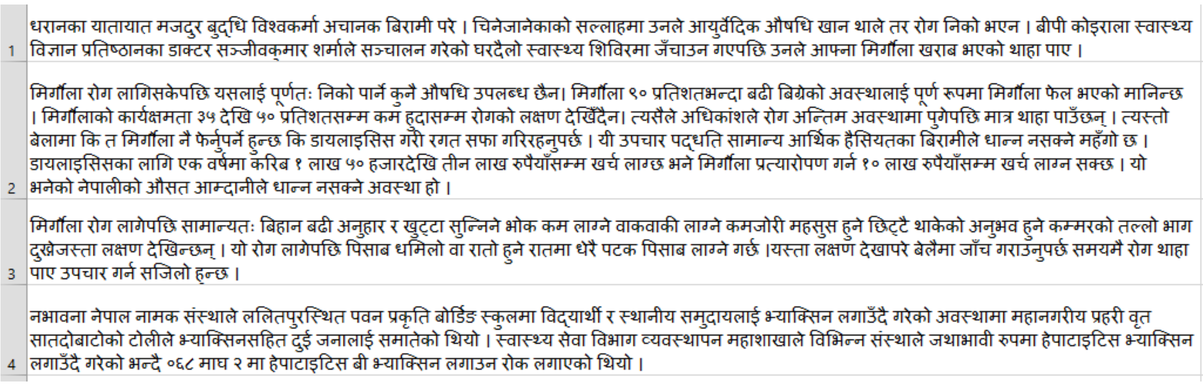
[29] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, pp. 989–993, Nov. 1994.

[30] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manage.*, vol. 45, pp. 427–437, jul 2009.

Appendix A

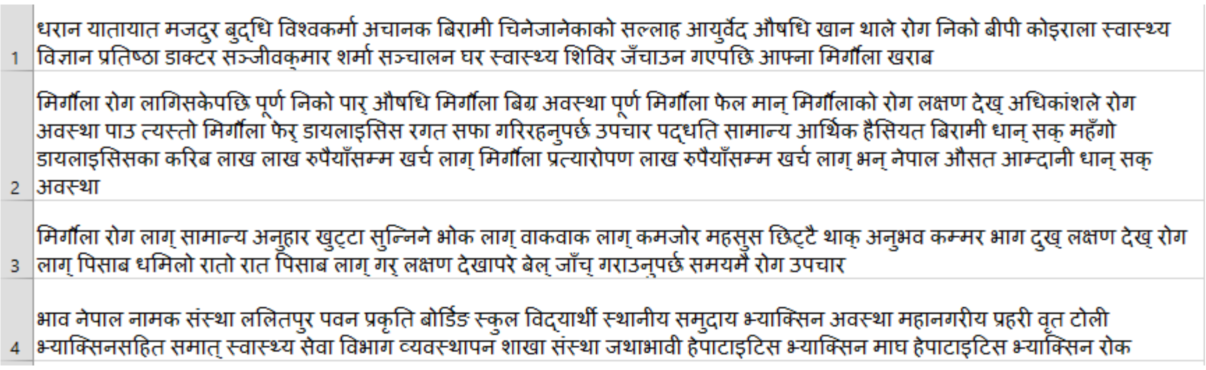
Sample Data

Sample Input



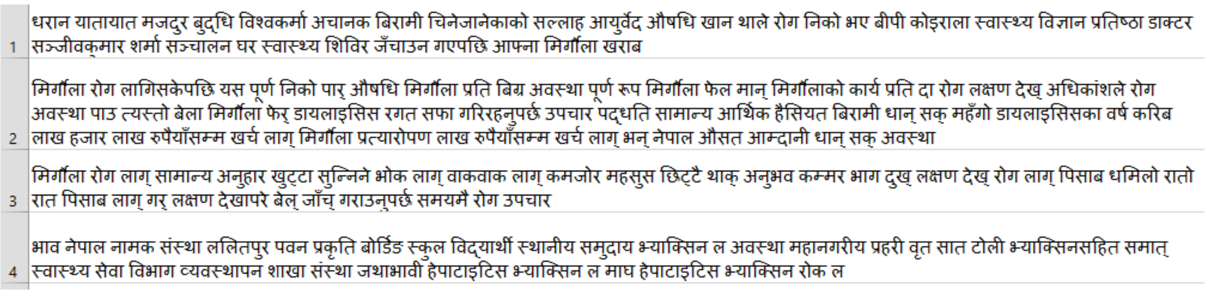
1	धरानका यातायात मजदुर बुद्धि विश्वकर्मा अचानक बिरामी परे । चिनेजानेकाको सल्लाहमा उनले आयुर्वेदिक औषधि खान थाले तर रोग निको भएन । बीपी कोइराला स्वास्थ्य विज्ञान प्रतिष्ठानका डाक्टर सञ्जीवकुमार शर्माले सञ्चालन गरेको घरदैलो स्वास्थ्य शिविरमा जँचाउन गएपछि उनले आफ्ना मिर्गौला खराब भएको थाहा पाए ।
2	मिर्गौला रोग लागिसकेपछि यसलाई पूर्णतः निको पार्ने कुनै औषधि उपलब्ध छैन। मिर्गौला ९० प्रतिशतभन्दा बढी बिग्रेको अवस्थालाई पूर्ण रूपमा मिर्गौला फेल भएको मानिन्छ । मिर्गौलाको कार्यक्षमता ३५ देखि ५० प्रतिशतसम्म कम हुदासम्म रोगको लक्षण देखिँदैन। त्यसैले अधिकांशले रोग अन्तिम अवस्थामा पुगेपछि मात्र थाहा पाउँछन् । त्यस्तो बेलामा कि त मिर्गौला नै फेर्नुपर्ने हुन्छ कि डायलाइसिस गरी रगत सफा गरिरहनुपर्छ । यी उपचार पद्धति सामान्य आर्थिक हैसियतका बिरामीले धान्न नसक्ने महँगो छ । डायलाइसिसका लागि एक वर्षमा करिब १ लाख ५० हजारदेखि तीन लाख रुपैयाँसम्म खर्च लाग्छ भने मिर्गौला प्रत्यारोपण गर्न १० लाख रुपैयाँसम्म खर्च लाग्न सक्छ । यो भनेको नेपालीको औसत आम्दानीले धान्न नसक्ने अवस्था हो ।
3	मिर्गौला रोग लागेपछि सामान्यतः बिहान बढी अनुहार र खुट्टा सुन्निने भोक कम लाग्ने वाकवाकी लाग्ने कमजोरी महसुस हुने छिट्टै थाकेको अनुभव हुने कम्मरको तल्लो भाग दुख्नेजस्ता लक्षण देखिन्छन् । यो रोग लागेपछि पिसाब धमिलो वा रातो हुने रातमा धेरै पटक पिसाब लाग्ने गर्छ ।यस्ता लक्षण देखापरे बेलैमा जाँच गराउनुपर्छ समयमै रोग थाहा पाए उपचार गर्न सजिलो हुन्छ ।
4	नभावना नेपाल नामक संस्थाले ललितपुरस्थित पवन प्रकृति बोर्डिङ स्कुलमा विद्यार्थी र स्थानीय समुदायलाई भ्याक्सिन लगाउँदै गरेको अवस्थामा महानगरीय प्रहरी वृत सातदोबाटोको टोलीले भ्याक्सिनसहित दुई जनालाई समातेको थियो । स्वास्थ्य सेवा विभाग व्यवस्थापन महाशाखाले विभिन्न संस्थाले जथाभावी रुपमा हेपाटाइटिस भ्याक्सिन लगाउँदै गरेको भन्दै ०६८ माघ २ मा हेपाटाइटिस बी भ्याक्सिन लगाउन रोक लगाएको थियो ।

Symbol and Digit Removed



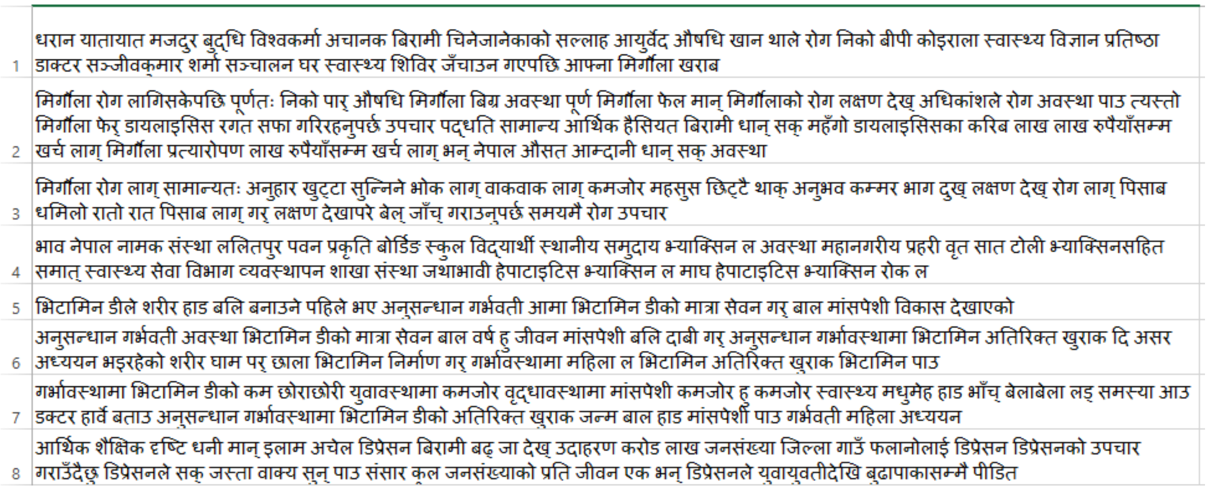
1	धरान यातायात मजदुर बुद्धि विश्वकर्मा अचानक बिरामी चिनेजानेकाको सल्लाह आयुर्वेद औषधि खान थाले रोग निको बीपी कोइराला स्वास्थ्य विज्ञान प्रतिष्ठा डाक्टर सञ्जीवकुमार शर्मा सञ्चालन घर स्वास्थ्य शिविर जँचाउन गएपछि आफ्ना मिर्गौला खराब
2	मिर्गौला रोग लागिसकेपछि पूर्ण निको पार् औषधि मिर्गौला बिग्र अवस्था पूर्ण मिर्गौला फेल मान् मिर्गौलाको रोग लक्षण देख् अधिकांशले रोग अवस्था पाउ त्यस्तो मिर्गौला फेर् डायलाइसिस रगत सफा गरिरहनुपर्छ उपचार पद्धति सामान्य आर्थिक हैसियत बिरामी धान् सक् महँगो डायलाइसिसका करिब लाख लाख रुपैयाँसम्म खर्च लाग् मिर्गौला प्रत्यारोपण लाख रुपैयाँसम्म खर्च लाग् भन् नेपाल औसत आम्दानी धान् सक् अवस्था
3	मिर्गौला रोग लाग् सामान्य अनुहार खुट्टा सुन्निने भोक लाग् वाकवाक लाग् कमजोर महसुस छिट्टै थाक् अनुभव कम्मर भाग दुख् लक्षण देख् रोग लाग् पिसाब धमिलो रातो रात पिसाब लाग् गर् लक्षण देखापरे बेल् जाँच् गराउनुपर्छ समयमै रोग उपचार
4	भाव नेपाल नामक संस्था ललितपुर पवन प्रकृति बोर्डिङ स्कुल विद्यार्थी स्थानीय समुदाय भ्याक्सिन अवस्था महानगरीय प्रहरी वृत टोली भ्याक्सिनसहित समात् स्वास्थ्य सेवा विभाग व्यवस्थापन शाखा संस्था जथाभावी हेपाटाइटिस भ्याक्सिन माघ हेपाटाइटिस भ्याक्सिन रोक

Stemming



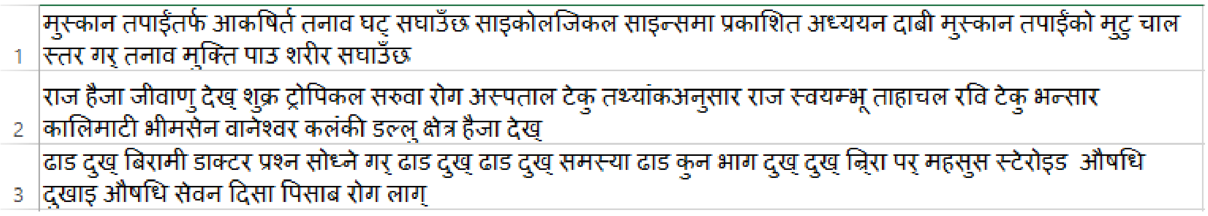
1	धरान यातायात मजदुर बुद्धि विश्वकर्मा अचानक बिरामी चिनेजानेकाको सल्लाह आयुर्वेद औषधि खान थाले रोग निको भए बीपी कोइराला स्वास्थ्य विज्ञान प्रतिष्ठा डाक्टर सञ्जीवकुमार शर्मा सञ्चालन घर स्वास्थ्य शिविर जँचाउन गएपछि आफ्ना मिर्गौला खराब
2	मिर्गौला रोग लागिसकेपछि यस पूर्ण निको पार् औषधि मिर्गौला प्रति बिग्र अवस्था पूर्ण रूप मिर्गौला फेल मान् मिर्गौलाको कार्य प्रति दा रोग लक्षण देख् अधिकांशले रोग अवस्था पाउ त्यस्तो बेला मिर्गौला फेर् डायलाइसिस रगत सफा गरिरहनुपर्छ उपचार पद्धति सामान्य आर्थिक हैसियत बिरामी धान् सक् महँगो डायलाइसिसका वर्ष करिब लाख हजार लाख रुपैयाँसम्म खर्च लाग् मिर्गौला प्रत्यारोपण लाख रुपैयाँसम्म खर्च लाग् भन् नेपाल औसत आम्दानी धान् सक् अवस्था
3	मिर्गौला रोग लाग् सामान्य अनुहार खुट्टा सुन्निने भोक लाग् वाकवाक लाग् कमजोर महसुस छिट्टै थाक् अनुभव कम्मर भाग दुख् लक्षण देख् रोग लाग् पिसाब धमिलो रातो रात पिसाब लाग् गर् लक्षण देखापरे बेल् जाँच् गराउनुपर्छ समयमै रोग उपचार
4	भाव नेपाल नामक संस्था ललितपुर पवन प्रकृति बोर्डिङ स्कुल विद्यार्थी स्थानीय समुदाय भ्याक्सिन ल अवस्था महानगरीय प्रहरी वृत सात टोली भ्याक्सिनसहित समात् स्वास्थ्य सेवा विभाग व्यवस्थापन शाखा संस्था जथाभावी हेपाटाइटिस भ्याक्सिन ल माघ हेपाटाइटिस भ्याक्सिन रोक ल

Training Sample



1	धरान यातायात मजदुर बुद्धि विश्वकर्मा अचानक बिरामी चिनेजानेकाको सल्लाह आयुर्वेद औषधि खान थाले रोग निको बीपी कोइराला स्वास्थ्य विज्ञान प्रतिष्ठा डाक्टर सञ्जीवकुमार शर्मा सञ्चालन घर स्वास्थ्य शिविर जँचाउन गएपछि आफ्ना मिर्गौला खराब
2	मिर्गौला रोग लागिसकेपछि पूर्णतः निको पार् औषधि मिर्गौला बिग्र अवस्था पूर्ण मिर्गौला फेल मान् मिर्गौलाको रोग लक्षण देख् अधिकांशले रोग अवस्था पाउ त्यस्तो मिर्गौला फेर् डायलाइसिस रगत सफा गरिरहनुपर्छ उपचार पद्धति सामान्य आर्थिक हैसियत बिरामी धान् सक् महँगो डायलाइसिसका करिब लाख लाख रुपैयाँसम्म खर्च लाग् मिर्गौला प्रत्यारोपण लाख रुपैयाँसम्म खर्च लाग् भन् नेपाल औसत आम्दानी धान् सक् अवस्था
3	मिर्गौला रोग लाग् सामान्यतः अनुहार खुट्टा सुन्निने भोक लाग् वाकवाक लाग् कमजोर महसुस छिट्टै थाक् अनुभव कम्मर भाग दुख् लक्षण देख् रोग लाग् पिसाब धमिलो रातो रात पिसाब लाग् गर् लक्षण देखापरे बेल् जाँच् गराउनुपर्छ समयमै रोग उपचार
4	भाव नेपाल नामक संस्था ललितपुर पवन प्रकृति बोर्डिङ स्कुल विद्यार्थी स्थानीय समुदाय भ्याक्सिन ल अवस्था महानगरीय प्रहरी वृत सात टोली भ्याक्सिनसहित समात् स्वास्थ्य सेवा विभाग व्यवस्थापन शाखा संस्था जथाभावी हेपाटाइटिस भ्याक्सिन ल माघ हेपाटाइटिस भ्याक्सिन रोक ल
5	भिटामिन डीले शरीर हाड बलि बनाउने पहिले भए अनुसन्धान गर्भवती आमा भिटामिन डीको मात्रा सेवन गर् बाल मांसपेशी विकास देखाएको
6	अनुसन्धान गर्भवती अवस्था भिटामिन डीको मात्रा सेवन बाल वर्ष ह् जीवन मांसपेशी बलि दाबी गर् अनुसन्धान गर्भावस्थामा भिटामिन अतिरिक्त खुराक दि असर अध्ययन भइरहेको शरीर घाम पर् छाला भिटामिन निर्माण गर् गर्भावस्थामा महिला ल भिटामिन अतिरिक्त खुराक भिटामिन पाउ
7	गर्भावस्थामा भिटामिन डीको कम छोराछोरी युवावस्थामा कमजोर वृद्धावस्थामा मांसपेशी कमजोर ह् कमजोर स्वास्थ्य मधुमेह हाड भाँच् बेलाबेला लड् समस्या आउ डक्टर हार्वे बताउ अनुसन्धान गर्भावस्थामा भिटामिन डीको अतिरिक्त खुराक जन्म बाल हाड मांसपेशी पाउ गर्भवती महिला अध्ययन
8	आर्थिक शैक्षिक दृष्टि धनी मान् इलाम अचेल डिप्रेसन बिरामी बढ् जा देख् उदाहरण करोड लाख जनसंख्या जिल्ला गाउँ फलानोलाई डिप्रेसन डिप्रेसनको उपचार गराउँदैछ् डिप्रेसनले सक् जस्ता वाक्य सुन् पाउ संसार कुल जनसंख्याको प्रति जीवन एक भन् डिप्रेसनले युवायुवतीदेखि बुढापाकासम्मै पीडित

Testing Sample



1	मुस्कान तपाईंतर्फ आकर्षित तनाव घट् सघाउँछ साइकोलजिकल साइन्समा प्रकाशित अध्ययन दाबी मुस्कान तपाईंको मुटु चाल स्तर गर् तनाव मुक्ति पाउ शरीर सघाउँछ
2	राज हैजा जीवाणु देख् शुक्र ट्रोपिकल सरुवा रोग अस्पताल टेकु तथ्यांकअनुसार राज स्वयम्भू ताहाचल रवि टेकु भन्सार कालिमाटी भीमसेन वानेश्वर कलंकी डल्लु क्षेत्र हैजा देख्
3	ढाड दुख् बिरामी डाक्टर प्रश्न सोध्ने गर् ढाड दुख् ढाड दुख् समस्या ढाड कुन भाग दुख् दुख् न्रिरा पर् महसुस स्टेरोइड औषधि दुखाइ औषधि सेवन दिसा पिसाब रोग लाग्

Appendix B

Sample Source Code

Document Preprocessing

```
public class Preprocessing{
public List<List<String>> stopwordRomoval(List<String> stopwordDict,
        List<List<String>> input) {
    List<List<String>> output = remove(stopwordDict, input);
    return output;
}

public List<List<String>> symbolRemoval(List<String> symbolDict,
        List<List<String>> input) {
    List<List<String>> output = remove(symbolDict, input);
    return output;
}
    public List<List<String>> digitRemoval(List<List<String>> input) {
        List<List<String>> output=new ArrayList<List<String>>();
        int nClasses=input.size();
        for(int iClass=0;iClass<nClasses;iClass++){
            int nSamples=input.get(iClass).size();
            List<String> sOut = new ArrayList<String>();
            for(int iSample=0;iSample<nSamples;iSample++){
                String sentence=input.get(iClass).get(iSample);
                if (sentence.length()>0) {
                    sentence= sentence.replaceAll("[nepaliDigits]+",""); // nepaliDigits contain Nepali digits.
                    sentence=sentence.replaceAll("[0-9]+", "");
                    sentence= sentence.replaceAll("[A-Za-z]+", "");
                }
                sOut.add(iSample,sentence);
            }
            output.add(iClass,sOut);
        }
    return output;
    }

private List<List<String>> remove(List<String> noise,
        List<List<String>> input) {
    int nClasses = input.size();

    List<List<String>> output = new ArrayList<List<String>>();
    for (int i = 0; i < nClasses; i++) {
        int nSamples = input.get(i).size();
        List<String> sampleArray = new ArrayList<String>();
        for (int j = 0; j < nSamples; j++) {
            List<String> tokenArray = new ArrayList<String>();
            String[] tArray = input.get(i).get(j)
                    .split(”[  .|]+”);
            if (tArray.length == 0) {
                continue;
            }
            for (int k = 0; k < tArray.length; k++) {
                if (!tArray[k].isEmpty() || !tArray[k].trim().isEmpty())
                    tokenArray.add(tArray[k].trim());
            }
            if (tokenArray.isEmpty()) {
                continue;
            }
            tokenArray.removeAll(noise);
            StringBuilder builder = new StringBuilder();
            for (String s : tokenArray) {
                builder.append(s);
                builder.append(” ”);
            }
            sampleArray.add(j, builder.toString());
            tokenArray.clear();
        }
        output.add(i, sampleArray);
    }
    return output;
}

public List<List<String>> stemming(List<List<String>> input) {
    List<List<String>> output = new ArrayList<List<String>>();
    int nClasses = input.size();
    for (int iClass = 0; iClass < nClasses; iClass++) {
        int nSamples = input.get(iClass).size();
        List<String> sTempOut = new ArrayList<String>();
        List<String> sOut = new ArrayList<String>();
        for (int iSample = 0; iSample < nSamples; iSample++) {
            String sentence = input.get(iClass).get(iSample);
            if (sentence.length() > 0) {
                sTempOut = StemmerStart.stemmer(sentence); //StemmerStart.
                    stemmer() is a Nepali Madan Puruskar Pustakalaya's
                    stemmer.
                StringBuilder builder = new StringBuilder();
                for (String s : sTempOut) {
                    builder.append(s);
                    builder.append(” ”);
                }
                sOut.add(builder.toString());
            } else {
                sOut.add(” ”);
            }
        }
        output.add(iClass, sOut);
    }
    return output;
}
```

```

    public List<String> listTokanize(List<String> input) {

        List<String> temp = new ArrayList<String>();

        for (Object o : input) {
            String[] tt = o.toString().split(”[ ,??\t\n]”);
            for (int i = 0; i < tt.length; i++) {
                if (!tt[i].trim().isEmpty()) {
                    temp.add(tt[i]);
                }
            }
        }
        return temp;
    }

    @Override
    public List<String> sentenceTokanize(String sentence) {
        List<String> temp = new ArrayList<String>();
        String[] tt = sentence.split(”[ ,|.\t]”);
        for (int i = 0; i < tt.length; i++) {
            if (!tt[i].trim().isEmpty()) {
                temp.add(tt[i]);
            }
        }
        return temp;
    }

    @Override
    public List<String> listSentencise(List<String> input) {
        List<String> temp = new ArrayList<String>();
        for (String s : input) {
            String[] tt = s.trim().split(”[?.|\n]”);
            for (int i = 0; i < tt.length; i++) {
                if (tt[i].length() != 0 || !tt[i].trim().equals(””)
                        || !tt[i].trim().isEmpty()) {
                    temp.add(tt[i].trim());
                }
            }
        }
        return temp;
    }
}
```

Feature Extraction

```
public class FeatureExtraction {
    public double df(List<List<String>> input, String token) {
        double df = 1.0; // no avoid divide by zero error
        int nClasses = input.size();
        for (int iClass = 0; iClass < nClasses; iClass++) {
            int nSamples = input.get(iClass).size();
            for (int iSample = 0; iSample < nSamples; iSample++) {
                String doc = input.get(iClass).get(iSample);
                if (!doc.isEmpty()) {
```

```java
                    Boolean found = Arrays.asList(doc.split(” ”)).contains(
                            token);
                    if (found) {
                        df = df + 1;
                    }
                }
            }
        }

        return df;
    }

    public double idf(List<List<String>> input, String token) {
        int N = 0; // Total number of documents in the collection
        for (int iClass = 0; iClass < input.size(); iClass++)
            for (int iSample = 0; iSample < input.get(iClass).size();
                iSample++)
                N = N + 1;
        double df_t = df(input, token);
        double idf = Math.log10(N / df_t);
        return idf;
    }

    public double tf(Map<String, Integer> docFreqMap, String token) {
        int tf_td = 0;
        if (docFreqMap.containsKey(token)) {
            tf_td = docFreqMap.get(token);
        }
        return (1 + Math.log10(tf_td));
    }

    public Map<String, Integer> wordFrequencyCount(String doc) {
        Map<String, Integer> hmap = new HashMap<String, Integer>();
        for (String tempStr : doc.split(”[ ]+”)) {
            if (hmap.containsKey(tempStr)) {
                Integer i = hmap.get(tempStr);
                i += 1;
                hmap.put(tempStr, i);
            } else
                hmap.put(tempStr, 1);
        }
        return hmap;
    }

    public double tfidf(double tf, double idf) {
        return tf * idf;
    }

    public double[][] createFeatureVector(HashSet<String> dict,
            List<List<String>> input) {
        int nDocs = 0; // Total number of documents
        for (int iClass = 0; iClass < input.size(); iClass++)
            for (int iSample = 0; iSample < input.get(iClass).size();
                iSample++)
                nDocs = nDocs + 1;

        // Total number of features
        int nFeatures = 1 + dict.size(); // 1 for class index

        Object[] dictionary = dict.toArray();
        double[][] featureVector = new double[nDocs][nFeatures];
        for (int i = 0; i < nDocs; i++) {
            for (int j = 0; j < nFeatures; j++) {
                featureVector[i][j] = 0.0;
            }
        }

        int iDoc = 0;
        for (int iClass = 0; iClass < input.size(); iClass++) {
            for (int iSample = 0; iSample < input.get(iClass).size();
               iSample++) {
                Map<String, Integer> docFreqMap = wordFrequencyCount(input.
                   get(
                        iClass).get(iSample));
                featureVector[iDoc][0] = iClass; // featureVector[docId
                   ][0]=classId
                for (Entry<String, Integer> entry : docFreqMap.entrySet())
                   {
                    String iWord = entry.getKey();
                    Integer value = entry.getValue();
                    int iWordIndex = Arrays.asList(dictionary).indexOf(
                       iWord);
                    if (iWordIndex != -1) { // If word is not in dictionary
                       ,
                                            // leave it.
                        double tf = tf(docFreqMap, iWord);
                        double idf = idf(input, iWord);
                        double tfidf = tfidf(tf, idf);
                        featureVector[iDoc][iWordIndex + 1] = tfidf;
                    }
                }
                iDoc++;
            }
        }
        return featureVector;
    }
}
```

ANN Training/Testing

```

import weka.classifiers.Classifier;
import weka.classifiers.Evaluation;
import weka.classifiers.functions.MultilayerPerceptron;
import weka.core.Attribute;
import weka.core.FastVector;
import weka.core.Instance;
import weka.core.Instances;
import weka.core.Utils;

public class ANN{
public Classifier trainANN(int numClasses, int attribSize, double[][]
    featureVector)
```

```java
        throws Exception {
    // Declare two numeric attributes
    Attribute[] fvAttribute = new Attribute[attribSize];
    for (int i = 0; i < attribSize; i++) {
        String attribname = "" + i;
        fvAttribute[i] = new Attribute(attribname);

    }
    // Declare the class attribute along with its values
    FastVector fvClassVal = new FastVector(numClasses); // Classes
    for (int i = 0; i < numClasses; i++) {
        String className = "" + i;
        fvClassVal.addElement(className);
    }

    Attribute ClassAttribute = new Attribute("classes", fvClassVal);

    // Declare the feature vector
    FastVector fvWekaAttributes = new FastVector(attribSize + 1);
    for (int i = 0; i < attribSize; i++) {
        fvWekaAttributes.addElement(fvAttribute[i]);

    }
    fvWekaAttributes.addElement(ClassAttribute);

    // Create an empty training set
    Instances trainingSet = new Instances("Rel", fvWekaAttributes,
            featureVector.length);

    // Set class index
    trainingSet.setClassIndex(attribSize);

    // Fill the training set
    Instance iSample = new Instance(attribSize + 1);

    for (int i = 0; i < featureVector.length; i++) {
        int k = 0;
        for (int j = 1; j < featureVector[i].length; j++) {
            iSample.setValue((Attribute) fvWekaAttributes.elementAt(k),
                    featureVector[i][j]);
            k++;
        }
        iSample.setValue(
                (Attribute) fvWekaAttributes.elementAt(attribSize),
                Integer.toString((int) featureVector[i][0]));
        trainingSet.add(iSample);

    }

    double learningRate=0.3; // between [0 1]
    double momentum =0.9; // between [0 1]
    int numEpoch=500;
    int validationSet=0; //between [0 100]
    int seed=0; //geater or equal to 0, weight seed
    int consequtiveErrThresh=20; //default 20
    int hiddenLayerNeurons=100;
    Classifier cModel = (Classifier) new MultilayerPerceptron();
    String opt=” –L ”+learningRate+” –M ”+momentum+” –N ”+numEpoch+” –V ”+
        validationSet+” –S ”+seed+” –E ”+consequtiveErrThresh+” –H ”+
        hiddenLayerNeurons;
    cModel.setOptions(Utils.splitOptions(opt));
    cModel.buildClassifier(trainingSet);
    return cModel;
}

public double[][] testANN(Classifier cModel,  int numClasses, int
    attribSize , double[][] featureVector)
        throws Exception {
    // Declare two numeric attributes
int numInstances=featureVector.length;

    Attribute[] fvAttribute = new Attribute[attribSize];
    for (int i = 0; i < attribSize; i++) {
        String attribname = ”” + i;
        fvAttribute[i] = new Attribute(attribname);

    }
    FastVector fvClassVal = new FastVector(numClasses); // Classes
    for (int i = 0; i < numClasses; i++) {
        String className = ”” + i;
        fvClassVal.addElement(className);
    }

    Attribute ClassAttribute = new Attribute(”classes”, fvClassVal);

    // Declare the feature vector
    FastVector fvWekaAttributes = new FastVector(attribSize + 1);
    for (int i = 0; i < attribSize; i++) {
        fvWekaAttributes.addElement(fvAttribute[i]);
    }
    fvWekaAttributes.addElement(ClassAttribute);

    // Create an empty training set
    Instances testingSet = new Instances(”Rel”, fvWekaAttributes,
            featureVector.length);

    // Set class index
    testingSet.setClassIndex(attribSize);
    // Fill the training set
    Instance iSample = new Instance(attribSize + 1);

    for (int i = 0; i < featureVector.length; i++) {
        int k = 0;
        for (int j = 1; j < featureVector[i].length; j++) {
            iSample.setValue((Attribute) fvWekaAttributes.elementAt(k),
                    featureVector[i][j]);
            k++;
        }
        iSample.setValue(
                (Attribute) fvWekaAttributes.elementAt(attribSize),
                Integer.toString((int) featureVector[i][0]));
        testingSet.add(iSample);

    }
    // Test the model
    Evaluation eTest = new Evaluation(testingSet);
    eTest.evaluateModel(cModel, testingSet);


    double[][] output = new double[numInstances][numClasses];
    for (int i = 0; i < numInstances; i++)
        for (int j = 0; j < numClasses; j++)
                output[i][j]=0.0;

    for (int i = 0; i < numInstances; i++) {
        double y = eTest.evaluateModelOnce(cModel, testingSet.instance(i));
        for (int k = 0; k < numClasses; k++) {
            if ((int) y == k) {
                output[i][k] = 1;
                break;
            }
        }
    }

    for (int i = 0; i < numInstances; i++){
        for (int j = 0; j < numClasses; j++){
                System.out.print(" "+output[i][j]);
        }
    System.out.println();
    }

    // Get the confusion matrix
    double[][] cmMatrix = eTest.confusionMatrix();
    for (int row_i = 0; row_i < cmMatrix.length; row_i++) {
        for (int col_i = 0; col_i < cmMatrix.length; col_i++) {
            System.out.print(cmMatrix[row_i][col_i]);
            System.out.print("|");
        }
        System.out.println();
    }
    return output;
}
}
```

Naive Bayes Training/Testing

```

import weka.classifiers.Classifier;
import weka.classifiers.Evaluation;
import weka.classifiers.bayes.NaiveBayes;
import weka.core.Attribute;
import weka.core.FastVector;
import weka.core.Instance;
import weka.core.Instances;

public class Naive {
    public Classifier trainNaive(int numClasses, int attribSize, double[][]
         featureVector)
            throws Exception {
        // Declare two numeric attributes
        Attribute[] fvAttribute = new Attribute[attribSize];
        for (int i = 0; i < attribSize; i++) {
            String attribname = ”” + i;
            fvAttribute[i] = new Attribute(attribname);

        }
        // Declare the class attribute along with its values
        FastVector fvClassVal = new FastVector(numClasses); // Classes
        for (int i = 0; i < numClasses; i++) {
            String className = ”” + i;
            fvClassVal.addElement(className);
        }

        Attribute ClassAttribute = new Attribute(”classes”, fvClassVal);

        // Declare the feature vector
        FastVector fvWekaAttributes = new FastVector(attribSize + 1);
        for (int i = 0; i < attribSize; i++) {
            fvWekaAttributes.addElement(fvAttribute[i]);

        }
        fvWekaAttributes.addElement(ClassAttribute);

        // Create an empty training set
        Instances trainingSet = new Instances(”Rel”, fvWekaAttributes,
                featureVector.length);

        // Set class index
        trainingSet.setClassIndex(attribSize);
        // Fill the training set
        Instance iSample = new Instance(attribSize + 1);

        for (int i = 0; i < featureVector.length; i++) {
            int k = 0;
            for (int j = 1; j < featureVector[i].length; j++) {
                iSample.setValue((Attribute) fvWekaAttributes.elementAt(k),
                        featureVector[i][j]);
                k++;
            }
            iSample.setValue(
                    (Attribute) fvWekaAttributes.elementAt(attribSize),
                    Integer.toString((int) featureVector[i][0]));
            trainingSet.add(iSample);

        }
        Classifier cModel = (Classifier) new NaiveBayes();
        cModel.buildClassifier(trainingSet);

        return cModel;
    }

    public double[][] testNaive(Classifier cModel, int numClasses, int
        attribSize, double[][] featureVector)
            throws Exception {
        // Declare two numeric attributes
int numInstances=featureVector.length;

```

```
        Attribute[] fvAttribute = new Attribute[attribSize];
        for (int i = 0; i < attribSize; i++) {
            String attribname = ”” + i;
            fvAttribute[i] = new Attribute(attribname);

        }
        FastVector fvClassVal = new FastVector(numClasses); // Classes
        for (int i = 0; i < numClasses; i++) {
            String className = ”” + i;
            fvClassVal.addElement(className);
        }

        Attribute ClassAttribute = new Attribute(”classes”, fvClassVal);

        // Declare the feature vector
        FastVector fvWekaAttributes = new FastVector(attribSize + 1);
        for (int i = 0; i < attribSize; i++) {
            fvWekaAttributes.addElement(fvAttribute[i]);

        }
        fvWekaAttributes.addElement(ClassAttribute);

        // Create an empty training set
        Instances testingSet = new Instances(”Rel”, fvWekaAttributes,
                featureVector.length);

        // Set class index
        testingSet.setClassIndex(attribSize);
        // Fill the training set
        Instance iSample = new Instance(attribSize + 1);

        for (int i = 0; i < featureVector.length; i++) {
            int k = 0;
            for (int j = 1; j < featureVector[i].length; j++) {
                iSample.setValue((Attribute) fvWekaAttributes.elementAt(k),
                        featureVector[i][j]);
                k++;
            }
            iSample.setValue(
                    (Attribute) fvWekaAttributes.elementAt(attribSize),
                    Integer.toString((int) featureVector[i][0]));
            testingSet.add(iSample);
        }
        // Test the model
        Evaluation eTest = new Evaluation(testingSet);
        eTest.evaluateModel(cModel, testingSet);

        double[][] output = new double[numInstances][numClasses];
        for (int i = 0; i < numInstances; i++)
            for (int j = 0; j < numClasses; j++)
                    output[i][j]=0.0;

        for (int i = 0; i < numInstances; i++) {
            double y = eTest.evaluateModelOnce(cModel, testingSet.instance(
               i));
            for (int k = 0; k < numClasses; k++) {
                if ((int) y == k) {
                    output[i][k] = 1;
                    break;
                }
            }
        }
        return output;
    }
}
```

System Evaluation

```
public class Confusion{
    public void confusionResults(double [][] targets,double [][] outputs) {
        ConfusionResults cr=  confusion(targets, outputs);
        cr.printC();
        cr.printCm();
        cr.printInd();
        cr.printPer();
        EvaluationResults er=evaluation(cr);
        er.printEvaluationResults();
    }
    public  ConfusionResults confusion(double targets[][], double outputs[][]) {
        ConfusionResults cr = new ConfusionResults();

        int numClasses = outputs.length;
        cr.setClasses(numClasses);
        if (numClasses == 1) {
            System.out.println(”Code is not written for this case.”);
            //return;
        }

        // Unknown/dont-care targets
        //Code is not written to handle infinite or nan numbers in the target and output.

        int numSamples = targets[0].length;
        cr.setSamples(numSamples);
        //Transform outputs  (maximum value is set to 1 and other values to 0, column-wise)
        for (int col = 0; col < numSamples; col++) {
            double max = outputs[0][col];
            int ind = 0;

            for (int row = 1; row < numClasses; row++) {
                if (outputs[row][col] > max) {
                    max = outputs[row][col];
                    ind = row;
                }
                outputs[row][col] = 0.0;
            }
            outputs[0][col] = 0.0;
            outputs[ind][col] = 1;
        }
        //Confusion value
        int count = 0;
        for (int row = 0; row < numClasses; row++) {
            for (int col = 0; col < numSamples; col++) {
                if (targets[row][col] != outputs[row][col])
                    count++;
            }
        }
        double c = (double) count / (double) (2 * numSamples);

//              Confusion matrix
        int[][] cm = new int[numClasses][numClasses];
        for (int row = 0; row < numClasses; row++) {
            for (int col = 0; col < numClasses; col++) {
                cm[row][col] = 0;
            }
        }

        int[] i = new int[numSamples];
        int[] j = new int[numSamples];

        for (int col = 0; col < numSamples; col++) {
            for (int row = 0; row < numClasses; row++) {
                if (targets[row][col] == 1.0) {
                    i[col] = row;
                    break;
                }
            }
        }

        for (int col = 0; col < numSamples; col++) {
            for (int row = 0; row < numClasses; row++) {
                if (outputs[row][col] == 1.0) {
                    j[col] = row;
                    break;
                }
            }
        }

        for (int col = 0; col < numSamples; col++) {
            cm[i[col]][j[col]] = cm[i[col]][j[col]] + 1;
        }

//              Indices
        int[][][] ind1 = new int[numClasses][numClasses][3];

        String[][] ind = new String[numClasses][numClasses];
        for (int row = 0; row < numClasses; row++)
            for (int col = 0; col < numClasses; col++)
                ind[row][col] = ””;


        for (int col = 0; col < numSamples; col++) {
            if (ind[i[col]][j[col]].equals(””))
                ind[i[col]][j[col]] = new StringBuilder().append(col).
                    toString();
            else
                ind[i[col]][j[col]] = new StringBuilder().append(ind[i[col
                    ]][j[col]]).append(”,”).append(col).toString();
        }
```

```
//             Percentages

        double[][] per = new double[numClasses][4];
        for (int row = 0; row < numClasses; row++) {
            for (int col = 0; col < 4; col++) {
                per[row][col] = 0.0;
            }
        }

        for (int row = 0; row < numClasses; row++) {
            double[] yi = new double[numSamples];
            double[] ti = new double[numSamples];
            for (int col = 0; col < numSamples; col++) {
                yi[col] = outputs[row][col];
                ti[col] = targets[row][col];

            }

            int a = 0, b = 0;
            for (int col = 0; col < numSamples; col++) {
                if (yi[col] != 1 && ti[col] == 1) a = a + 1;
                if (yi[col] != 1) b = b + 1;
            }
            per[row][0] = (double) a / (double) b;


            a = 0;
            b = 0;
            for (int col = 0; col < numSamples; col++) {
                if (yi[col] == 1 && ti[col] != 1) a = a + 1;
                if (yi[col] == 1) b = b + 1;
            }
            per[row][1] = (double) a / (double) b;


            a = 0;
            b = 0;
            for (int col = 0; col < numSamples; col++) {
                if (yi[col] == 1 && ti[col] == 1) a = a + 1;
                if (yi[col] == 1) b = b + 1;
            }
            per[row][2] = (double) a / (double) b;

            a = 0;
            b = 0;
            for (int col = 0; col < numSamples; col++) {
                if (yi[col] != 1 && ti[col] != 1) a = a + 1;
                if (yi[col] != 1) b = b + 1;
            }
            per[row][3] = (double) a / (double) b;

        }
        //NAN handling
        for (int row = 0; row < numClasses; row++) {
            for (int col = 0; col < 4; col++) {
                if(Double.isNaN(per[row][col]))
                per[row][col] = 0.0;
            }
        }

        cr.setC(round(c,2));
        cr.setCm(cm);
        cr.setInd(ind);
        cr.setPer(per);
        return cr;

    }
    public  EvaluationResults evaluation(ConfusionResults cr){

        double[][] per =cr.getPer();
        int numClasses=cr.getClasses();

        //Average Accuracy (The average per-class effectiveness of a classifier)
        double avgAccuracy=0.0;
        double fn=0.0,fp=0.0,tp=0.0,tn=0.0;
        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            avgAccuracy=+avgAccuracy+((tp+tn)/(tp+fn+fp+tn));
        }
        avgAccuracy=avgAccuracy/numClasses;

        //Error Rate (The average per-class classification error)
        double errRate=0.0;
        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            errRate=+errRate+((fp+fn)/(tp+fn+fp+tn));
        }
        errRate=errRate/numClasses;

        //Precision-Micro (Agreement of the data class labels with those of a classifiers if calculated from sums of per-text decisions)
        double  numerator=0.0;
        double  denominator=0.0;
        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            numerator=numerator+tp;
            denominator=denominator+ (tp+fp);
        }

        double     precisionMicro=numerator/denominator;

        //Recall-Micro (Effectiveness of a classifier to identify class labels if calculated from sums of per-text decisions)
        numerator=0.0;
        denominator=0.0;
        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            numerator=numerator+tp;
            denominator=denominator+ (tp+fn);
        }

        double  recallMicro=numerator/denominator;

        //Fscore-Micro (Relations between data's positive labels and those given by a classifier based on sums of per-text decisions)
        double beta=1;
        numerator=(Math.pow(beta,2)+1)*precisionMicro*recallMicro;
        denominator=Math.pow(beta,2)*precisionMicro+recallMicro;
        double fscoreMicro=numerator/denominator;

        //Precision-Macro (An average per-class agreement of the data class labels with those of a classifiers)
        double precisionMacro=0.0;
        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            precisionMacro=precisionMacro+(tp/(tp+fp));
        }

        precisionMacro=precisionMacro/numClasses;

        //Recall-Micro (An average per-class effectiveness of a classifier to identify class labels)
        double recallMacro=0.0;

        for (int i=0;i<numClasses;i++){
            fn=per[i][0];
            fp=per[i][1];
            tp=per[i][2];
            tn=per[i][3];
            recallMacro=recallMacro+(tp/(tp+fn));
        }
        recallMacro=recallMacro/numClasses;

        //Fscore-Macro (Relations between data's positive labels and those given by a classifier based on a per-class average)
        beta=1;
        numerator=(Math.pow(beta,2)+1)*precisionMacro*recallMacro;
        denominator=Math.pow(beta,2)*precisionMacro+recallMacro;
        double  fscoreMacro=numerator/denominator;

        EvaluationResults er=new EvaluationResults();
        er.setAvgAccuray(round(avgAccuracy,4));
        er.setErrRate(round(errRate,4));
        er.setPrecisionMicro(round(precisionMicro,4));
        er.setRecallMicro(round(recallMicro,4));
        er.setFscoreMicro(round(fscoreMicro,4));
        er.setPrecisionMacro(round(precisionMacro,4));
        er.setRecallMacro(round(recallMacro ,4));
        er.setFscoreMacro(round(fscoreMacro ,4));
        return er;

    }
    public  double round(double valueToRound, int numberOfDecimalPlaces)
    {
        double multipicationFactor = Math.pow(10, numberOfDecimalPlaces);
        double interestedInZeroDPs = valueToRound * multipicationFactor;
        return Math.round(interestedInZeroDPs) / multipicationFactor;
    }

}
```