DWIT COLLEGE

DEERWALK INSTITUTE OF TECHNOLOGY

DWIT EMAIL CLASSIFIER USING LOGISTIC REGRESSION

A MINI PROJECT REPORT

Submitted to

Department of Computer Science

DWIT College

Submitted by

Asmita Subedi

December, 2016

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

SUPERVISOR’S RECOMENDATION

I hereby recommend that this project prepared under my supervision by ASMITA SUBEDI entitled **“DWIT EMAIL CLASSIFIER”** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology be processed for the evaluation.

…………………………………………

Dr. Basanta Joshi

Assistant Professor
IOE, Tribhuvan University

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

LETTER OF APPROVAL

This is to certify that this project prepared by ASMITA SUBEDI entitled **“DWIT EMAIL CLASSIFIER”** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

…………………………………… Basanta Joshi [Supervisor] Assistant Professor IOE, Tribhuvan University	………………………………………… Hitesh Karki Chief Academic Officer DWIT College

ACKNOWLEDGEMENT

I would like to thank Mr. Basanta Joshi, Assistant Professor, IOE, TU and Ms. Shweta Shrestha, Mentor, DWIT for supervising this project and giving assistance when faced with problems. Without their guidance and persistent help, this project would not have been possible.

I would like to thank Mr. Abhishek Gupta, mentor, DWIT for providing technical help and providing deeper knowledge about the technology needed for this project. Without his help, I would be struggling to understand the theory needed and might have reached an incorrect conclusion.

At last my sincere thanks goes to my parents and members of my family, who have always supported me and to all of my friends who directly or indirectly helped me to complete this project report.

Asmita Subedi
Roll no: 0346

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.

… … … … … … … …
Asmita Subedi
Date: December, 2016

ABSTRACT

Emails are the most common mode of communication for events, assignments, and other notices among DWIT members. The bunch of emails are send to inboxes each day concerning various events, assignments or informational notices. Unfortunately, with so many emails going around, not all of the emails sent out to the inbox can be read or even organized into useful categories that could help disseminate information better.

The goal of DWIT Email Classifier is to address this problem, and the approach used is to build a training model using Logistic Regression Classification Model which automatically syncs with the user Gmail inbox and classifies the new, unseen emails into one of the four categories; Club emails, Class emails, Administration emails and Miscellaneous. The 4,000 existing emails were used as the training set in this system.

Keywords: *Text classification, Document classification, Feature extraction, Content classification, Natural Language Processing, Opinion Mining*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CSV: Comma Separated Values

VSM: Vector Space Model

MaxEnt: Maximum Entropy Model

WEKA: Waikato Environment for Knowledge Analysis

IMAP: Internet Message Access Protocol

CHAPTER 1: INTRODUCTION

1.1 Introduction

Email is one of the most popular, fastest and cheapest means of communication. It has become a part of everyday life for millions of people, changing the way we work and collaborate. Email is not only used to support conversation but also as a task manager, document delivery system and archive. The shortcoming of this success is the constantly growing volume of emails we receive. To deal with this problem and facilitate efficient retrieval when needed, many people categorize emails and prioritize them according to its importance.

Emails are also the most common mode of communication for events, assignments, and other notices among DWIT members. Hence, an email classifier system is necessary to organize emails into useful categories that could help disseminate information better. DWIT Email Classifier automatically classifies the new, unseen emails using supervised learning approach into one of these four categories:

1. Club emails (club events/programs emails)
2. Class emails (assignments/notes emails)
3. Administration emails
4. Miscellaneous.

In a supervised learning approach, given a supervision in the form of a set of labelled training sets, the goal is to build a classifier, which is then used to predict the category of an unseen incoming email.

1.2 Problem Statement

Email is one of the most popular, fastest and cheapest means of communication. It has become a part of everyday life for millions of people which results in the constant growth in the volume of emails we receive. To deal with this problem and facilitate efficient retrieval when needed, many people categorize emails and prioritize them according to its importance.

Email packages typically allow the user to hand construct keyword-based rules to organize emails into labeled folders and filter messages. However, manually constructing a set of robust rules is a difficult task as users constantly create, delete, and reorganize their folders. Even if the folders remain the same, the nature of the e-mails within a folder may well change over time. Hence, the rules must be constantly modified by the user which is a time consuming process. A system that can automatically learn how to classify e-mails into a set of folders is highly desirable.

1.3 Objectives

1.3.1 General objective

To implement Logistic Regression algorithm for creating a training model to automatically classify the new, unseen emails.

1.3.2 Specific objective

a. To organize and maintain emails properly.

b. To categorize emails into 4 categories:

   1. Club emails (club events/programs emails)
   2. Class emails (assignments/notes emails)
   3. Administration emails
   4. Miscellaneous.

1.4 Scope

DWIT Email Classifier can be used by students, faculty, staffs and other members of DWIT to classify the emails into different categories for better management of emails.

1.5 Limitation

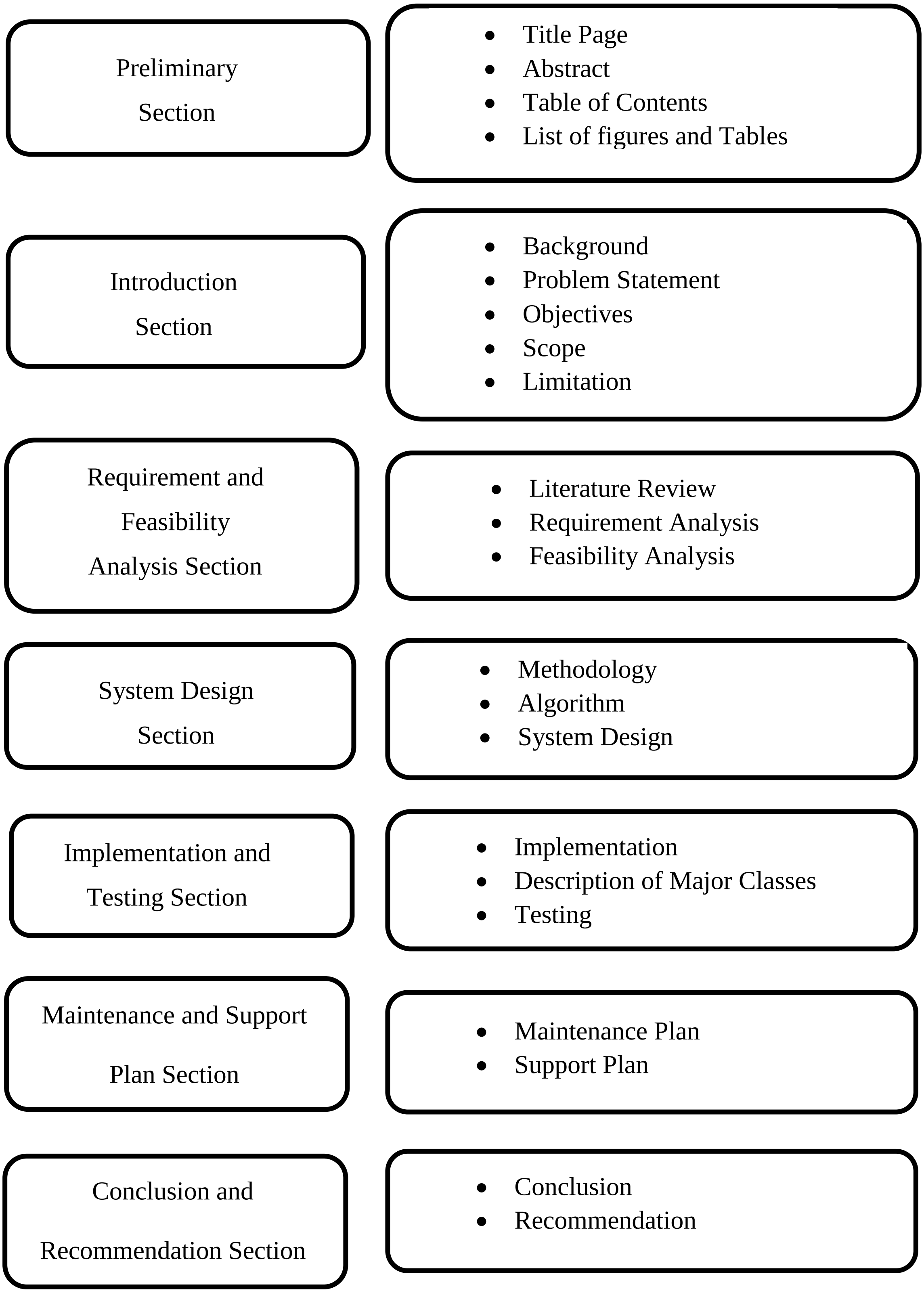
a) Sample of emails used in the training set cannot represent all types of emails circulated at DWIT.

b) Since the classification is based on supervised learning, the accuracy rate of the training model varies the efficiency and reliability of the system.

c) DWIT Email Classifier is only designed for the use of DWIT members.

1.6 Outline of Document

The report is organized as follows:



Preliminary Section	• Title Page • Abstract • Table of Contents • List of figures and Tables
Introduction Section	• Background • Problem Statement • Objectives • Scope • Limitation
Requirement and Feasibility Analysis Section	• Literature Review • Requirement Analysis • Feasibility Analysis
System Design Section	• Methodology • Algorithm • System Design
Implementation and Testing Section	• Implementation • Description of Major Classes • Testing
Maintenance and Support Plan Section	• Maintenance Plan • Support Plan
Conclusion and Recommendation Section	• Conclusion • Recommendation

Figure 1 - Outline of document

CHAPTER 2: REQUIREMENT AND FEASIBILITY ANALYSIS

2.1 Literature Review

Kiritchenko and Matwin presented a paper on email classification through combining labeled and unlabeled data. Similar to many other papers, VSM is showed to be the best classifier in terms of prediction or classification performance. Text classification is used to classify emails into different folders based on predefined categories. Authors tried to define classes as interesting and uninteresting categories. An initial list of manually labeled emails can be used for the future automatic training and classification. (Svetlana Kiritchenko, 2001)

Carmona-Cejudo et al.'s paper is related to real time email classification and introduced GNUsmail open source for email folder classification. The application is developed to parse emails from different email clients and perform some data mining analysis using WEKA data mining tool. (José M. Carmona-Cejudo, 2011) In email, folder classification is also based on the time of email messages (Ron Bekkerman, 2004). The paper used Enron and SRI email datasets for the case study. Some new classification methods such as: MaxEnt were evaluated in the paper. The major decision to make in all email classification papers is what features to select. Features can be related to email title, from or to addresses or can be related to the content; words, sequence of words, etc. Natural language processing activities such as parsing and stemming are then involved to parse email contents and eliminate any words that may not be relevant for the classification process.

2.2 Requirement Analysis

Table 1- Functional and non-functional requirements

Functional requirement	Non-functional requirement
Application connects to Gmail	Accesses the user's Gmail through the application using OAuth 2 authorization.
Get the new, unseen emails	The new, unseen emails are extracted and saved on a CSV file
Categorize the email	The CSV file is sent as a test email set to the training model and the new email is categorized
Fill into the labeled folder	The categorized email is then filled into the corresponding labeled folder

Table 1 describes the basic functionality of DWIT Email Classifier which includes the application establishing a connection with Gmail, accessing the new unseen emails from the user's inbox, categorizing them and filling them into corresponding labeled folders.

2.3 Feasibility Analysis

2.3.1 Technical feasibility

DWIT Email Classifier is a browser plugin for Gmail. It uses HTML/CSS, JavaScript to create an extension while the underlying application is built in Python. It requires a server, client and internet connection to function properly. All of the technologies required by the project are easily available and can be accessed freely. This determines that DWIT Email Classifier was determined technically feasible.

2.3.2 Operational feasibility

DWIT Email Classifier is an extension and has no User Interface. It simply uses the Gmail UI and is easy to use. The application can be accessed from anywhere if there is a working internet connection. Hence, DWIT Email Classifier was determined operationally feasible.

CHAPTER 3: SYSTEM DESIGN

3.1 Methodology

This project uses training dataset of 4000 emails that I extracted from my own DWIT account using IMAP. This dataset consists of 1000 Club emails, 1000 Class emails, 1000 Administrative emails and 1000 Miscellaneous emails.

Firstly, tokenization was performed on the training dataset by using different parsing methods. Then, stop-words removal is performed on the input text to remove the insignificant words from the text provided. Stop-words do not provide much significance in classifying emails into distinct class, so those words were removed.

Then, Porter Stemming was performed on the processed texts. Porter Stemming helps to remove redundant use of words that mean the same thing. This means that instead of using fascinating, fascinated, fascinates, etc., only 'fascinate' could be used.

Then, a vocabulary of unique tokens is created from each document. The counts of how often each word occurs in the particular document is calculated which is then used to create a feature vector of each email document. The new email is then taken as input, the words in the text document is checked with the polarity of the words in feature vector, where large set of words are stored with their respective polarities. Lastly, the polarity of each word in the text is calculated and the total polarity of the full text is calculated. The email is then classified as a class which has maximum polarity. In this way, the final output of the classification was generated.

3.1.1 Data Collection

The emails was extracted from my own DWIT account using IMAP. The extracted 4000 emails were then manually categorized into four different categories and a training dataset was created. This dataset consists of 1000 Club emails, 1000 Class emails, 1000

Administrative emails and 1000 Miscellaneous emails. The new unseen emails, extracted using IMAP will then be used to perform test on the developed application.

Sample Input Data:

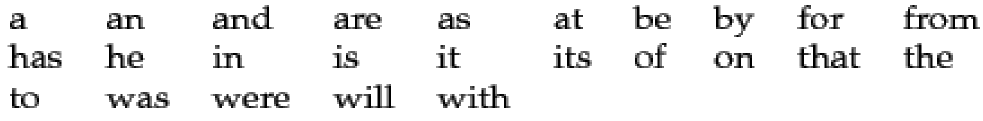
Mini Project 2017 Batch Please upload your documents in the link shared

3.1.2 Tokenization

Firstly, the stream of text are broken up into words, known as tokens. Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens. The list of tokens becomes input for further processing such as parsing or text mining. It helps to break the word from the sentence.

3.1.3 Stop word Removal

In computing, stop words are words which are filtered out before or after processing of natural language data. Stop words usually refer to the most common word in the NLP that helps to connect the words together to form meaningful sentence. Example is shown below:



a	an	and	are	as	at	be	by	for	from
has	he	in	is	it	its	of	on	that	the
to	was	were	will	with					

Figure 2 - Different examples of stop-words

Generally, in performing sentiment analysis, stop words are filtered out before processing the data.

3.1.4 Porter Stemming

Porter Stemming is the technique to bring all the related words (verb) to a recognizable verb form. In general, it is the morphological process for removing common morphological endings from words in English. It is mainly used to remove the redundant words that give the same meaning. For example, played, playing, plays, etc. are different forms of word play. So, instead of keeping all of those words, we represented all the words with a single word, i.e. play, by performing Porter Stemming.

3.1.5 Term Frequency - Inverse Document Frequency (TFIDF)

When we are analyzing text data, we often encounter words that occur across multiple documents from both classes. Those frequently occurring words typically don't contain useful or discriminatory information. A useful technique called term frequency-inverse document frequency (tf-idf) can be used to down weight those frequently occurring words in the feature vectors. The tf-idf can be defined as the product of the term frequency and the inverse document frequency:

$$\text{tf-idf}\ (t,d)= \text{tf}(t,d) * \text{idf}(t,d)$$

Here the tf(t, d) is the term frequency that signifies the number of times a term *t* occurs in a document *d*, and the inverse document frequency idf(t, d) can be calculated as:

$$\text{idf}(t,d) =\log n/(1+\text{df}(d,t)),$$

where is the total number of documents, and df(d, t) is the number of documents d that contain the term t.

3. 2 Algorithm

For the classification algorithm, Logistic Regression classifier is used. In regression analysis, we are given a number of predictor (explanatory) variables and a continuous response variable (outcome), and we try to find a relationship between those variables that allows us to predict an outcome.

"Logistic regression" means obtaining a best-fit logistic function.

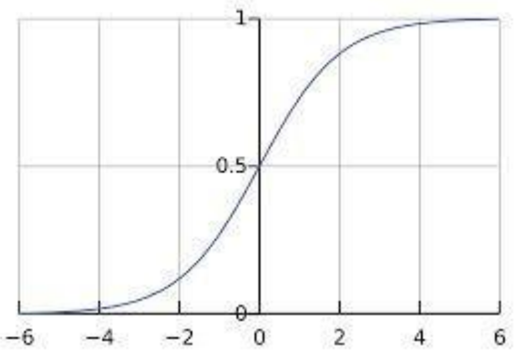


Figure 3 - Sigmoid Function

This probability function is the 'Sigmoid Function' which is:

$$1/(1+e^{-z})$$

3.3 System Design

3.3.1 State diagram

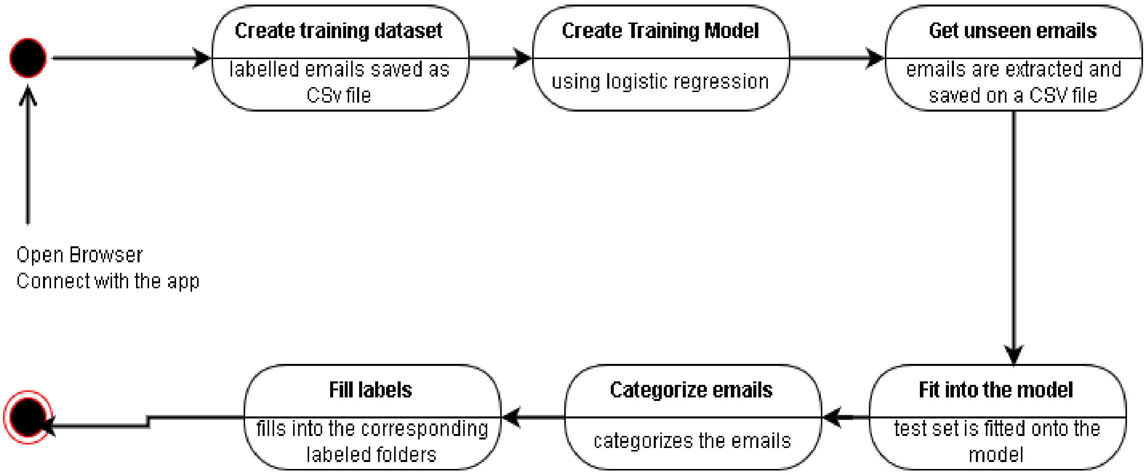

Create training dataset
labelled emails saved as CSv file
Create Training Model
using logistic regression
Get unseen emails
emails are extracted and saved on a CSV file
Open Browser
Connect with the app
Fill labels
fills into the corresponding labeled folders
Categorize emails
categorizes the emails
Fit into the model
test set is fitted onto the model

Figure 4 - State Diagram of DWIT Email Classifier

3.3.2 Sequence diagram

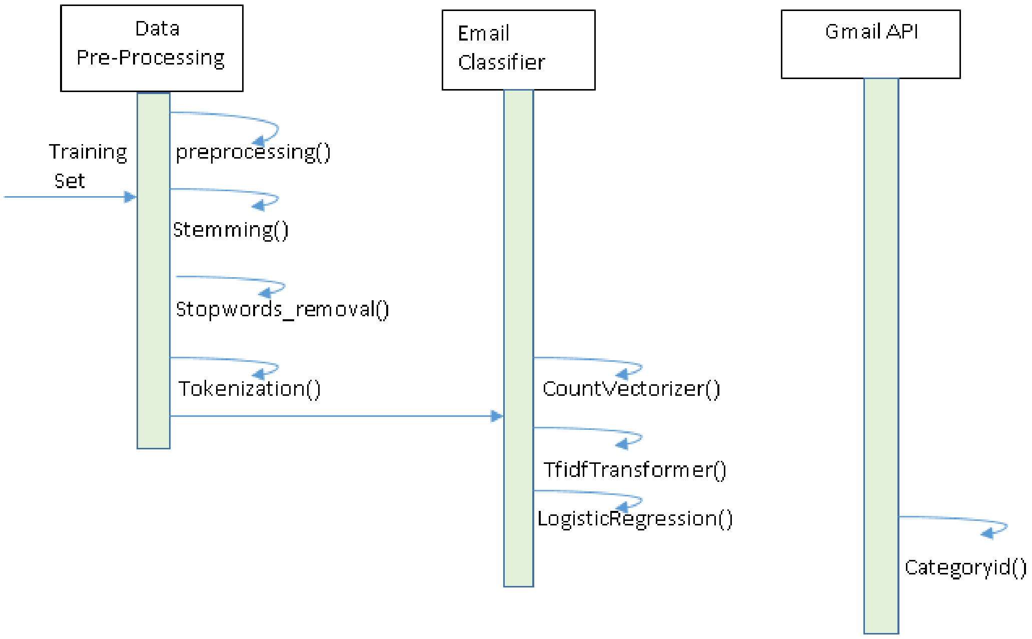


Figure 5 - Sequence Diagram of DWIT Email Classifier

CHAPTER 4: IMPLEMENATION AND TESTING

4.1 Implementation

The main purpose of implementation is to be able to automatically classify if the new unseen email is club email, class email, admin email or miscellaneous email. This application consists of a list of manually classified emails labelled with its' four categories: club, class, admin or misc. Based on the polarity of each words the polarity of the whole document is calculated and are stored in the feature vector.

Sample test sentences:

1. Mini Project 2017 Batch Please upload your documents in the link shared. **Class email**.
2. Internship Opportunity - Only for Class of 2019 Please write a proper email expressing why college should pick you up for this task/opportunity. **Admin email**.
3. IT Club Computers not available in lab. **Club email**.
4. Chautariko Basai | Episode 7 Please feel free to send your articles and feedback to our address chautariko.basai@deerwalk.edu.np. **Miscellaneous email**.

To implement the project, the main focus was to compare the polarity of the input email with a set of words that are stored in the system with their respective polarities. This project uses CSV file that contains of 4000 categorized email datasets. When the user submits the input email, the email goes through tokenization, which creates a. list of tokens. The list of tokens are then pre-processed to get rid of the stop-words and stems. Those processed list of words are then compared with the file that stores the list of words with their respective polarities.

Then, the polarity of each words is calculated. With the help of that, the total polarity of the input email is calculated. The email is then classified into a class which has maximum polarity.

This means that if we provide the email, “As communicated in last class, for the final chapter I am dividing topic for presentation. So here is the list of topic and presenter.”, then the system goes through the above mentioned process to determine that the email is of class emails category.

4.1.1 Tools used

CASE tools:

a) Draw.io

Programming Language

a. Python

4.2 Listing of major classes

a) Preprocessor

```
# preprocessing training set text by removing non-words from training set text
def preprocessor(text):
    text = re.sub('[\W]+', ' ', text)
    return text
```

b) Remove_stop_words

```
def remove_stop_words(text):
    return [w for w in text.split() if w not in stop]
```

c) Porter Stemmer

```
porter = PorterStemmer()
df['emails'] = df['emails'].apply(remove_stop_words)
print("df['emails']----->>>", df['emails'])
```

```
val = 0
for x in df['emails']:
    df['emails'][val] = [porter.stem(word) for word in x]
    text = ""
    for word in df['emails'][val]:
        text = text + word + " "
    df['emails'][val] = text
    val += 1
print(df['emails'])
```

d) CountVextorizer and Tfidfransformer

```
# testing usage of CountVextorizer and Tfidfransformer which computes tfidf
count = CountVectorizer()
bag = count.fit_transform(df['emails'])
print("frequency of words--->>", count.vocabulary_)

print("Data in numeric form-->>\n" ,bag.toarray())

tfidf = TfidfTransformer()
np.set_printoptions(precision=2)
print("TFIDF Vector ---->>\n",
tfidf.fit_transform(count.fit_transform(df['emails'])).toarray())

# Applying logistic regression classifier
text_clf = Pipeline([('vect', CountVectorizer()), ('tfidf', TfidfTransformer()), ('clf',
LogisticRegression()), ])

_ = text_clf.fit(df['emails'], df['category'])

# get predicted values
predicted = text_clf.predict(tf['emails'])
print("Predicted labels-->" ,predicted)
```

4.3 Testing

Other than evaluating and validating the classifier's performance, different test cases were created in order to make sure that the system delivers the required output and functions properly. These test cases ensured the validity and reliability of the entire system. Unit and were performed on the system components.

4.3.1 Unit Testing

Each unit of the system was tested for its correct and proper functionality.

Table 2 - Unit testing of different components

Test no.	Unit	Test	Expected Result	Test Outcome
1	Preprocessor	Trimming delimiter symbols	Remove delimiter from input text	Successful
2	Stop word removal	Removing stop words	Remove stop words from input text	Successful
4	CountVectorizer	Constructs the bag-ofwords Model	Takes an array of sentences and Constructs the bag-ofwords	Successful
5	TfidfTransformer	Calculates the tf-idfs	Calculates the tf-idfs of the documents	Successful

CHAPTER 5: MAINTENANCE AND SUPPORT PLAN

DWIT Email Classifier will implement corrective maintenance for resolving different bugs and errors that may occur when this project is made live. Perfective maintenance will be implemented for increasing efficiency of the DWIT Email Classifier by optimizing various implementation methods. Preventive maintenance will be implemented to make sure that DWIT Email Classifier will not be harmed by hackers and security mechanism will be added.

CHAPTER 6: CONCLUSION AND RECOMMENDATION

6.1 Conclusion

DWIT Email Classifier was successfully implemented as a Gmail plugin in Chrome. From the testing carried out the accuracy obtained is 70%. Hence, use of email classification system that implements Logistic Regression algorithm is not recommended and further improvements need to be made to be used for classification purpose.

6.2 Recommendation

This project uses the students email set as a training dataset so the accuracy varies among the type of users; faculty, students, admins. Hence, the wide variety of heterogeneous email datasets needs to be implemented for increasing accuracy of email classification system.

REFERENCES

José M. Carmona-Cejudo, M. B.-G. (2011). *Using GNUsmail to compare data stream mining methods for on-line email classification.*

Ron Bekkerman, A. M. (2004). *Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora.*

Svetlana Kiritchenko, S. M. (2001). Email classification with co-training. *CASCON '01: Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative research.*

SEM VI – Project Documents Modification

To

The Examination Section

DWIT College

As per the suggestions provided during the midterm documentation review of my project on December 18, I have made following changes.

Suggestions	Changed Location [Mention new document page number, topic id and paragraph number]	Reference [If any]	Remark [sign of supervisor]
Project Title should be explanatory	Cover Page		
Formatting of the Table of Contents	Table of Contents		
Formatting of the List of Figures	List of Figures		
Formatting of the List of Tables	List of Tables		
Each Chapters should begin in new page	Each new chapters		
Citations in Literature Review	Literature Review (2.1)		
References	Page 30		

……………………………………………..

Asmita Subedi