PANIMALAR INSTITUTE OF TECHNOLOGY
CHENNAI – 600 123

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

AD8512 MINI PROJECT ON DATA SCIENCES PIPELINE

DETERMINATION OF SPAM EMAILS BY PARTICLE SWARM OPTIMIZATION WITH TENSOR FLOW USING MULTINOMIAL NAIVE BAYES

A MINI PROJECT REPORT Submitted by

RAVIRAJAN SR (211520243044)
VIGNESH S (211520243058)

in the fifth semester of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

During the academic year 2022 - 23

JANUARY 2023

PANIMALAR INSTITUTE OF TECHNOLOGY CHENNAI – 600 123

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

AD8512 MINI PROJECT ON DATA SCIENCES PIPELINE

BONAFIDE CERTIFICATE

Certified that this project report "**Determination Of Spam Emails By Particle Swarm Optimization With Tensor Flow Using Multinomial Naive Bayes**" is the bonafide work of RAVIRAJAN SR (211520243044) and VIGNESH S (211520243058) of third year B.Tech – AI & DS during the academic year 2022 –23 who carried out under my supervision.

SIGNATURE
Mrs. K. SARANYA M. Tech,
ASSISTANT PROFESSOR,
SUPERVISOR

SIGNATURE
Dr. T. KALAICHELVI, M.E., Ph.D.
HEAD OF DEPARTMENT

Certified that the above candidates were examined in the university **AD8512 Mini Project on Data Sciences Pipeline** viva-voce held on_____at Panimalar Institute of Technology, Chennai – 600 123.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

A project of this magnitude and nature requires the kind cooperation and support of many, for successful completion. We wish to express our sincere thanks to all those who were involved in the completion of this project.

We would like to express our deep gratitude to Our **Beloved Secretary and Correspondent, Dr. P. CHINNADURAI, M.A., Ph.D.,** for his kind words and enthusiastic motivation which inspired us a lot in completing the project.

We also express our sincere thanks to Our **Dynamic Directors Mrs.C.VIJAYARAJESWARI, Dr. C. SAKTHIKUMAR, M.E., Ph.D.,** and **Dr.**
S. SARANYA SREE SAKTHIKUMAR, B.E., M.B.A., Ph.D for Providing us with the necessary facilities for the completion of this project.

We also express our gratefulness to our **Principal Dr.T.JAYANTHY, M.E.,Ph.D.,** who helped us in the completion of this project.

We wish to convey our thanks and gratitude to our **Head of the Department Dr. T. KALAICHELVI, M.E., Ph.D.,** Department of Artificial Intelligence and Data Science, for her support and providing us ample time to complete our project.

We express our indebtedness and gratitude to our **Assistant Professor, Mrs. K. SARANYA M. Tech** of Artificial Intelligence and Data Science, for his guidance throughout the project.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF FIGURES	iv
	LISTOFABBREVIATIONS	v
1.	**INTRODUCTION**	1
	1.1 Introduction to spam classification	1
	1.2 Aim and Project Scope	2
	1.3 Project Synopsis	3
2.	**LITERATURE SURVEY**	4
	2.1 Literature survey	4
3.	**SYSTEMANALYSIS**	6
	3.1 Existing System	6
	3.1.1 Limitations	6
	3.2 Proposed System	6
	3.3 Requirement Analysis	7
	3.3.1 Hardware Requirements	7
	3.3.2 Software Requirements	8
4.	**SOFTWAREDESCRIPTION**	9
	4.1 Jupyter Notebook	9
	4.1.1 Syntax	9
5.	**SYSTEM DESIGN**	12
	5.1 Architecture of the system	12
	5.2 Dataflow diagram	12
	5.3 UML Diagram	13
	5.3.1 Use Case Diagram	14
	5.3.2 Activity Diagram	15

6.	**MODULES**	16
	6.1 Module Description	16
	6.1.1 Exploratory data analysis	16
	6.1.2 Data Preprocessing	24
	6.1.3 Feature Extraction	25
	6.1.4 Scoring And Metrics	26
7.	**ALGORITHM AND PERFORMANCE**	27
	7.1 Algorithm	27
	7.1.1 Naïve Bayes	27
	7.1.2 Algorithm Steps	27
8.	**CONCLUSION AND FUTURE ENHANCEMENTS**	29
9.	**REFERENCES**	30
10.	**APPENDIX**	31

LIST OF FIGURES

FIG NO	NAME OF THE FIGURE	PAGE NO
1.1	Classifier Diagram	1
4.1	Jupyter NoteBook	11
5.1	Architecture of the system	3
5.2	Dataflow Diagram	5
5.3	UML Diagram	21
	5.3.1 Use case Diagram	23
	5.3.1 Activity Diagram	24
6.1	Visualization of spam email	29
6.2	Visualization of non-spam email	41
6.3	Bar chart visualization of 1-gram model	42
6.4	Bar chart visualization of 2-gram model	43
6.5	Target Count For Train Data	44
6.6	Train Data Distribution	46
6.7	Target Count For Test Data	47
6.8	Train Data Distribution	48

LIST OF ABBREVIATIONS

S. No	Abbreviation	Expansion
1	SVM	Support Vector Machines
2	KNN	K-Nearest neighbor
3	EDA	Exploratory Data Analysis
4	NB	Naive Bayes
5	BIC	Bayesian Information Criterion
6	MRI	Magnetic resonance imaging

ABSTRACT

Spam detection is considered to be one of the most challenging issues in Online Social Networks (OSNs). In this paper, a supervised method is used to detect spam on these platforms. The accuracy of supervised methods depended on two factors: (i) the desired feature selection and (ii) the use of an appropriate classifier. An innovative method is also used for the first factor. This method is a combination of association rule mining and genetic algorithm with the aim of the desired feature selection from a variety of features. On the other hand, the second factor uses a large number of popular classifiers. The proposed method is assessed on three datasets, and the results show the effectiveness of the proposed feature selection method on the accuracy of the classifiers. The average accuracy for both approaches compared to the basic methods is 87.99% and 95.24%, respectively.

Keywords: Spam Detection ,Online Social Network,Feature Selection, ,GeneticAlgorithm,Association Rule Mining

.

CHAPTER 1

INTRODUCTION

1.1. INTRODUCTION TO SPAM FILTERING:

Spam definition :

Spam is any kind of unwanted, unsolicited digital communication that gets sent out in bulk. Often spam is sent via email, but it can also be distributed via text messages, phone calls, or social media.

What does spam stand for?

Spam is not an acronym for a computer threat, although some have been proposed (stupid pointless annoying malware, for instance). The inspiration for using the term “spam” to describe mass unwanted messages is a Monty Python skit in which the actors declare that everyone must eat the food Spam, whether they want it or not. Similarly, everyone with an email address must unfortunately be bothered by spam messages, whether we like it or not.

What is a spam filter?

A spam filter is a program used to detect unsolicited, unwanted and virus-infected emails and prevent those messages from getting to a user's inbox. Like other types of filtering programs, a spam filter looks for specific criteria on which to base its judgment

For example, whenever users mark emails from a specific sender as spam, the Bayesian filter recognizes the pattern and automatically moves future emails from that sender to the spam folder.

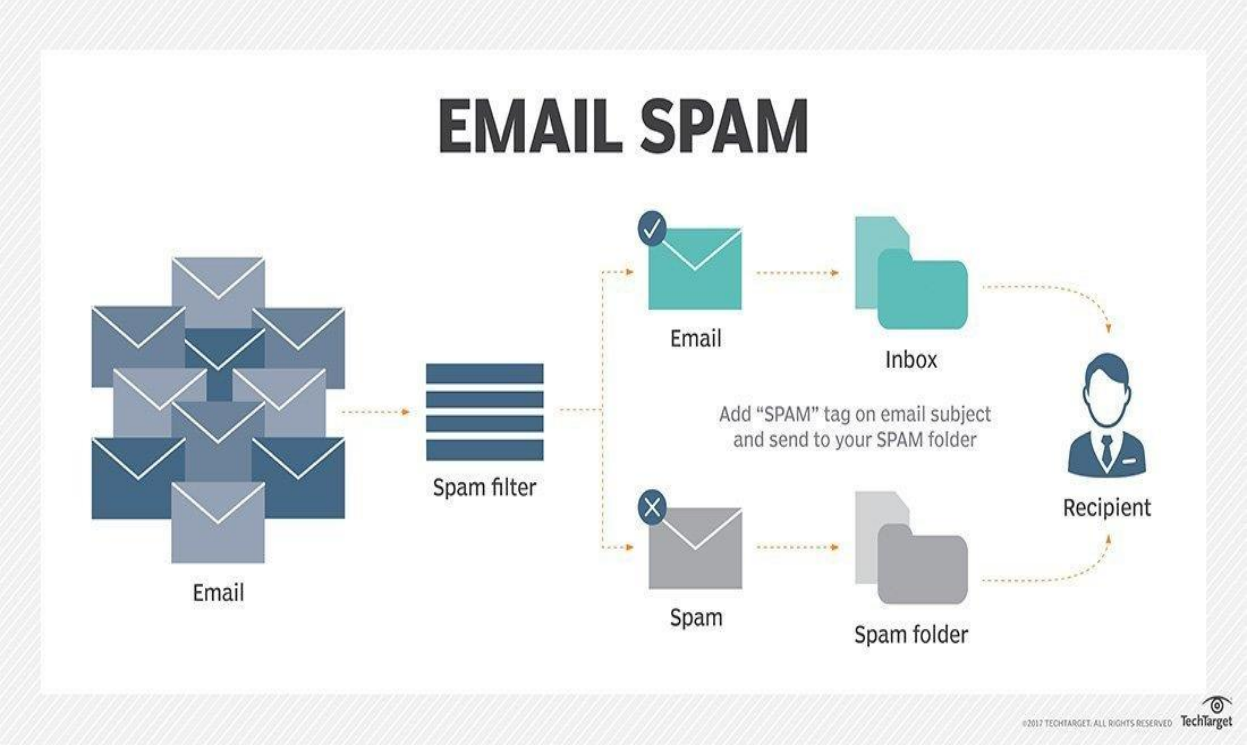

FIG 1.1: *Classifier diagram*

1.2. AIM AND PROJECT SCOPE:

It provides sensitivity to the client and adapts well to the future spam techniques. It considersa complete message instead of a single word with respect to its organization. It increases security and control it. Reduces IT Administration costs and also Network Resource Costs. The project's suggested solution will successfully identify spam emails and remove spam emails using a few machine learning algorithms, producing results with higher accuracy and better performance. Additionally, by blocking and removing spam emails, the suggested system will optimise data storage. The proposed system will also carry out the filtering of spam messages and find out whether the mail is reliable with the aid of the Opinion Rank model. The user will benefit from time savings and a reduction in spam mail risk thanks to the suggested solution.

1.3. PROJECT SYNOPSIS:

Spam e-mails containing unwanted and irrelevant product promotions sent by spammers have a tendency to frequently flood the inbox and create storage issues for other relevant and important e-mails. E- mails carrying information such as commercial offers, lotteries, bank- related offers, etc, are most likely to be sent by scammers who impart information that can fool a certain number of gullible users, and extort money from them. Moreover, such e-mails may contain viruses that severely affect the user's device.

These activities by spammers and scammers pose major concerns because information conveyed by spam and non-spam emails are perceived differently by different users and even though a vast majority of users are able to identify spam emails and are aware of threats posed by them, a significant amount of users who receive spam e-mails are unaware of and respond to such e-mails be- coming a viable source of income for the sender.

This is why detection of spam e-mails has become crucial in this fast evolving and technology- dependent world. In this study, different spam-detection methodologies are established and employed for classification between spam e-mails and regular (legitimate) emails. The classification method of Naïve bayes theorem is used in this study because the frequency of occurrence of spam and legitimate e-mails are seen to be comparable in the data set employed.

CHAPTER 2

LITERATURE SURVEY

2.1. LITERATURE SURVEY

1.PAPER TITLE : MACHINE LEARNING METHODS FOR SPAM E-MAIL CLASSIFICATION

AUTHORS: ***W.A. Awad1 and S.M. ELseuofi2***

The increasing volume of unsolicited bulk e-mail (also known as spam) has generated a need for reliable anti-spam filters. Machine learning techniques now days used to automatically filter the spam e-mail in a very successful rate. In this paper we review some of the most popular machine learning methods (Bayesian classification, k-NN, ANNs, SVMs, Artificial immune system and Rough sets) and of their applicability to the problem of spam Email classification. Descriptions of the algorithms are presented, and the comparison of their performance on the SpamAssassin spam corpus is presented.

MERITS: Increase the attractiveness of these catalogs to customers to promote a recommendation system

DEMERITS: Decreases the amount of bundled product information

2. **.PAPER TITLE: A Hybrid BSO-Chi2-SVM Approach to Arabic Text Categorization**

AUTHORS: Riadh Belkebir, Ahmed Guessoum

Automatic categorization of documents has become an important task, especially with the rapid growth of the number of documents available online. Automatic categorization of documents consists in assigning a category to a text based on the information it contains. It aims to automate the association of a document with a category. Automatic categorization can allow solving several problems such as identifying the language of a document, the filtering and detection of spam (junk mail), the routing and forwarding of emails to their recipients, etc. In this paper, we present the results of Arabic text categorization based on three different approaches: artificial neural networks, support vector machines (SVMs) and a hybrid approach BSOCHI-SVM. We explain the approach and present the results of the implementation and evaluation using two types of representations: root-based stemming and light stemming. The evaluation in each case was done on the Open Source Arabic Corpora (OSAC) using different performance measures.

MERITS: The classification methods can be adapted to big data analytics framework

DEMERITS: This article does not contain any studies with human participants performed by any of the authors.

CHAPTER 3

SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

Separation of spam e-mails from legitimate e-mails is a classification problem for which meta-heuristic solutions are proposed in this study in the form of BIC algorithms integrated with k-NN. The results obtained show 100% mean values for accuracy, precision, recall and F1-measure when Manhattan distance is used for classification in k-NN, which makes it the most suitable distance metric for classification purposes. The No Free Lunch (NFL) Theorem (Wolpert and Macready, 1997) proposes lack of suitability of a specific meta-heuristic approach for solving all the issues identified in different fields of optimization. GOA provides the highest average accuracy of 74%, FOA converges to global optimal solution the fastest, WOA provides the highest average value of precision (69.83%) and F1-measure (74.46%), and takes the least computation time (99.67 s), followed by CSO (105.67 s) and GWO (681.33 s), both of which have a recall of 100%.

3.1.1. LIMITATIONS:

The biggest drawback of utilising tf/idf is that it clusters texts based on keyword similarity, making it useful only to find documents that are nearly identical.
Because TF-IDF is based on the bag-of-words (BoW) paradigm, it
is unable to account for factors such as text location, semantics,
co-occurrences across texts, etc.

3.2. PROPOSED SYSTEM:

In this system, to solve the problem of spam, the spam classification system (naïve bayes classifier) is created to identify spam and non-spam. Since spammers may send spam messages many times, it is difficult to identify it every time manually. So, we will be using some of the strategies in our proposed system to detect the spam. The proposed solution not only identifies the spam word but also identifies the IP address of the system through which the spam message is sent so that next time when the spam message is sent from the same system our proposed system directly identifies it as blacklisted based on the IP address.

3.1. REQUIREMENTS ANALYSIS:

3.1.1. HARDWARE REUIREMENTS

Sample hardware configurations are as follows: For a server handling 20,000 emails/day, a 500MHz CPU and 512MB of RAM is the minimum recommended. For a server handling 200,000 emails/day, a dual-core 2GHz Xeon CPU and4GBMB of RAM is the minimum recommended.

HDD : >90GB

PROCESSOR : >Pentium IV 2.4GHz

RAM : >2GB

OS : Windows 7/8/8.1/10

3.1.2. SOFTWARE REQUIREMENTS

The software specification are the specification of the system. It should include both the specification and a definition of the requirements. It is a set of what the system should do rather than how it should do it. The software requirements provides the basis for creating the software requirement specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's progress throughout the development activity.

DATASET : mail dataset

Tool : JUPYTER NOTES

CHAPTER 4

SOFTWARE DESCRIPTION

4.1. JUPYTER NOTEBOOK:

Jupyter Notebook is an open-source, web-based interactive environment, which allows you to create and share documents that contain live code, mathematical equations, graphics, maps, plots, visualizations, and narrative text. It integrates with many programming languages like Python, PHP, R, C#, etc.

4.1.1 SYNTAX:

A representation of all content viewable in the web application, including calculation inputs and outputs, explanatory language, math, graphics, and rich media representations of objects are included in notebook papers.

- **Creating a new notebook document**

A new notebook may be created at any time, either from the dashboard, or using the File ‣ New menu option from within an active notebook. The new notebook is created within the same directory and will open in a new browser tab. It will also be reflected as a new entry in the notebook list on the dashboard.

- **Opening notebooks**

An open notebook has exactly one interactive session connected to a kernel, which will execute code sent by the user and communicate back results. This kernel remains active if the web browser window is closed, and reopening the same notebook from the dashboard will reconnect the web application to the same kernel. In the dashboard, notebooks with an active kernel have a Shutdown button next to them, whereas notebooks without an active kernel have a Delete button in its place.

Other clients may connect to the same kernel. When each kernel is started, the notebook server prints to the terminal a message like this:

[NotebookApp] Kernel started: 87f7d2c0-13e3-43df-8bb8-1bd37aaf3373

This long string is the kernel's ID which is sufficient for getting the information necessary to connect to the kernel. If the notebook uses the IPython kernel, you can also see this connection data by running the %connect_info magic, which will print the same ID information along with other details.

You can then, for example, manually start a Qt console connected to the same kernel from the command line, by passing a portion of the ID:

$ jupyterqtconsole --existing 87f7d2c0

Without an ID, --existing will connect to the most recently started kernel.

With the IPython kernel, you can also run the %qtconsole magic in the notebook to open a Qt console connected to the same kernel.

- **Notebook user interface**

- ***FIG 5:*** *Jupyter Notebook*

When you create a new notebook document, you will be presented with the notebook name, a menu bar, a toolbar and an empty code cell

Notebook name: The name displayed at the top of the page, next to the Jupyter logo, reflects the name of the .ipynb file. Clicking on the notebook name brings up a dialog which allows you to rename it. Thus, renaming a notebook from "Untitled0" to "My first notebook" in the browser, renames the Untitled0.ipynb file to My first notebook.ipynb.

Menu bar: The menu bar presents different options that may be used to manipulate the way the notebook functions.

Toolbar: The tool bar gives a quick way of performing the most-used operations within the notebook, by clicking on an icon.

Code cell: the default type of cell; read on for an explanation of cells

CHAPTER 5

SYSTEM DESIGN

5.1. ARCHITECTURE DIAGRAM:

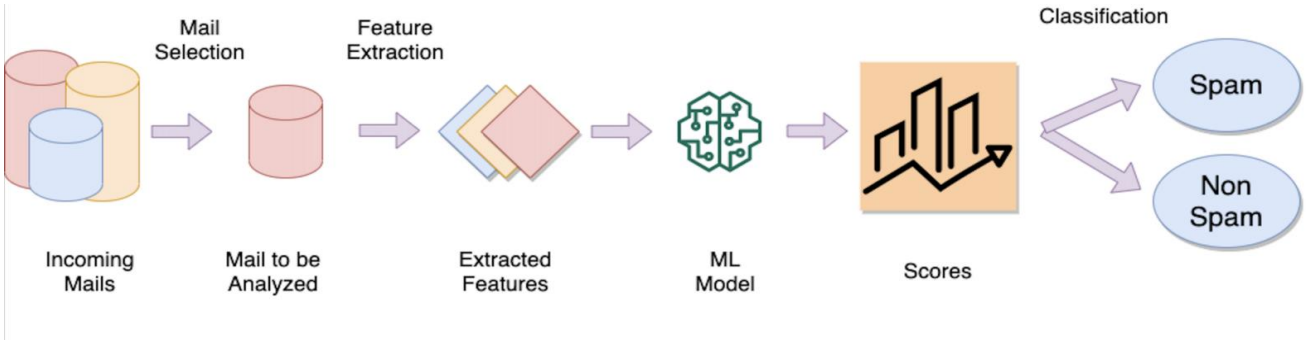

FIG 5.1: *Architecture of the system*

5.2.DATAFLOW DIAGRAMS:

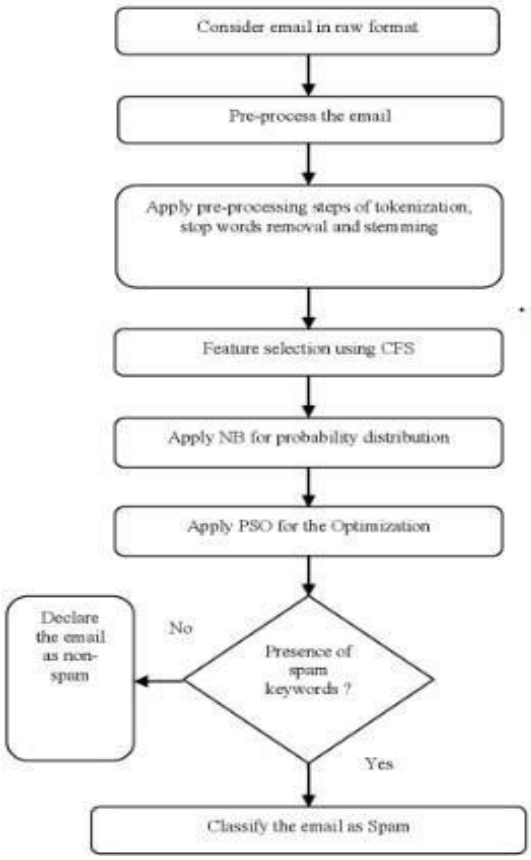

FIG 5.2: *Dataflow diagram*

5.3.UML DIAGRAM:

A Class diagram is a Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modelling.

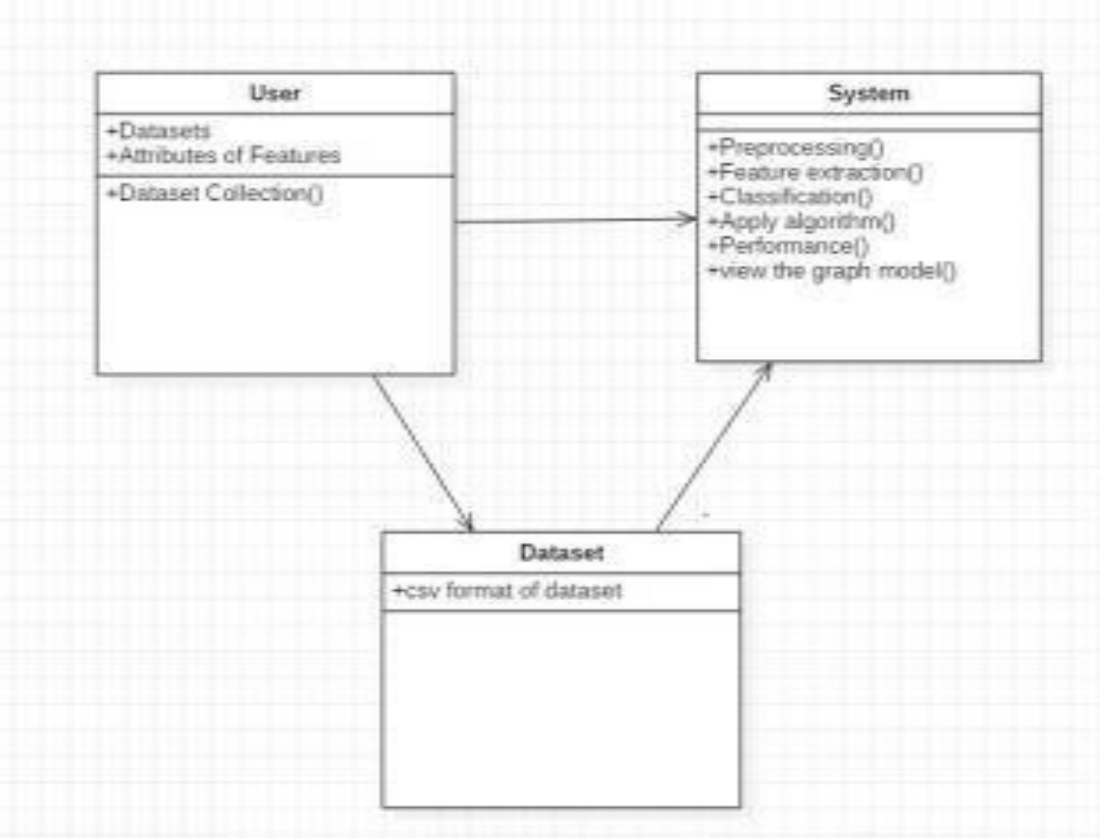

FIG 5.3: *UML Diagram*

5.3.1 USE CASE DIAGRAM:

A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site.

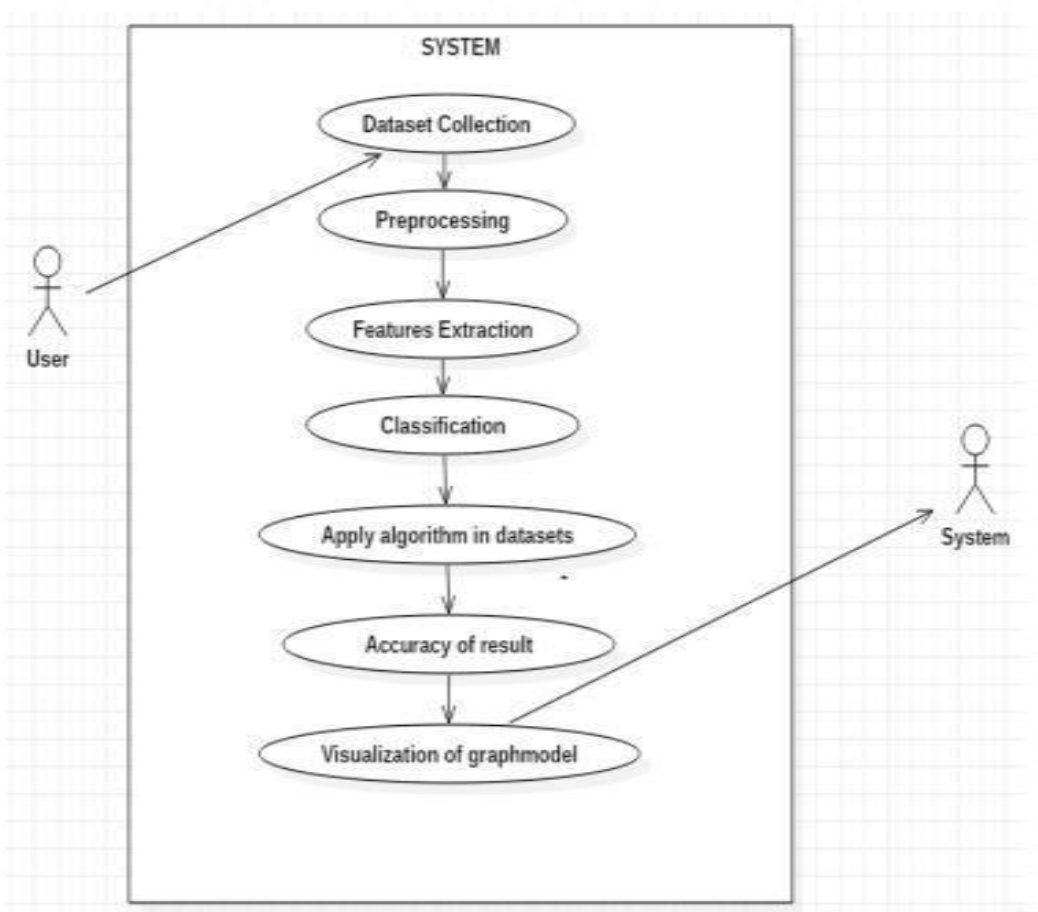

FIG5.3.1 :*Use Case Diagram*

5.3.2 ACTIVITY DIAGRAM:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity.

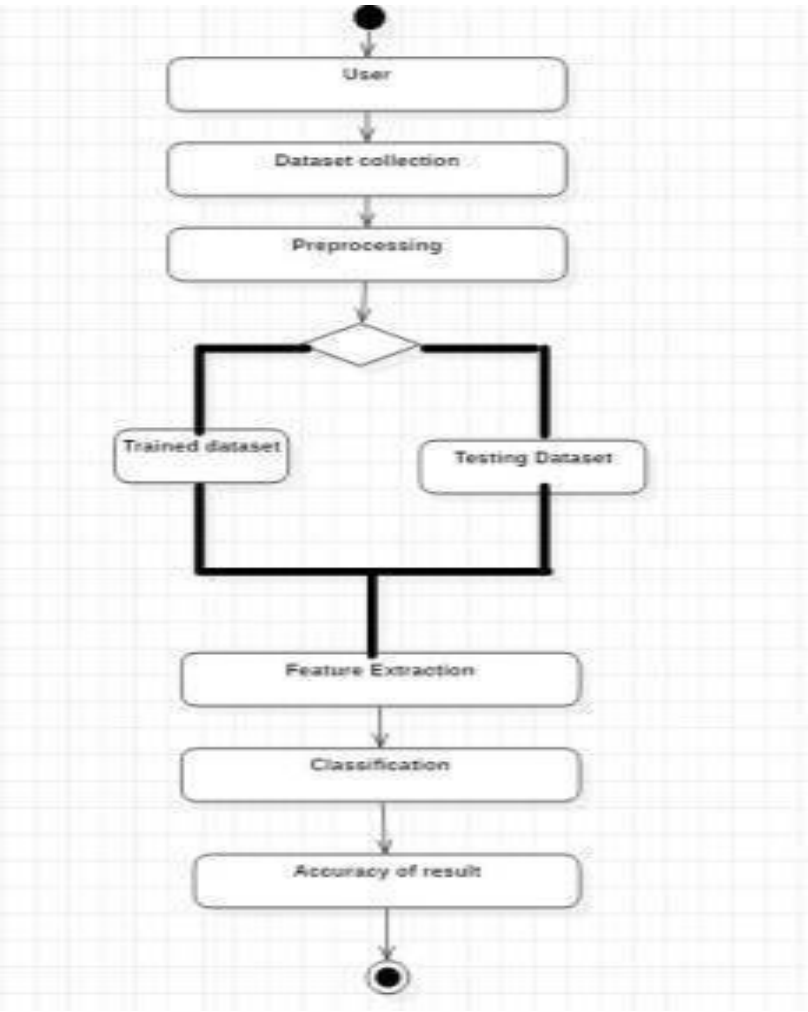

FIG5.3.2 :*Activity Diagram*

CHAPTER 6

MODULES

6.1. MODULE DESCRIPTION:

As a software developer, email is one of the very important tool for communication. To have effective communication, spam filtering is one of the important feature..

Outlines summarized as below:

1. EDA (Exploratory data analysis)
2. Feature Extraction
3. Scoring & Metrics

6.1.1.EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis is a very important process of data science. It helps the data scientist to understand the data at hand and relates it with the business context.

The open source tools that I will be using in visualizing and analyzing my data is Word Cloud.

Word Cloud is a data visualization tool used for representing text data. The size of the texts in the image represent the frequency or importance of the words in the training data.

Steps to take in this section:

1) Get the email data
2) Explore and analyze the data
3) Visualize the training data with Word Cloud & Bar Chart

Get the spam data

Data is the essential ingredients before we can develop any meaningful algorithm. Knowing where to get your data can be a very handy tool especially when you are just a beginner.

For this email spamming data set, it is distributed by Spam Assassin, you can click this link to go to the data set. There are a few categories of the data, you can read the readme.html to get more background information on the data.

In short, there is two types of data present in this repository, which is ham (non-spam) and spam data. Furthermore, in the ham data, there are easy and hard, which mean there is some non-spam data that has a very high similarity with spam data. This might pose a difficulty for our system to make a decision.

Explore and analyze the data

Ham

This looks like a normal email reply to another person, which is not difficult to classified as a ham:

Hard Ham (Ham email
that is trickier) Hard Ham is indeed more

```
This is a bit of a messy solution but might be useful -

If you have an internal zip drive (not sure about external)
and you bios supports using a zip as floppy drive, you
could use a bootable zip disk with all the relevant dos
utils.
```

difficult to differentiate from the spam data, as they contain some key words such as *limited time order*, *Special "Back to School" Offer*, this makes it very suspicious!

Spam

One of the spam training data does look like one of those spam advertisement email in our junk folder:

```
IMPORTANT                                                    INFORMATION:
The new domain names are finally available to the general public at discount
prices. Now you can register one of the exciting new .BIZ or .INFO domain names,
as well as the original .COM and .NET names for just $14.95. These brand new
domain extensions were recently approved by ICANN and have the same rights as
the original .COM and .NET domain names. The biggest benefit is of-course that
the .BIZ and .INFO domain names are currently more available. i.e. it will be
much easier to register an attractive and easy-to-remember domain name for the
same price.  Visit: http://www.affordable-domains.com today for more info.
```

Visualization

Wordcloud

Wordcloud is a useful visualization tool for you to have a rough estimate of the words that has the highest frequency in the data that you have.

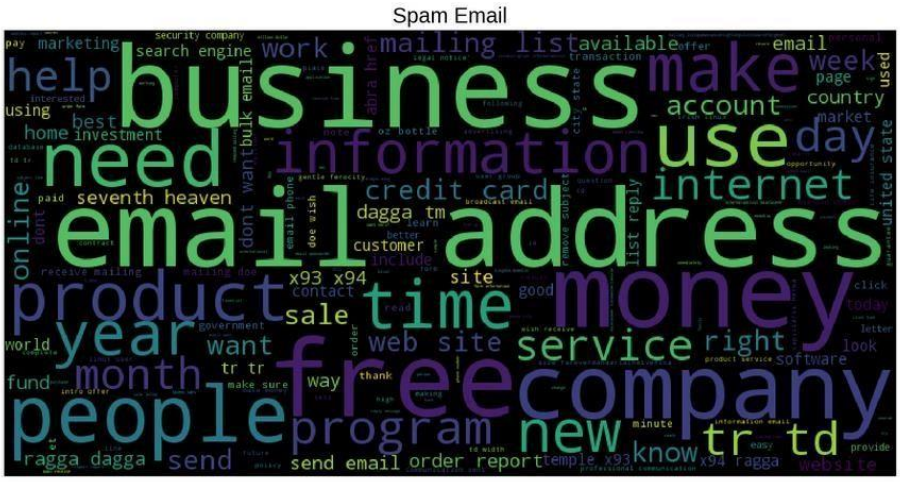

FIG 6.1. :*Visualization for spam email*

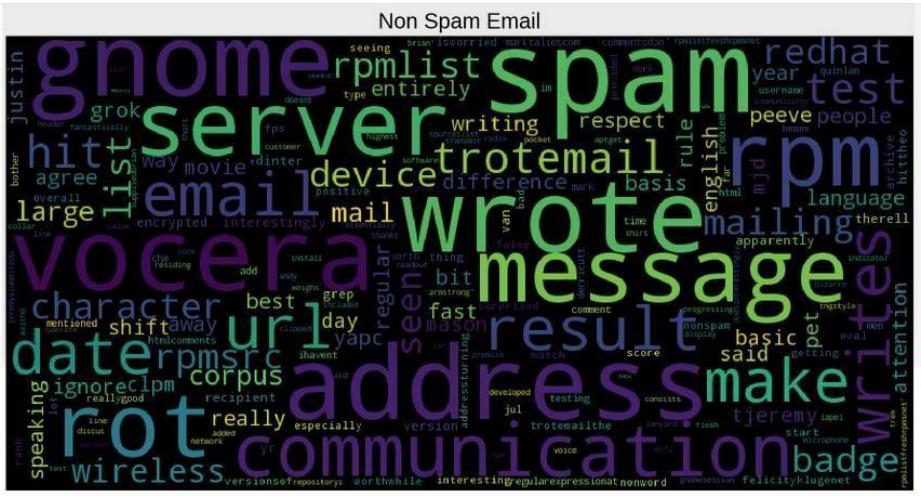

FIG 6.2 :*Visualization for non spam email*

From this visualization, you can notice something interesting about the spam email. A lot of them are having high number of “spammy” words such as: free, money, product etc. Having this awareness might help us to make better decision when it comes to designing the spam detection system.

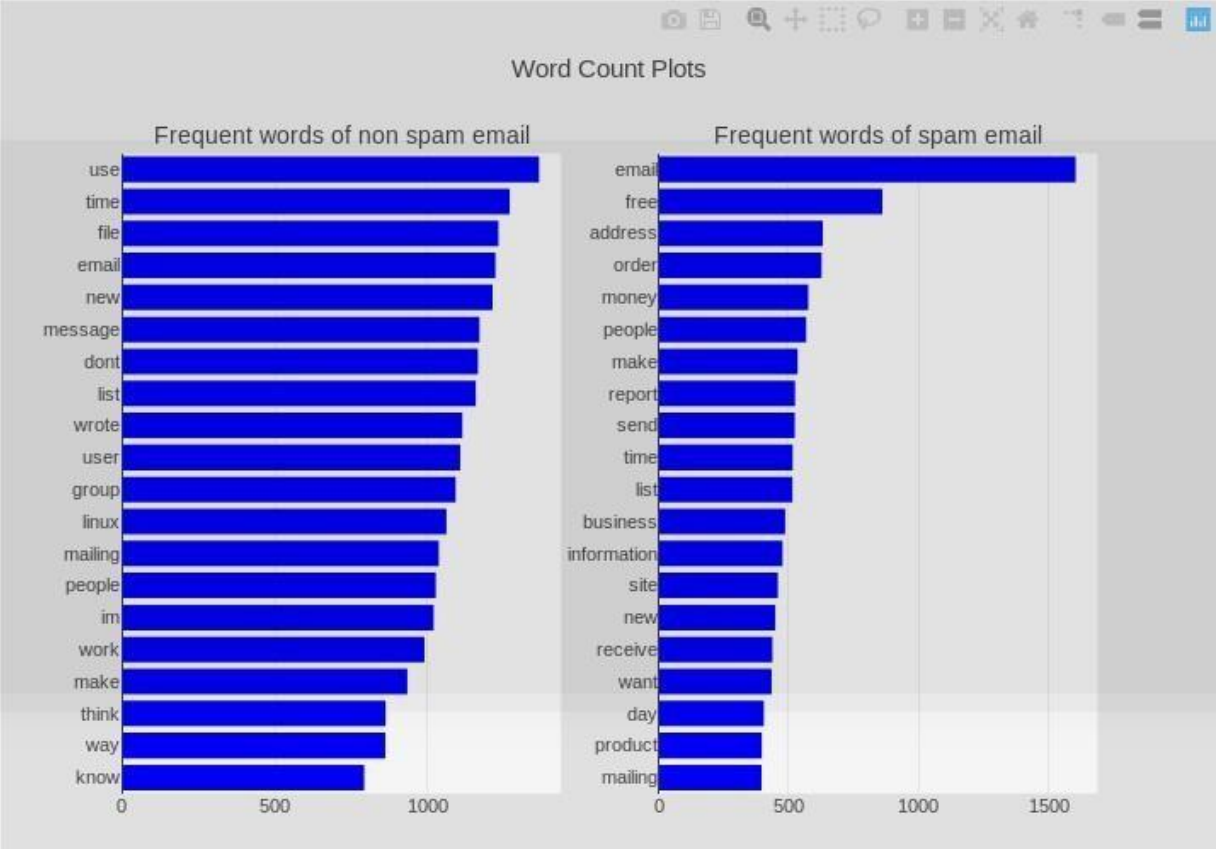
Word Count Plots
Frequent words of non spam email
use
time
file
email
new
message
dont
list
wrote
user
group
linux
mailing
people
im
work
make
think
way
know
0
500
1000
Frequent words of spam email
email
free
address
order
money
people
make
report
send
time
list
business
information
site
new
receive
want
day
product
mailing
0
500
1000
1500

FIG 6.3 :*Bar chart visualization of 1-gram model*

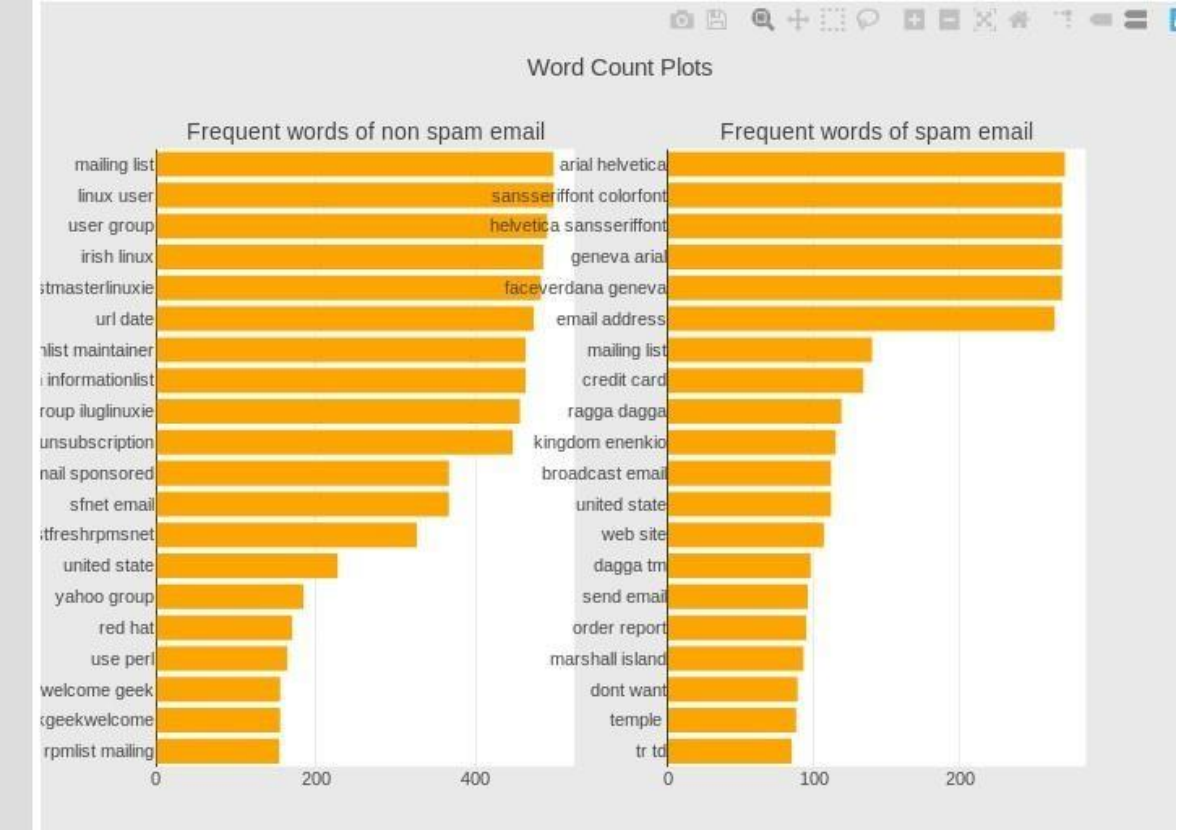

FIG 6.4:*Bar chart visualization of 2-gram model*

Train Test Split

It is important to split your data set to training set and test set, so that you can evaluate the performance of your model using the test set before deploying it in a production environment.

One important thing to note when doing the train test split is to make sure the distribution of the data between the training set and testing set are similar.

What it means in this context is that the percentage of spam email in the training set and test set should be similar.

FIG 6.5 :*Target Count For Train Data*

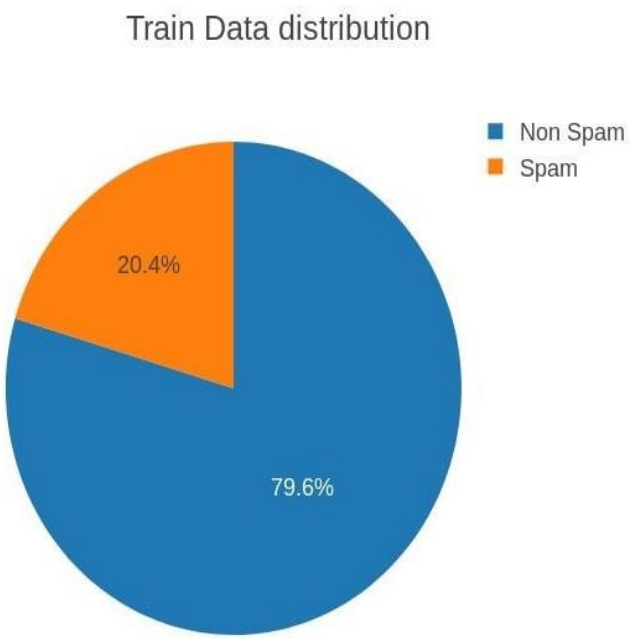

***FIG 6.6**: Train Data Distribution*

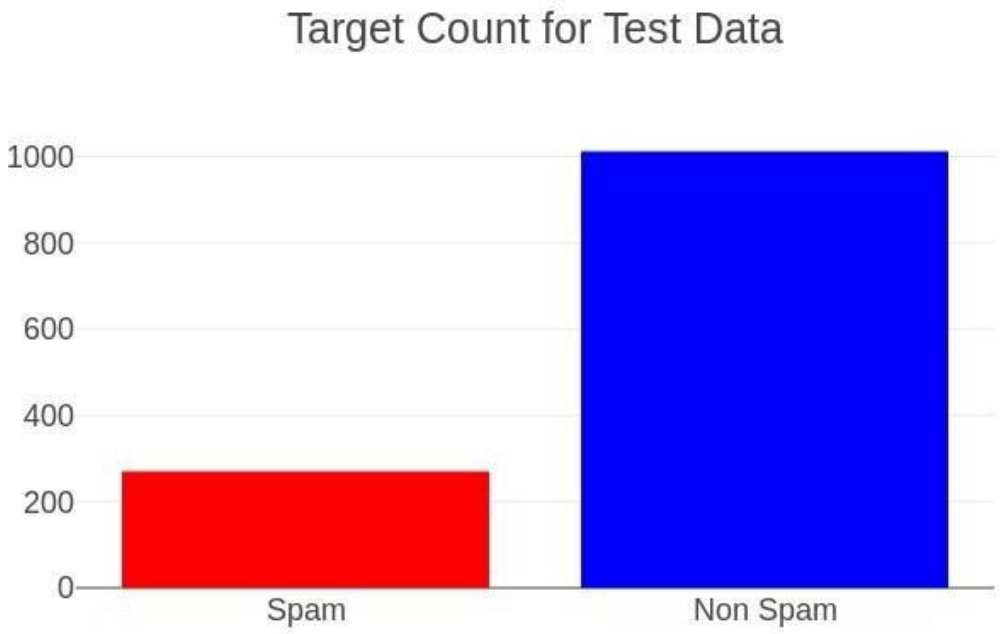

***FIG 6.7:** Target Count For Test Data*

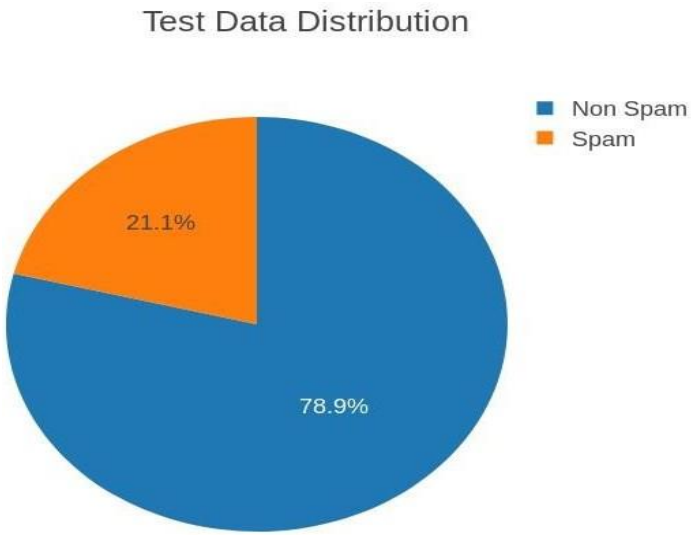

***FIG 6.8:**Test Data Distribution*

The distribution between train data and test data are quite similar which is around 20–21%, so we are good to go and start to process our data!

6.1.2.DATA PREPROCESSING

Text Cleaning

Text Cleaning is a very important step in machine learning because your data may contain a lot of noise and unwanted character such as punctuation, white space, numbers, hyperlink and etc.

Some standard procedures that people generally use are:

- convert all letters to lower/upper case
- removing numbers
- removing punctuation
- removing white spaces
- removing hyperlink
- removing stop words such as a, about, above, down, doing and the list goes on…
- **Word Stemming**
- **Word lemmatization**

The two techniques that might seem foreign to most people are word stemming and word lemmatization. Both these techniques are trying to reduce the words to its most basic form, but doing this with different approaches.

- Word stemming — Stemming algorithms work by removing the end or the beginning of the words, using a list of common prefixes and suffixes that can be found in that language.

- Word Lemmatization — Lemmatization is utilizing the dictionary of a particular language and tried to convert the words back to its base form. It will try to take into account of the meaning of the verbs and convert it back to the most suitable base form.

6.1.3.FEATURE EXTRACTION

Our algorithm always expect the input to be integers/floats, so we need to have some feature extraction layer in the middle to convert the words to integers/floats.

There are a couples ways of doing this, and today I am going to introduce:

1. CountVectorizer
2. TfidfVectorizer
3. Word Embedding

CountVectorizer

First we need to input all the training data into CountVectorizer and the CountVectorizer will keep a dictionary of every word and its respective id and this id will relate to the word count of this word inside this whole training set.

TfidfVectorizer

One issue with simple word count is that some words like ‘the’, ‘and’ will appear many times and they don’t really add too much meaningful information.Another popular alternative is TfidfVectorizer. Besides of taking the word count of every words, words that often appears across multiple documents or sentences, the vectorizer will try to downscale them.

Word Embedding

Word embedding is trying to convert a word to a vectorized format and this vector represents the position of this word in a higher dimensional space.

For words that have similar meaning, the cosine distance of those two word vectors will be shorter and they will be closer to each other.

Example 1 : King- Man + Woman = Queen

Example 2: Madrid-Spain+France = Paris

Example 3: Walking-Swimming+Swam= Walked

Simply put, word embedding is a very powerful representation of the words and one of the well known techniques in generating this embedding is Word2Vec.

6.1.4 SCORING AND METRICS

Algorithm Implementation

TfidfVectorizer + Naive Bayes Algorithm

Precision & Recall

Precision & Recall is the common evaluation metrics that people use when they are evaluating class-imbalanced classification model.

Confusion Matrix

Confusion Matrix is a very good way to understand results like true positive, false positive, true negative and so on.

CHAPTER 7

ALGORITHM AND PERFORMANCE

7.1. ALGORITHM:

7.1.1. NAIVE BAYES

Naive Bayes is a simple, yet effective and commonly-used, machine learning classifier. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

7.1.2. ALGORITHM STEPS:

Step 1: Select the email

Step 2: Extract features with help of tokenization and word count algorithm.

Step 3: Training the dataset with the help of Naive Bayesian Classifier.

Step4: Find the probability of spam and non-spam mails.

Prob_spam = (sum(train_matrix (spam_indices,)) + 1) ./ (spam_wc + numtokens)

Prob_nonspam = (sum(train_matrix(nonspam_indices,)) + 1) ./ (nonspam_wc + numtokens)

Step 5: Testing the dataset log_a =

test_matrix*(log(prob_tokens_spam))' + log(prob_spam) log_b

= test_matrix*(log(prob_tokens_nonspam))'+ log(1 -

prob_spam)

if

output = log_a>log_b then document are spam else

the document are non-spam

Step 6: Classify the spam and non-spam mails.

Step 7: compute the error of the text data and calculate the word which is wrongly classified

Numdocs_wrong = sum(xor(output, text_lables))

Step 8: display the error rate of text data and calculate the fraction of wrongly classified word

Fraction_wrong = numdocs_wrong/numtest_docs

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION:

The spam mails are detected using machine learning algorithm. Email has been the most important medium of communication nowadays; through internet connectivity any message can be delivered to all over the world. In the future this system can be implemented by using different algorithms and also more features can be added to the existing system.

FUTURE ENHANCEMENT:

- In the future, deep leaning and deep adversarial learning techniques can be used so that it can effectively handle the menace of spam emails.
- The current proposed system is for English language mails but as future scope we can design the system for multiple languages.

CHAPTER 9

REFERENCES

[1] Diale, M., Celik, T., & Walt, C. V. D. (2019). Unsupervised feature learning for spam email filtering. Computers & Electrical Engineering, 74, 89–104. 10.1016/j.compeleceng.2019.01.004.

[2] Feng, Y., Chen, H., Li, T., & Luo, C. (2020). A novel communitydetection method based on whale optimization algorithm with evolutionary population. Applied Intelligence, 1–20. 10.1007/s10489020-01659-7.

[3] Karim, A., Azam, S., Shanmugam, B., Kannoorpatti, K., &Alazab, M. (2019). A comprehensive survey for intelligent spam email detection. IEEE Access, 7, 168261–168295. 10.1109/ACCESS.2019.2954791.

[4] Luo, J., & Liu, Z. (2019). Novel grey wolf optimization based on modified differential evolution for numerical function optimization. Applied Intelligence, 1–19. 10.1007/s10489-019-01521-5.

[5] Pan, X., Xue, L., & Li, R. (2019). A new and efficient firefly algorithm for numerical optimization problems. Neural Computing and Applications, 1445–1453. 10.1007/s00521-018-3449-6.

[6] Sekh, A. A., Dogra, D. P., Kar, S., Roy, P. P., & Prasad, D. K. (2020). ELM-HTM guided bio-inspired unsupervised learning for anomalous trajectory classification. Cognitive Systems Research, 63, 30–41. 10.1016/j.cogsys.2020.04.003.

CHAPTER 10

APPENDIX

APPENDIX – Source code

```
#import packages
import seaborn as sn
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
#import data
spam_df = pd.read_csv("mail_data.csv")
mail_data = spam_df.where((pd.notnull(spam_df)),'')
mail_data.head()
mail_data.shape
mail_data.loc[mail_data['Category'] == 'spam', 'Category',] = 0
mail_data.loc[mail_data['Category'] == 'ham', 'Category',] = 1
X = mail_data['Message']
Y = mail_data['Category']
print(X)
print(Y)
#inspect data
spam_df.groupby('Category').describe()
#turn a spam\ham into numerical data, creating a new column called 'spam'
spam_df['spam']=spam_df['Category'].apply(lambda x: 1 if x == 'spam' else 0)
spam_df
#create train\test split
```

```
x_train, x_test, y_train, y_test = train_test_split(spam_df.Message, spam_df.spam)
print(x_train.shape)
print(x_test.shape)
x_train.describe()
cv=CountVectorizer()
x_train_count=cv.fit_transform(x_train.values)
x_train_count
x_train_count.toarray()
#train model
model=MultinomialNB()
model.fit(x_train_count, y_train)
#pre test ham
email_ham=["hey wanna meet up for the game?"]
email_ham_count=cv.transform(email_ham)
model.predict(email_ham_count)
#pre-test spam
email_spam=["reward money click"]
email_spam_count=cv.transform(email_spam)
model.predict(email_spam_count)
#test model
x_test_count=cv.transform(x_test)
model.score(x_test_count,y_test)
y_pred = model.predict(cv.transform(x_test))
con_mat = pd.DataFrame(confusion_matrix(y_test, y_pred), ["Spam","Ham"], ["Spam","Ham"])
sn.heatmap(con_mat, annot=True, fmt="")
```

10.2 OUTPUT

Training set size: 4179

Testing set size: 1393

Out[4]:

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

CONFUSION MATRIX: