***OPTIMIZING SPAM FILTERING***

1.INTRODUCTION

# 1.1 Overview

The business problem for optimizing spam filtering with machine learning is to develop an accurate and efficient system that can automatically identify and filter out spam emails from legitimate emails in order to improve the productivity and security of email communication. The primary goal is to reduce the number of unwanted emails that users receive, which can include malicious messages that contain phishing links, malware attachments, or other harmful content. By using machine learning algorithms to analyze email content, metadata, and user behavior patterns, businesses can build a more effective spam filtering system that can identify and block spam messages with a high degree of accuracy while minimizing false positives that may incorrectly flag legitimate emails as spam. This can improve the user experience, reduce the risk of cyber attacks, and increase the efficiency of email communication for businesses of all sizes.

# 1.2 Purpose

To optimize spam filtering in a business environment, there are several business requirements that need to be considered. These requirements are as follows:

Accuracy: The spam filter must be highly accurate and must be able to distinguish between spam and legitimate emails with a high degree of accuracy.

Speed: The spam filter must be able to process emails quickly, without significantly slowing down the email system.

Scalability: The spam filter must be able to handle large volumes of email traffic, and be scalable to meet the needs of growing businesses.

Customization: The spam filter must be customizable to meet the specific needs of the business, including the ability to block or allow certain types of emails.

Reporting and Analytics: The spam filter should provide reporting and analytics that allow the business to monitor its email traffic and identify any issues or trends.

User-Friendly Interface: The spam filter should be easy to use and manage, with a user-friendly interface that allows administrators to configure the system and manage any issues that arise.

Integration with Existing Systems: The spam filter should integrate seamlessly with existing email systems and other security solutions to ensure optimal performance.

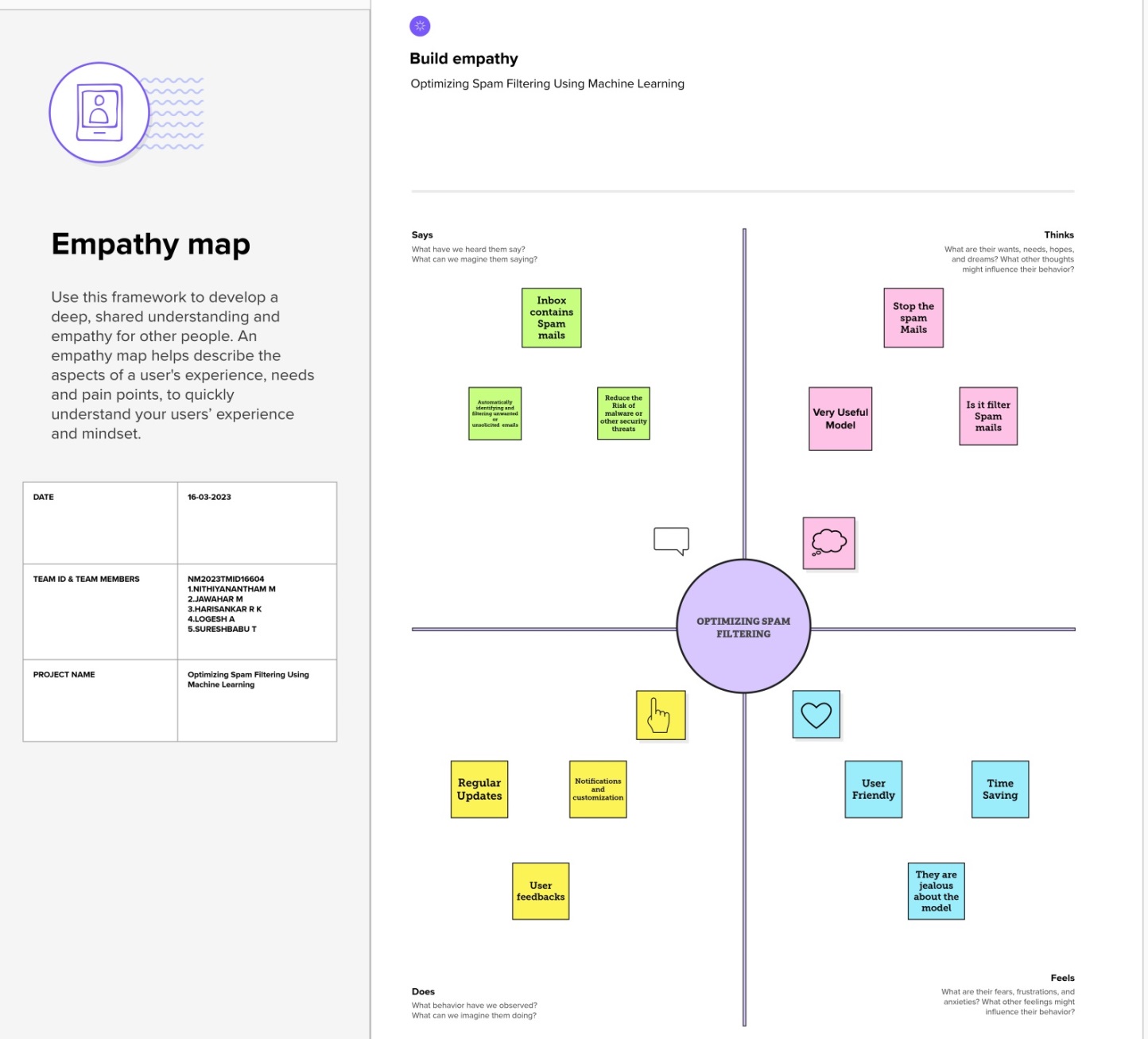
Compliance: The spam filter should comply with all relevant regulations and standards, including data protection and privacy laws.

Cost-Effective: The spam filter should be cost-effective and provide value for money, without compromising on quality or performance.

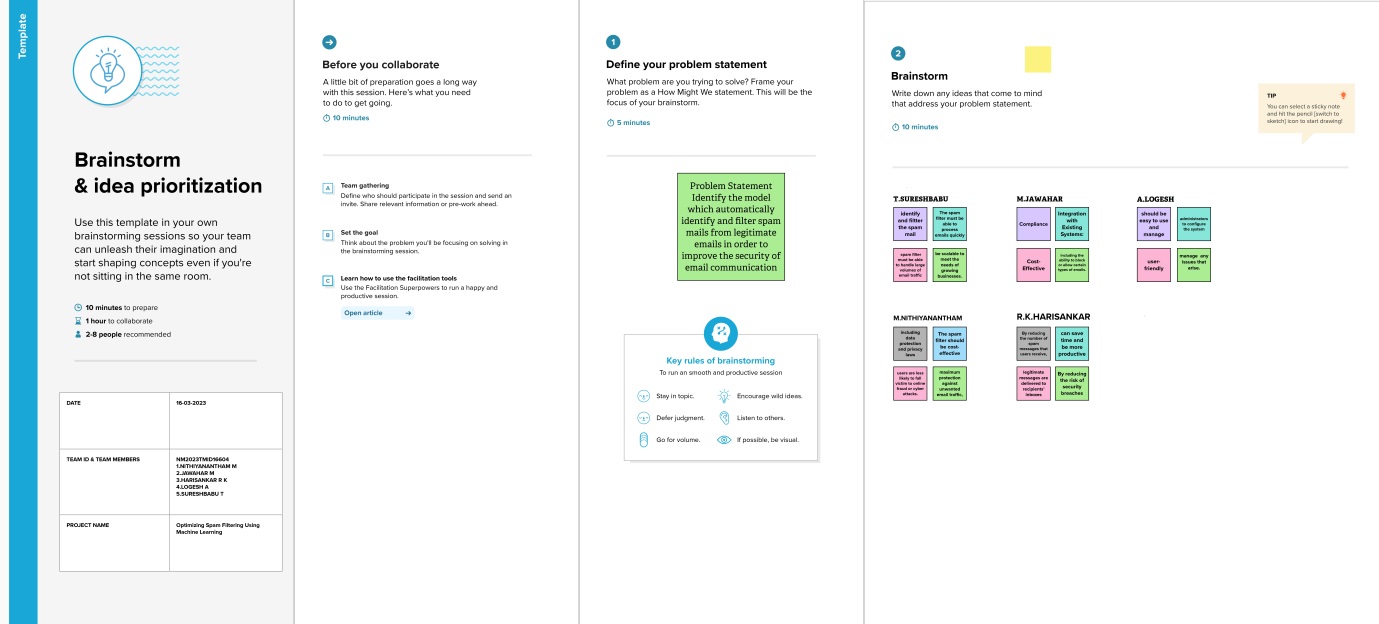
By meeting these requirements, businesses can ensure that their spam filter is optimized to provide maximum protection against unwanted email traffic, while minimizing the impact on legitimate communications

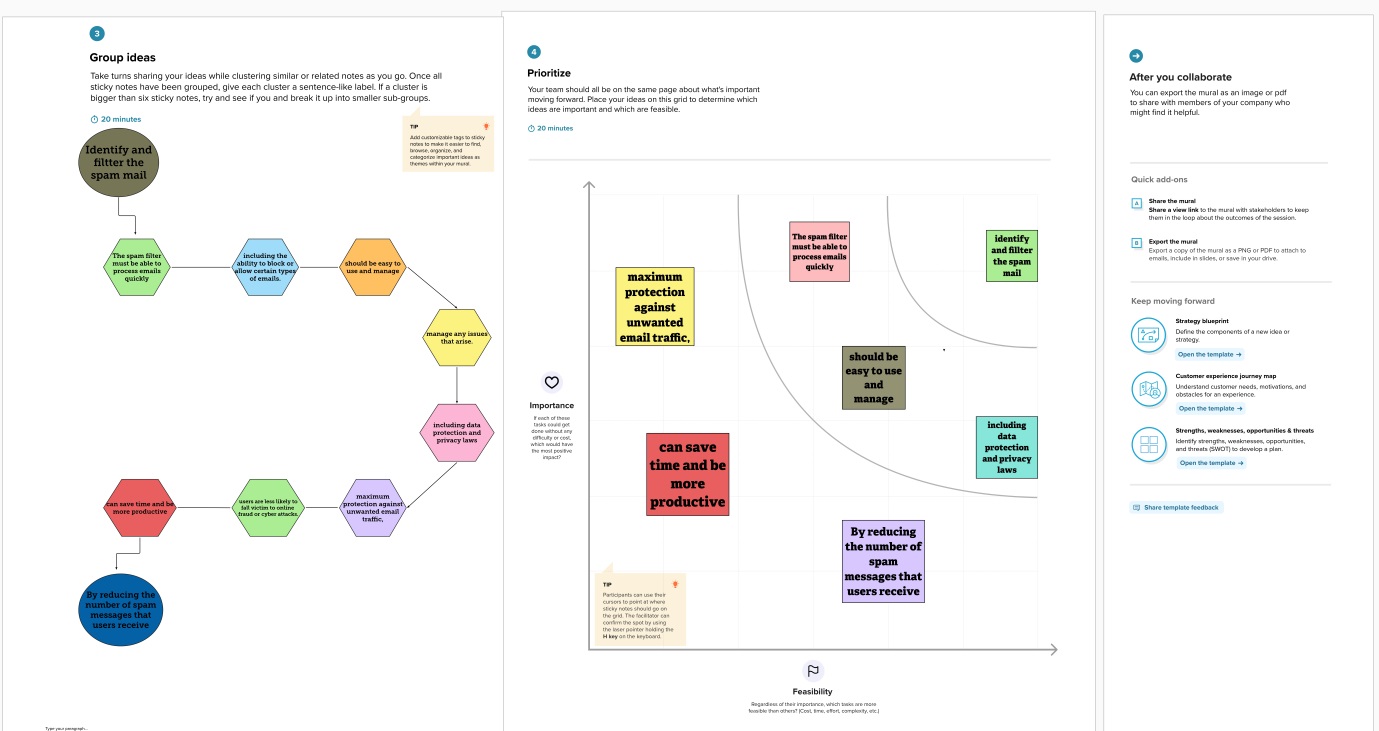
2.Problem Definition & Design Thinking:

# 2.1 Empathy map



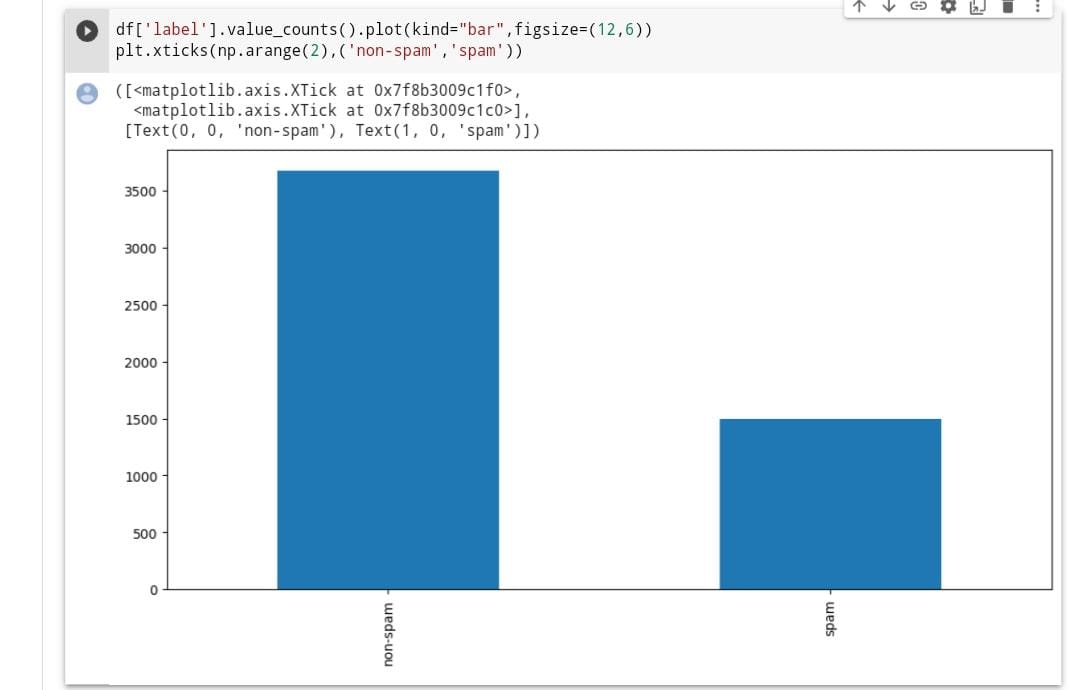
# 2.2 Idention and Brainstorming Map



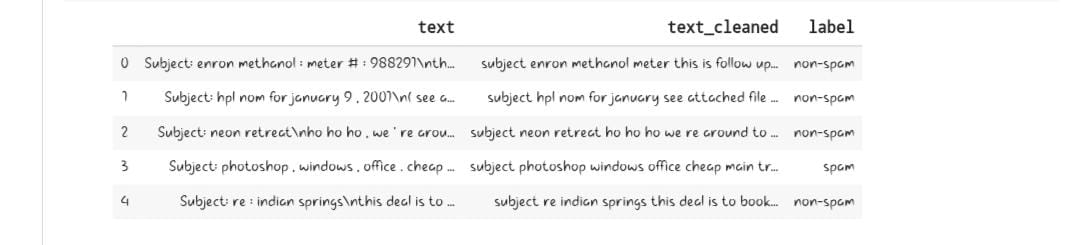


3.Results

# Data Analysis



# Text Cleaning

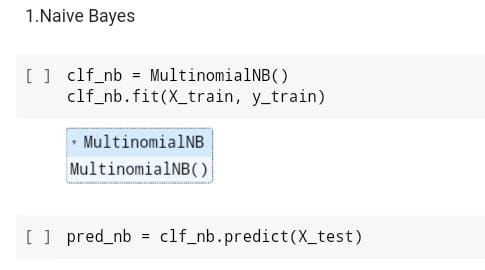


# Lemmatization

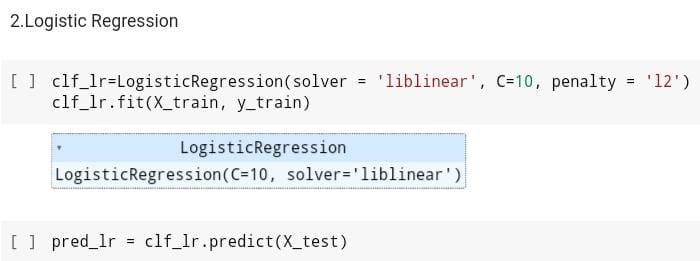


# Build Model

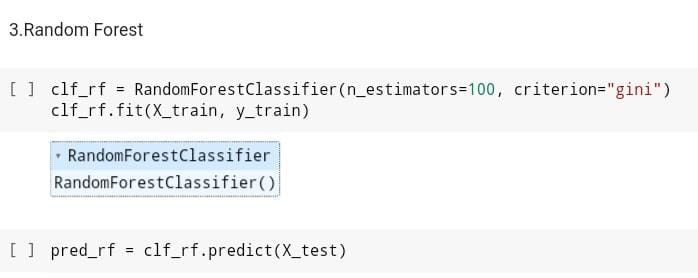
## 1.Naive Bayes



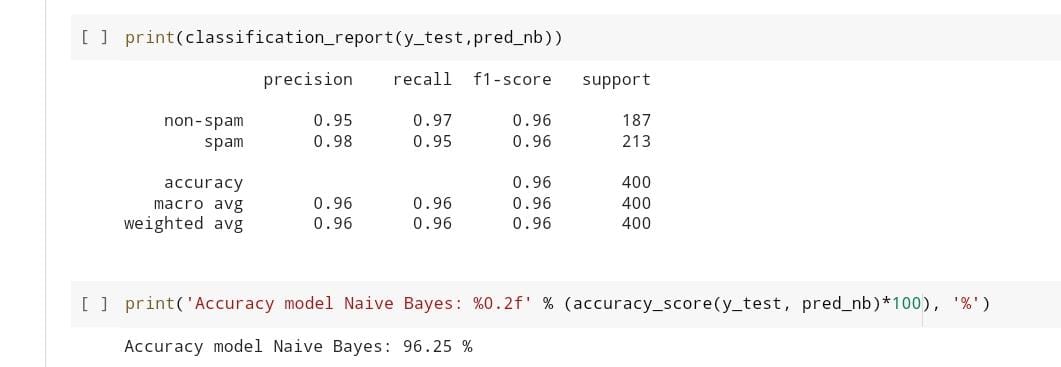
## 2.Logistic Regression

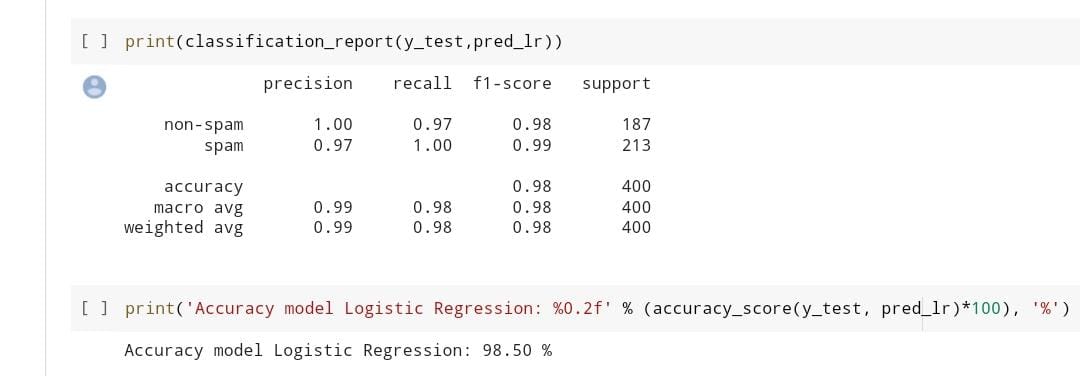


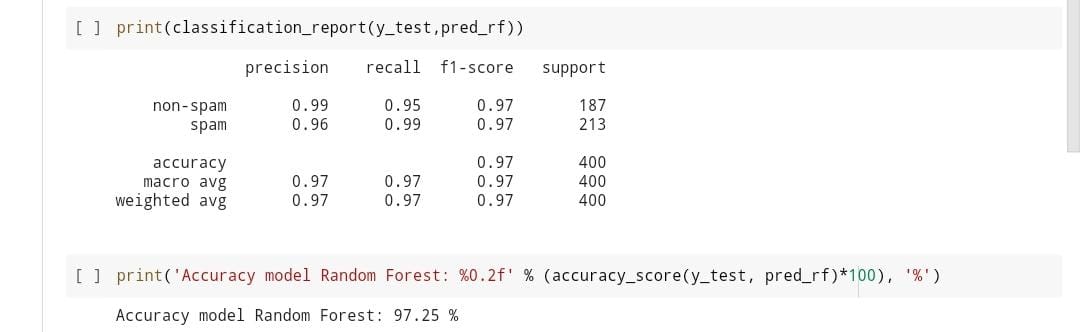
### 3.Random Forest



# Evaluation







4.Advantages and Disadvantages

Spam filtering is the process of identifying and removing unwanted and unsolicited email messages from an email inbox. Some of the advantages of spam filtering include:

1.Time savings: Spam filtering can save a significant amount of time by reducing the number of unwanted messages that need to be manually deleted or filtered.

2. Improved productivity: By reducing the amount of spam that employees receive, spam filtering can improve productivity and reduce the likelihood of distractions and interruptions.

3. Enhanced security: Spam filters can help protect against phishing attacks and other email-based threats by identifying and blocking suspicious messages.

4. Reduced risk of malware infections: Many spam emails contain malware such as viruses, trojans, and other malicious software. By filtering out these emails, spam filtering can reduce the risk of malware infections.

5. Improved email performance: By reducing the volume of spam emails, spam filtering can help improve the performance of email servers and reduce the likelihood of email downtime.

While spam filtering has many advantages, there are also some potential disadvantages to consider:

1. False positives: Spam filters may sometimes incorrectly identify legitimate emails as spam, resulting in important messages being blocked or sent to the spam folder. This can be particularly problematic for businesses that rely heavily on email communication.
2. False negatives: On the other hand, some spam messages may slip through the filters and end up in the inbox. This can be particularly problematic if the spam message contains malware or other malicious content.
3. Complexity: Spam filtering can be complex to set up and configure, particularly for organizations with multiple email accounts or complex email systems. This can result in additional costs and management overhead.
4. Increased server load: Spam filtering can increase the load on email servers, particularly if the volume of spam messages is high. This can result in slower email performance and increased costs for additional hardware or infrastructure.
5. Cost: While some basic spam filtering solutions are available for free, more advanced solutions can be costly, particularly for businesses with large email volumes.

5.Application

Spam filtering has a wide range of applications, including:

1. Email: Spam filtering is most commonly used to filter unwanted email messages. This can include both individual users filtering their personal email inboxes, as well as businesses and organizations filtering email for their entire network.
2. Social media: Spam filtering is also used on social media platforms to filter out unwanted messages and comments.
3. Online forums: Online forums and discussion boards often use spam filters to prevent spam and unwanted posts.
4. E-commerce: E-commerce websites can use spam filters to prevent fraudulent transactions and prevent spam from being posted in customer reviews or feedback sections.
5. Search engines: Search engines use spam filters to prevent websites from using spam techniques to manipulate search rankings.
6. Messaging apps: Messaging apps such as WhatsApp and Facebook Messenger use spam filters to prevent unwanted messages from being sent to users.

6.Conclusion

Spam filtering is a critical tool in the fight against unwanted and unsolicited messages, particularly in the context of email. By filtering out spam and other unwanted messages, spam filtering can improve productivity, enhance security, and reduce the risk of malware infections. However, spam filtering is not without its potential downsides, including false positives, false negatives, increased complexity and costs, and increased server load. Ultimately, the benefits of spam filtering must be weighed against the potential drawbacks, and organizations must carefully evaluate their options before implementing a spam filtering solution. Despite its potential challenges, spam filtering remains an important tool for organizations and individuals looking to manage their email and online communication effectively.

7.Future Scope

The future scope of spam filtering is likely to involve a continued focus on improving the accuracy and efficiency of spam filters, as well as addressing emerging threats and new forms of spam. Some potential areas of focus for the future of spam filtering include:

1. Machine learning and artificial intelligence: The use of machine learning and artificial intelligence is likely to continue to grow in the area of spam filtering, allowing for more accurate and efficient detection and filtering of spam messages.
2. Collaborative filtering: Collaborative filtering techniques, which involve using data from multiple sources to improve filtering accuracy, may become more common in the future.
3. Mobile spam filtering: With the increasing use of mobile devices for email and other forms of communication, there is likely to be a greater focus on developing spam filtering solutions for mobile devices.
4. Social media spam filtering: As social media platforms continue to grow in popularity, there will be an increasing need for spam filtering solutions for social media messages and comments.
5. Integration with other security solutions: Spam filtering is likely to become more closely integrated with other security solutions, such as antivirus software, firewalls, and intrusion detection systems.

8.Appendix

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import re

import nltk

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report, f1\_score, accuracy\_score, confusion\_matrix

from sklearn.naive\_bayes import MultinomialNB

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from nltk.corpus import stopwords

from nltk.stem import WordNetLemmatizer

from nltk.tokenize import word\_tokenize

from nltk.corpus import wordnet

nltk.download('stopwords')

nltk.download('punkt')

nltk.download('averaged\_perceptron\_tagger')

nltk.download('wordnet')

import warnings

warnings.filterwarnings("ignore")

[nltk\_data] Downloading package stopwords to /root/nltk\_data...

[nltk\_data] Unzipping corpora/stopwords.zip.

[nltk\_data] Downloading package punkt to /root/nltk\_data...

[nltk\_data] Unzipping tokenizers/punkt.zip.

[nltk\_data] Downloading package averaged\_perceptron\_tagger to

[nltk\_data] /root/nltk\_data...

[nltk\_data] Unzipping taggers/averaged\_perceptron\_tagger.zip.

[nltk\_data] Downloading package wordnet to /root/nltk\_data...

**Unnamed: 0 label text label\_num**

**0** 605 ham Subject: enron methanol ; meter # : 988291\nth... 0

**1** 2349 ham Subject: hpl nom for january 9 , 2001\n( see a... 0

**2** 3624 ham Subject: neon retreat\nho ho ho , we ' re arou... 0

**3** 4685 spam Subject: photoshop , windows , office . cheap ... 1

**4** 2030 ham Subject: re : indian springs\nthis deal is to ... 0

df = pd.read\_csv("spam\_ham\_dataset.csv")

df.head()

**label text label\_num**

**0** ham Subject: enron methanol ; meter # : 988291\nth... 0

**1** ham Subject: hpl nom for january 9 , 2001\n( see a... 0

**2** ham Subject: neon retreat\nho ho ho , we ' re arou... 0

**3** spam Subject: photoshop , windows , office . cheap ... 1

**4** ham Subject: re : indian springs\nthis deal is to ... 0

df = df.drop(['Unnamed: 0'], axis=1)

df.head()

Exploratory Data Analysis

print('Count %s data email'% len(df))

Count 5171 data email

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 2/7

df[

'label'

].value\_counts()

ham 3672spam 1499Name: label, dtype: int64

df[

'label'

] = df[

'label'

].replace([

'ham'

],

'non-spam'

)

([<matplotlib.axis.XTick at 0x7f8b3009c1f0>,

<matplotlib.axis.XTick at 0x7f8b3009c1c0>],

[Text(0, 0, 'non-spam'), Text(1, 0, 'spam')])

df[

'label'

].value\_counts().plot(kind=

"bar"

,figsize=(

12

,

6

))

plt.xticks(np.arange(

2

),(

'non-spam'

,

'spam'

))

print

(df[

'text'

][

0

])

Subject: enron methanol ; meter # : 988291this is a follow up to the note i gave you on monday , 4 / 3 / 00 { preliminaryflow data provided by daren } .please override pop ' s daily volume { presently zero } to reflect dailyactivity you can obtain from gas control .this change is needed asap for economics purposes .

Preprocessing

df = df.groupby(

'label'

).head(

1000

)

df[

'label'

].value\_counts()

non-spam 1000spam 1000Name: label, dtype: int64

Text Cleaning

import

string

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 3/7

punct = []

for

char

in

string.punctuation:

punct.append(char)

def

cleaning

(

txt

):

# case folding

text = txt.lower()

# remove multiple space, tabs, dan newlines

text = re.sub(

'\s+'

,

' '

,text)

# remove links

text = text.replace(

"http://"

,

" "

).replace(

"https://"

,

" "

)

# remove special characters

text = text.encode(

'ascii'

,

'replace'

).decode(

'ascii'

)

text =

' '

.join(re.sub(

"([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"

,

" "

, text).split())

# remove punctuation

text =

''

.join([word

for

word

in

text

if

word

not

in

punct])

#remove single character

text = re.sub(r

"\b[a-zA-Z]\b"

,

""

, text)

#remove numbers

text = re.sub(r

"\d+"

,

""

, text)

#remove multiple spaces (again)

text = re.sub(

'\s+'

,

' '

,text)

return

text

**text text\_cleaned label**

**0** Subject: enron methanol ; meter # :

988291\nth...

subject enron methanol meter this is follow

up...

non-spam

**1** Subject: hpl nom for january 9 , 2001\n(

see a...

subject hpl nom for january see attachedfile ...non-spam**2**Subject: neon retreat\nho ho ho , we ' rearou...

subject neon retreat ho ho ho we re

around to ...

non-spam

**3** Subject: photoshop , windows , office . subject photoshop windows office cheap

df[

'text\_cleaned'

] = df[

'text'

].apply(

lambda

x: cleaning(x))

df = df[[

'text'

,

'text\_cleaned'

,

'label'

]]

df.head()

Stopword Removal

top = stopwords.words('english')

f['text\_cleaned'] = df['text\_cleaned'].apply(lambda x: ' '.join([word for word in x.split() if word not in stop]))

Lemmatization

lemmatizer = WordNetLemmatizer()

# mapping the POS tags

def

get\_wordnet\_pos

(

word

):

"""Map POS tag to first character lemmatize() acce

pts"""

tag = nltk.pos\_tag([word])[

0

][

1

][

0

].upper()

tag\_dict = {

"J"

: wordnet.ADJ,

"N"

: wordnet.NOUN,

"V"

: wordnet.VERB,

"R"

: wordnet.ADV}

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 4/7

return tag\_dict.get(tag, wordnet.NOUN)

def do\_lemma(string):

lemmatized = ' '.join([lemmatizer.lemmatize(word, get\_wordnet\_pos(word)) for word in nltk.word\_tokenize(string)]

return lemmatized

sentence = "The striped bats are hanging on their feet for best"

lemmatized = ' '.join([lemmatizer.lemmatize(word, get\_wordnet\_pos(word)) for word in nltk.word\_tokenize(sentence)])

print(do\_lemma(sentence))

The strip bat be hang on their foot for best

df['text\_cleaned'] = df['text\_cleaned'].apply(lambda x: do\_lemma(x))

**text text\_cleaned label**

**0** Subject: enron methanol ; meter # : 988291\nth... subject enron methanol meter follow note give ... non-spam

**1** Subject: hpl nom for january 9 , 2001\n( see a... subject hpl nom january see attach file hplnol... non-spam

**2** Subject: neon retreat\nho ho ho , we ' re arou... subject neon retreat ho ho ho around wonderful... non-spam

**3** Subject: photoshop , windows , office . cheap ... subject photoshop window office cheap main tre... spam

**4** Subject: re : indian springs\nthis deal is to ... subject indian spring deal book teco pvr reven... non-spam

df.head()

Data Preparation

df = df.drop(['text'], axis=1)

df = df.rename(columns = {'text\_cleaned' : 'text'})

df.columns

Index(['text', 'label'], dtype='object')

Feature Extraction using TF-IDF

tfidf = TfidfVectorizer()

X = tfidf.fit\_transform(df['text'])

y = df['label']

train test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Build Model

1.Naive Bayes

▾ MultinomialNB

MultinomialNB()

clf\_nb = MultinomialNB()

clf\_nb.fit(X\_train, y\_train)

pred\_nb = clf\_nb.predict(X\_test)

2.Logistic Regression

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 5/7

▾ LogisticRegression

LogisticRegression(C=10, solver='liblinear')

clf\_lr=LogisticRegression(solver = 'liblinear', C=10, penalty = 'l2')

clf\_lr.fit(X\_train, y\_train)

pred\_lr = clf\_lr.predict(X\_test)

3.Random Forest

▾ RandomForestClassifier

RandomForestClassifier()

clf\_rf = RandomForestClassifier(n\_estimators=100, criterion="gini")

clf\_rf.fit(X\_train, y\_train)

pred\_rf = clf\_rf.predict(X\_test)

Evaluation

print(classification\_report(y\_test,pred\_nb))

precision recall f1-score support

non-spam 0.95 0.97 0.96 187

spam 0.98 0.95 0.96 213

accuracy 0.96 400

macro avg 0.96 0.96 0.96 400

weighted avg 0.96 0.96 0.96 400

print('Accuracy model Naive Bayes: %0.2f' % (accuracy\_score(y\_test, pred\_nb)\*100), '%')

Accuracy model Naive Bayes: 96.25 %

print(classification\_report(y\_test,pred\_lr))

precision recall f1-score support

non-spam 1.00 0.97 0.98 187

spam 0.97 1.00 0.99 213

accuracy 0.98 400

macro avg 0.99 0.98 0.98 400

weighted avg 0.99 0.98 0.98 400

print('Accuracy model Logistic Regression: %0.2f' % (accuracy\_score(y\_test, pred\_lr)\*100), '%')

Accuracy model Logistic Regression: 98.50 %

print(classification\_report(y\_test,pred\_rf))

precision recall f1-score support

non-spam 0.99 0.95 0.97 187

spam 0.96 0.99 0.97 213

accuracy 0.97 400

macro avg 0.97 0.97 0.97 400

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 6/7

Colab paid products

-

Cancel contracts here

weighted avg 0.97 0.97 0.97 400

print

(

'Accuracy model Random Forest: %0.2f'

% (accuracy\_score(y\_test, pred\_rf)\*

100

),

'%'

)

Accuracy model Random Forest: 97.25 %

pip install flask

Looking in indexes:

https://pypi.org/simple

,

https://us-python.pkg.dev/colab-wheels/public/simple/

Requirement already satisfied: flask in /usr/local/lib/python3.9/dist-packages (2.2.3)Requirement already satisfied: Werkzeug>=2.2.2 in /usr/local/lib/python3.9/dist-packages (from flask) (2.2.3)Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.9/dist-packages (from flask) (8.1.3)Requirement already satisfied: importlib-metadata>=3.6.0 in /usr/local/lib/python3.9/dist-packages (from flaskRequirement already satisfied: itsdangerous>=2.0 in /usr/local/lib/python3.9/dist-packages (from flask) (2.1.2Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.9/dist-packages (from flask) (3.1.2)Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.9/dist-packages (from importlib-metadata>=3Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-packages (from Jinja2>=3.0->fl

---------------------------------------------------------------------------

ImportError

Traceback (most recent call last)<ipython-input-3-d954d1bdd588>

in

<cell line: 1>

()

----> 1 from

flask

import

flask,render\_template,request

2

import pickle

3 import re

4

import nltk

5 from nltk.

corpus

import

stopwords

ImportError

: cannot import name 'flask' from 'flask' (/usr/local/lib/python3.9/dist-packages/flask/\_\_init\_\_.py)

---------------------------------------------------------------------------

NOTE: If your import is failing due to a missing package, you canmanually install dependencies using either !pip or !apt.To view examples of installing some common dependencies, click the"Open Examples" button below.

---------------------------------------------------------------------------

OPEN EXAMPLES

SEARCH STACK OVERFLOW

from

flask

import

flask,render\_template,request

import

pickle

import

re

import

nltk

from

nltk.corpus

import

stopwords

from

nltk.stem

import

porterstemmer

from

tensorflow.keras.models

import

load\_model

4/13/23, 5:12 PM Untitled17.ipynb - Colaboratory

https://colab.research.google.com/drive/1TH0xKUFTdVl9FU6GP8IRzFRubhnwpC7G?usp=sharing#scrollTo=-XpuVIg-o3TN&printMode=true 7/7