RAJESHWARI VEDHACHALAM GOVERNMENT ARTS COLLEGE-CHENGALPATTU

OPTIMIZING SPAM FILTERING USING MACHINE LEARNING FOR DETECT JUNK MAIL

TEAM MEMBERS NAME: K. Indhumathi

T. Sumithira

B. Kalaivani

D. Sasikala

CLASS : III-BSC COMPUTER SCIENCE

TABLE OF INDEX

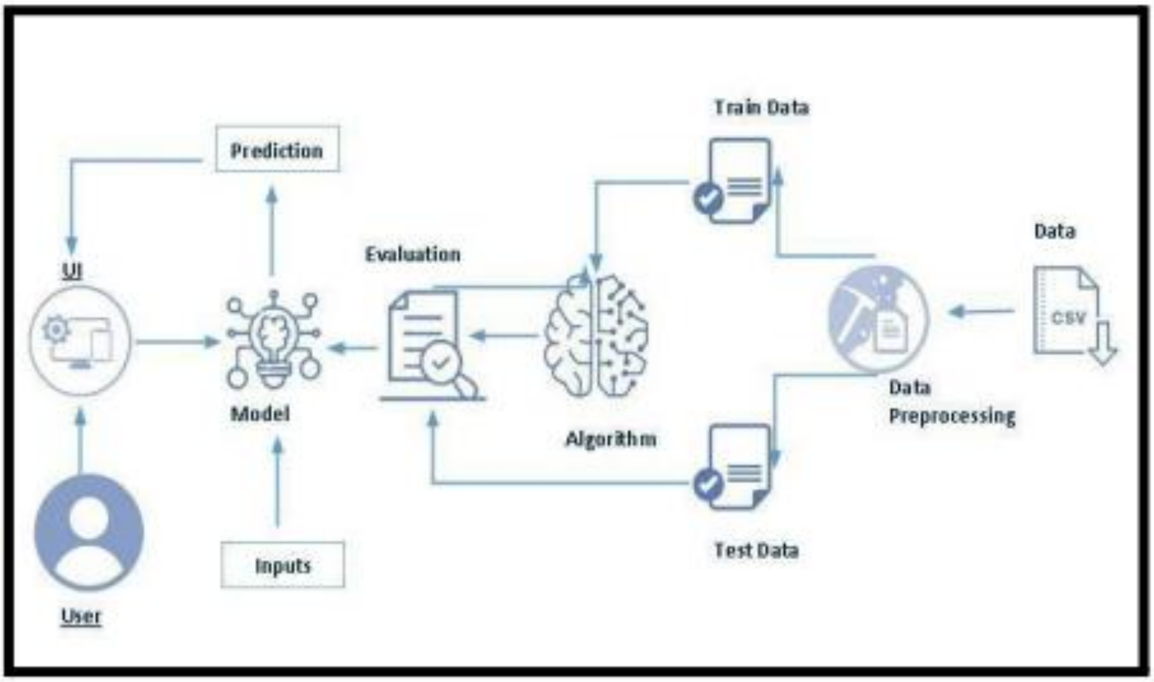
1. INTRODUCTION

PROJECT DESCRIPTION

Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS, Mobile service providers suffer from some sort of financial problems as well as it reduces calling time for users. Unfortunately, if the user accesses such Spam SMS they may face the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.

To avoid such Spam SMS people use white and black list of numbers. But this technique is not adequate to completely avoid Spam SMS. To tackle this problem it is needful to use a smarter technique which correctly identifies Spam SMS. Natural language processing technique is useful for Spam SMS identification. It analyses text content and finds patterns which are used to identify Spam and Non-Spam SMS.

Technical Architecture:



1.1.OVERVIEW

- User interacts with the UI to enter the input.
- Entered input is analysed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

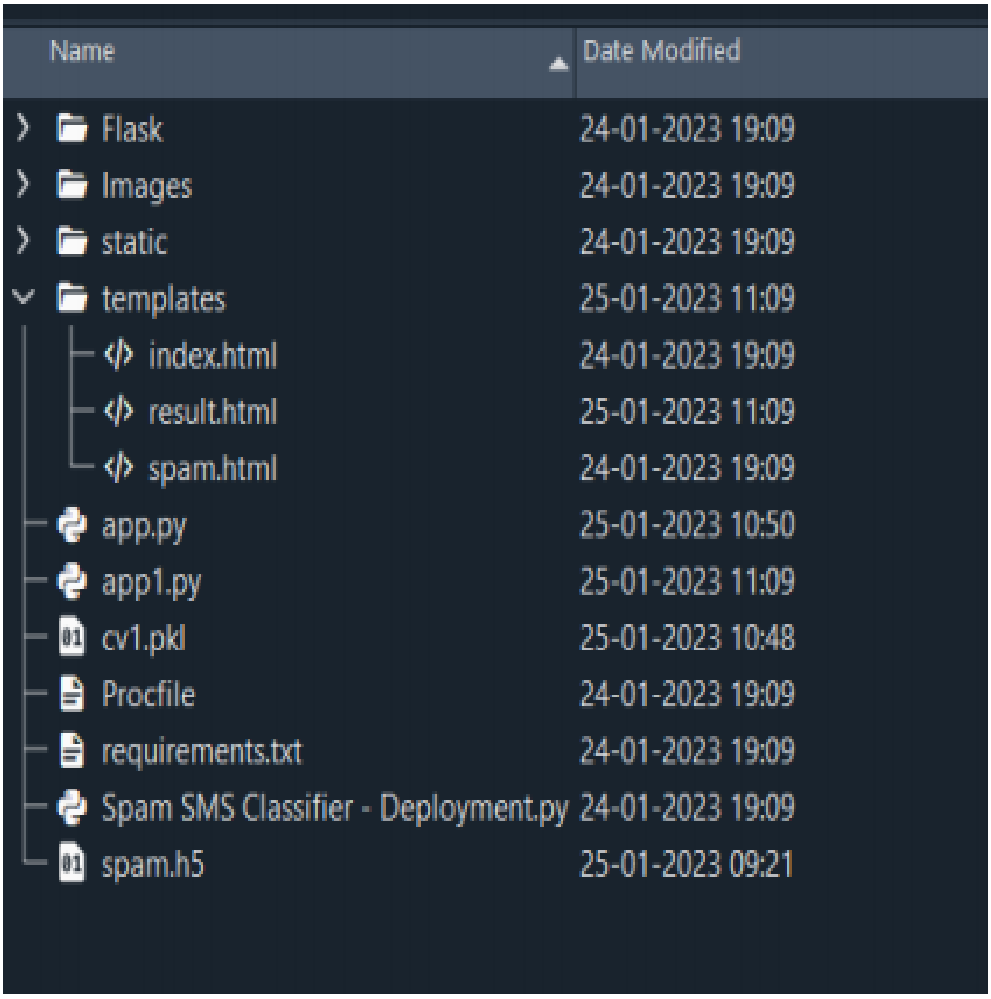
- Define Problem / Problem Understanding
  - **Specify the business problem**
  - **Business requirements**
  - **Literature Survey**
  - **Social or Business Impact.**
- Data Collection & Preparation
  - **Collect the dataset**
  - **Data Preparation**
- Exploratory Data Analysis
  - **Descriptive statistical**
  - **Visual Analysis**
- Model Building
  - **Training the model in multiple algorithms**
  - **Testing the model**
- Performance Testin& g Hyperparameter Tuning
  - **Testing model with multiple evaluation metrics**
  - **Comparing model accuracy before & after applying hyperparameter tuning**
- Model Deployment
  - **Save the best model**
  - **Integrate with Web Framework**
- Project Demonstration & Documentation

○ **Record explanation Video for project end to solution**

○ **Project Documentation-Step by step project development Procedure**

Project Structure:

Create the Project folder which contains files as shown below



- We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.

- Spam.h5 is our saved model. Further we will use this model for flask integration.

1.2. PURPOSE

Optimizing spam filtering with machine learning can be a highly effective way to improve the accuracy of spam detection and reduce the amount of unwanted email that users receive. There are several ways in which machine learning can be used to improve spam filtering, including .Supervised learning: This involves training a machine learning model on a dataset of labeled spam and non-spam messages.

The model can then use this training to classify new messages as spam or non-spam based on their features, such as keywords, sender address, and message content. Unsupervised learning: This involves training a machine learning model on a dataset of unlabeled messages and using clustering algorithms to group similar messages together. The model can then identify clusters of messages that are likely to be spam based on their similarity to known spam messages**. Natural Language Processing (NLP)**

2. Define Problem / Problem Thinking

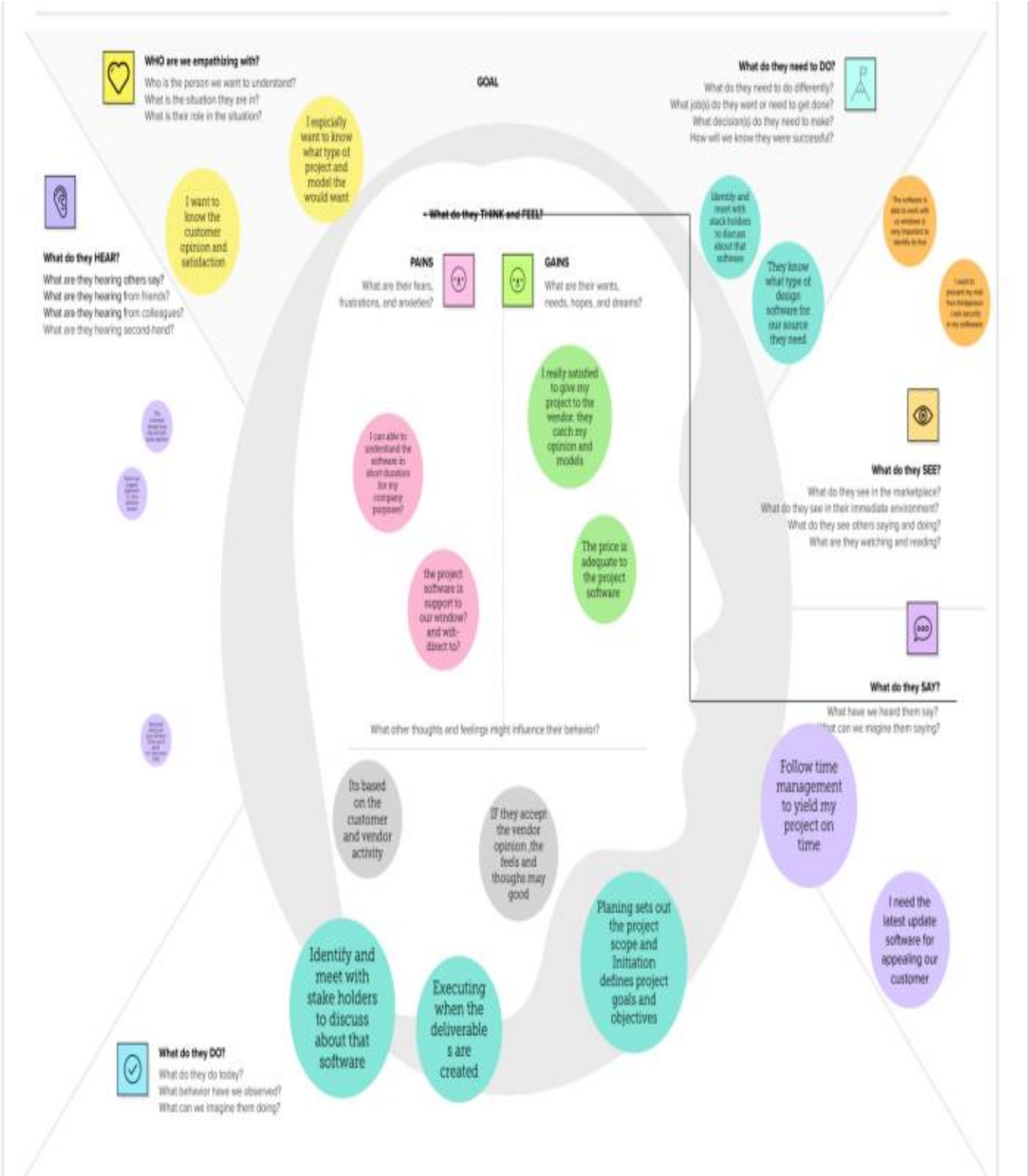
Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS, Mobile service providers suffer from some sort of financial problems as well as it reduces calling time for users. Unfortunately, if the user accesses such Spam SMS they may face the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.

To avoid such Spam SMS people use white and black list of numbers. But this technique is not adequate to completely avoid Spam SMS. To tackle this problem it is needful to use a smarter technique which correctly identifies Spam SMS. Natural language processing technique is useful for Spam SMS identification. It analyses text content and finds patterns which are used to identify Spam and Non-Spam SMS.

This is the initial step in building a machine learning model which aims to understand the need for it in the organisation. The machine learning development process can be resource intensive, so clear objectives should be agreed and set at the start. Clearly define the problem that a model needs to solve and what success looks like. A deployed model will bring much more value if it’s fully aligned with the objectives of the organisation. Before the project begins, there are key elements that need to be explored and planned.

2.1. EMPATHY MAP

In the ideation phase we have empathized as our client optimizing spam filtering with machine learning and we have acquired the details which are represented in the Empathy Map given



2.2: IDEATION & BRAINSTORMING MAP

Under this activity our team members have gathered and discussed various idea to solve out project problem. Each meber contributed 6 to 10 ideas after gathering all ideas we have assessed the impact and feasibility of each points. Finally, we have assigns the priority for each points based on the impact values.

STEP1 : TEAN GATHERING,COLLABORATION AND SELECT THE PROBLEM

1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

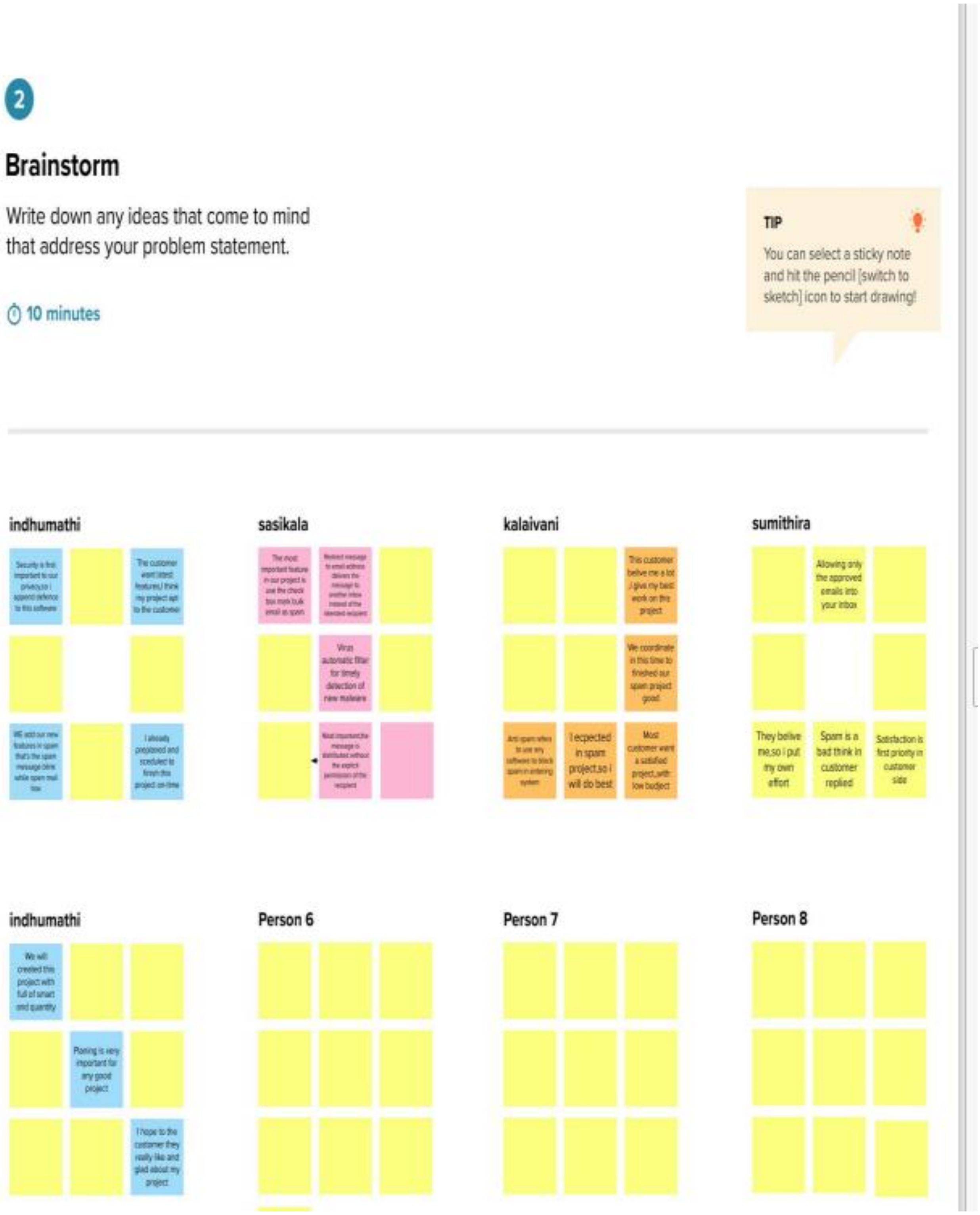
How might we[optimize spam filtering]?

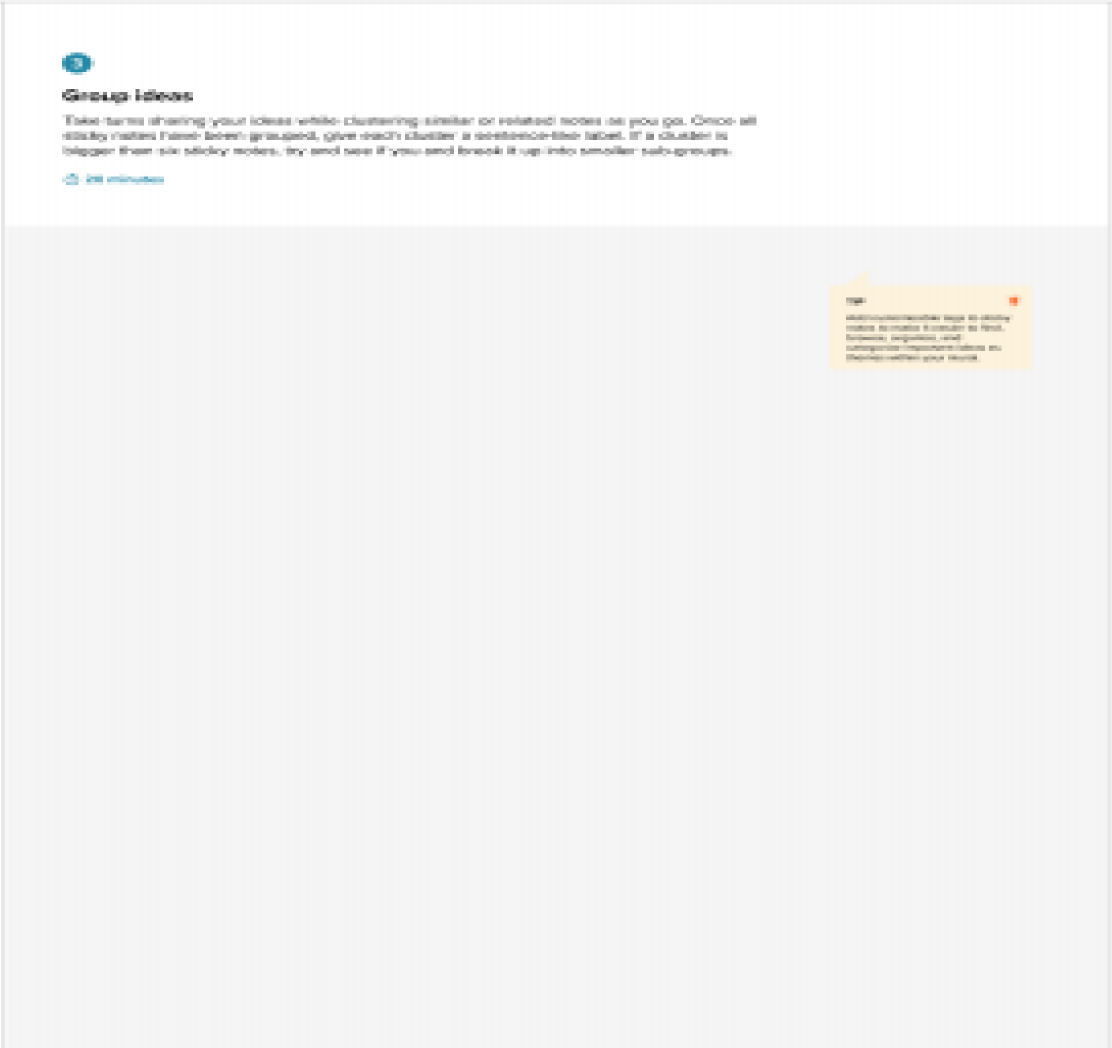
Key rules of brainstorming

To run an smooth and productive session

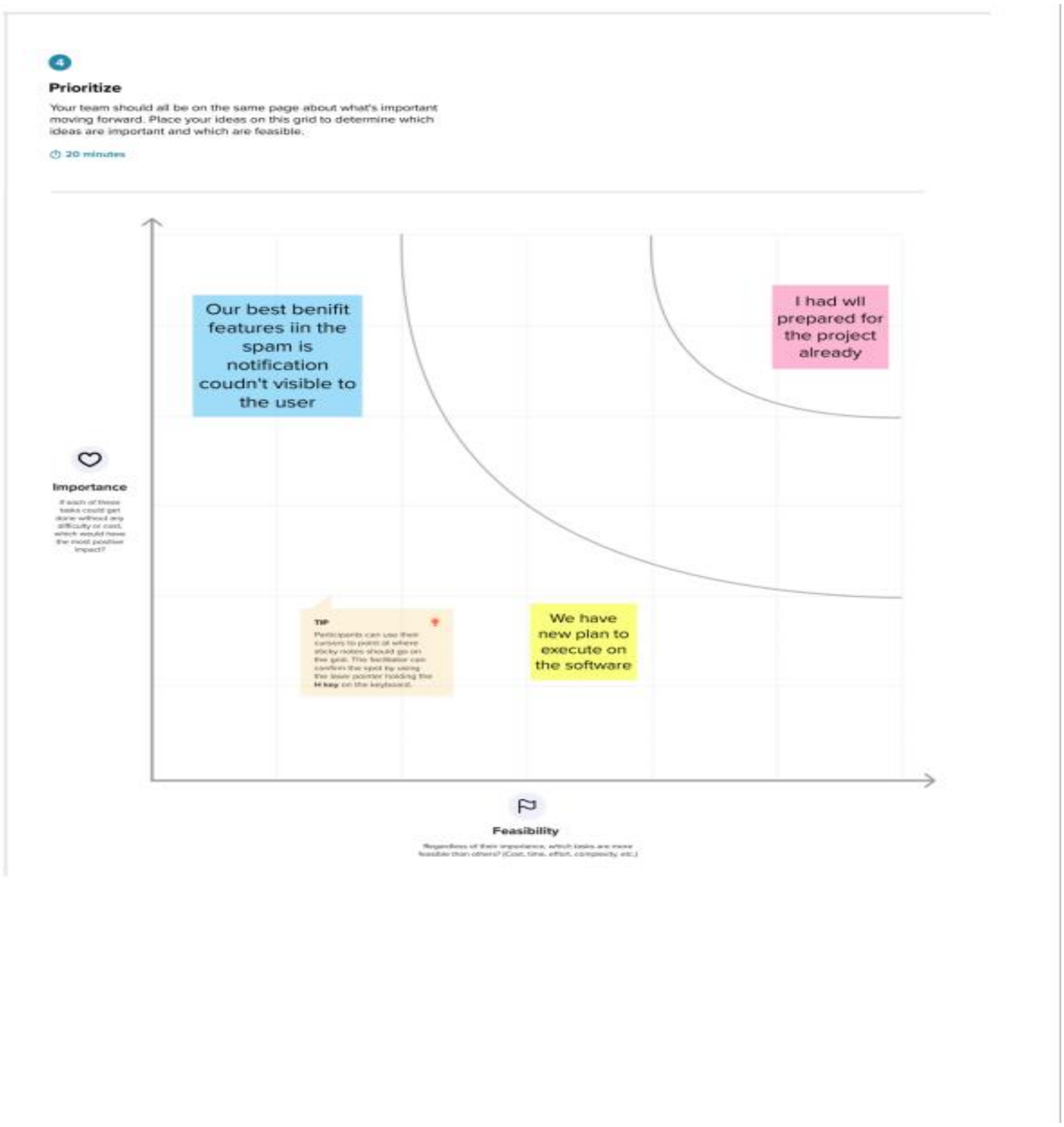
- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

STEP-2: BRAINSTORM ,IDEA LISTING AND GROUPING




3
Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.
Tip

STEP-3: IDEA PRIORITIZATION



After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

3. RESULT

Read the Dataset

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0

```
<bound method DataFrame.info of       Unnamed: 0 label                                               text  \
0            605   ham  Subject: enron methanol ; meter # : 988291\r\n...
1           2349   ham  Subject: hpl nom for january 9 , 2001\r\n( see...
2           3624   ham  Subject: neon retreat\r\nho ho ho , we ' re ar...
3           4685  spam  Subject: photoshop , windows , office . cheap ...
4           2030   ham  Subject: re : indian springs\r\nthis deal is t...
...          ...   ...                                                ...
5166        1518   ham  Subject: put the 10 on the ft\r\nthe transport...
5167         404   ham  Subject: 3 / 4 / 2000 and following noms\r\nhp...
5168        2933   ham  Subject: calpine daily gas nomination\r\n>\r\n...
5169        1409   ham  Subject: industrial worksheets for august 2000...
5170        4807  spam  Subject: important online banking alert\r\ndea...

      label_num
0             0
1             0
2             0
3             1
4             0
...         ...
5166          0
5167          0
5168          0
5169          0
5170          1
```

```
<bound method NDFrame._add_numeric_operations.<locals>.sum of       Unnamed: 0  label   text  label_num
0          False  False  False      False
1          False  False  False      False
2          False  False  False      False
3          False  False  False      False
4          False  False  False      False
...          ...    ...    ...        ...
5166       False  False  False      False
5167       False  False  False      False
5168       False  False  False      False
5169       False  False  False      False
5170       False  False  False      False

[5171 rows x 4 columns]>
```

Rename

	Unnamed: 0	SP_LABEL	SP_TEXT_GI	label_num
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\ndea...	1

Handling Categorical Values

Head

	unnamed:0	SP_LABEL	SP_TEXT	label_num
0	605	0	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	0	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	0	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	1	Subject: photoshop , windows , office . cheap ...	1
4	2030	0	Subject: re : indian springs\r\nthis deal is t...	0

Shape:

```
(5171, 4)
```

Handling Imbalance Data

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
Befor over sampling, counts of label '1':1075
Befor over sampling, counts of label '0' :2544
```

```
After oversampling, the shape of train_x:(5088, 50447)
After oversampling, the shape of train_y:(5088,)
```

Cleaning the text data

Stopwords

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True

Curpos

```
['988291 4 3 00',
 '9 2001 09 09',
 '25 1 5 6 12 13 13 5 15',
 '',
 '',
 '',
 '30 30 30 4 22 25 30 21 21 21 21 3 22 01 3 25 01',
 '3 100',
 '2 26 02 27 2001 10 33 02 27 2001 09 46 26 2 26 2 26 01 0 456 469 1 027',
 '21 23 2000 021 021',
 '12 5 0 025 5 0 12 15 0 15 2 2 21 0 025 5 0 12 15 0 15 8 27 1933 21 1934 17 49000 1995',
 '01405 1 4 45 60 2004',
 '28 2000 20 000 120 000 20 000',
 '',
 '384258 3 902 687 50',
 '30 0 45 1 80 1 8 20 8 1 80 2 14 5 2006 14 52 20 6 3 81 24 2 6 53 1 99 3 85 2 6 1 29 7 64 2 6 379 2 13 2 6 143 886 2 6 27 1933 21 1934 1934 042',
 '643714 02 23 2001 04 55 9879 800 85 2 17 2 28 3 6353 02 23 2001 04 21 02 22 2001 10 39',
 '2001 9643 04 26 2001 07 09 04 25 2001 06 13 2001 6 400',
 '4 09 19 2000 08 44 09 18 2000 04 53 4 4 9851 11 204 2000 9852 568 2000',
 '',
 '22 2001 8 38 05 22 2001 08 35 05 21 2001 03 26 2000 2001 16 18 2000 14 18 2001 17 2001 18 2000 18 2000 17 2001 5 17 2001 44 923 2 78 5 18 2000 43 709 5 17
2001 5 18 2000 3 3 5 4',
 '6461 35 300 300 45 350 6 33 300 42 300 1 2001 31 2001',
 '25 2001 26 26',
 '10 00 2000 0 04',
 '25 26 1 2 3 1 4 4 12 29 5 3 38 3 5 1950 40 50 3 38 3 5 7 3 38 3 5 1958 3 1960 2000 3 160 3 51 50 75 5 3 38 3 5 100 3 38 3 5 480 640 160 640 70 46 24 29
```

```
 '11 2000 11 2000 40 000 25 209 84 791 775 000 11 2000 3949 10 13 2000 03 04 12 2000 3949 40 000 110 000 10 13 2000 02 58 10 13 2000 10 04 12 2000 40 000 110 000',
 '150 20 25',
 '',
 '27 2001 16 07 26 0500 713 853 1411 0044 207 78 36777 888 853 9797 713 345 4745 402 398 7454',
 '8018 148376 26 03 13 2000 02 28 8018 148376 03 13 2000 02 25 03 13 2000 02 09 8018 148376 11 35 0 148381 2 11',
 '2000 07 27 2001 01 46 07 19 2001 10 27 2000 2000 138 98663 508426 0 14 115 986431 20 508426 115 0 14 20 222 2000 2001 547063 0 14 713 830 6960',
 '2 2000 60 000 40 000 15 000',
 '26 26 3 00 15 1 3 00 8 00 20 8 00 20 3 7979 6 8 00 29 5 00 19',
 '5 8 1300 2100 5 2 900 2100 06 23 2000 09 43 06 23 2000 09 30 06 23 2000 09 29 5 8 5 2 5 6',
 '1 5 0 7 0 10 0 1 1 2 1 2 215 2 7 0 10 0 215 3 0 215',
 '1 2 2 99 00 2 256 29 90 3 2 9 50 4 56 23 90 5 56 25 50 6 1100 158 00 7 1160 278 00 8 2 8 80 9 3 1 52 00 10 16 90 11 107 50 12 15 90 13 2 1 49 00 14 4 2 0 5 50 15 8 49 00 16 3 19 50 17 5 1 36 00 18 2 1 58 00 19 9 50 20 168 16 36 00 21 17 50 22 52 24 52 6 34 90 23 99 00 24 12 90 25 1 5 2 0 127 00 256 56 3 971 4 8834464 971 4 8834454 4176 101 1 1618 105 4 48',
 '02 04 2000 2 56 250 20 000 40 000 3 30 000 60 000 4 75 000',
 '',
 '19 01 18 2000 11 11 _ _ 01 18 2000 10 48 53 19',
 '2000 02 16 2000 09 43 02 16 2000 09 37 09 2000 2000 2 17 00 1 075 12 400 12 400 5555 1200 77056 713 964 9434',
 '0 80 0 100',
 '4',
 '700 2700 77002 713 830 8659 713 830 8722 1 1',
 '5 2000 60 000 65 000 15 000',
 '2000 02 22 2000 01 44 2000 2000 2117 33321',
 '348725 6 10 6 30 348729 7 1 7 31 07 28 2000 01 24 9835 600 96 0 14 6 10 7 31 9845 3000 100 0 10 7 21 7 31 9847 800 100 0 13 7 25 7 31 3 35 3 6353',
 '2002 270 99 50 00 220 00 7 0 609 99 60 00 550 00 2002 579 99 60 00 510 00 10 270 99 60 00 210 00 11 270 99 60 00 210 00 7 404 99 60 00 335 00 7 _ 233763 _ 601 aa',
 '2004',
 '1300 1472 aa 100',
 '31 2000 8 27 aa 07 2000 5 56 55 2201 000 26 2000 6 00 23 2000 11 13 _ 23 2000 1 37 0 3 30 _ 73 2 _ _ 527 246 _ 44 423 07 999 26 74 991107 327 75 51 _ 22 2000 23 16 25 _ _ _ _ _ 991107 23 31 _ 9538 312 0 714 2 8 732412 22 2000 13 44 16 0500 1 0 5 5 2448 0 822 _ _ _ _ _ 22 2000 10 22 15 0500 1 0 5 5 2448 0 18 2000 5 20 94 17 2000 1 27 57 3093 3 7 200 5 aa 8297 aa 906 _ 8 32 100',
 '34 234 80 33792343 _ 3 100 000 9 2002 68 23 _',
```

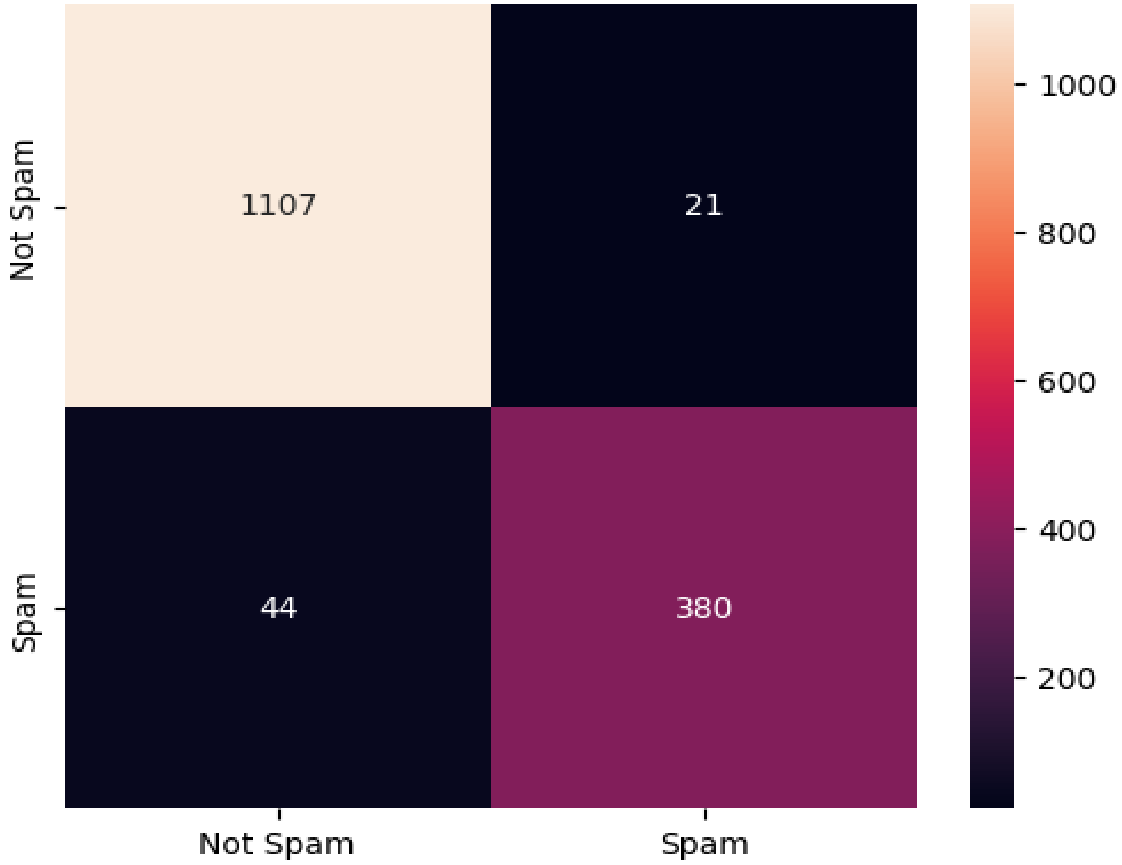

Accuracy

Accuracy: 0.7132731958762887

GaussianNB

```
▾ GaussianNB
GaussianNB()
```

Heat map



RandomForestClassifier

```
RandomForestClassifier
RandomForestClassifier()
```

Accuracy

Accuracy: 96.84278350515464

```
    unnamed:0  SP_LABEL                                            SP_TEXT  \
0         605         0  Subject: enron methanol ; meter # : 988291\r\n...
1        2349         0  Subject: hpl nom for january 9 , 2001\r\n( see...
2        3624         0  Subject: neon retreat\r\nho ho ho , we ' re ar...
3        4685         1  Subject: photoshop , windows , office . cheap ...
4        2030         0  Subject: re : indian springs\r\nthis deal is t...
5        2949         0  Subject: ehronline web address change\r\nthis ...
6        2793         0  Subject: spring savings certificate - take 30 ...
7        4185         1  Subject: looking for medication ? we ` re the ...
8        2641         0  Subject: noms / actual flow for 2 / 26\r\nwe a...
9        1870         0  Subject: nominations for oct . 21 - 23 , 2000\...
10       4922         1  Subject: vocable % rnd - word asceticism\r\nvc...
```

```
11       3799         1  Subject: report 01405 !\r\nwffur attion brom e...
12       1488         0  Subject: enron / hpl actuals for august 28 , 2...
13       3948         1  Subject: vic . odin n ^ ow\r\nberne hotbox car...
14       3418         0  Subject: tenaska iv july\r\ndarren :\r\nplease...
15       4791         1  Subject: underpriced issue with high return on...
16       2643         0  Subject: re : first delivery - wheeler operati...
17       3137         0  Subject: swift - may 2001 vols\r\nsean ,\r\nfy...
18       1629         0  Subject: meter variances - ua 4 clean - up\r\n...
19       1858         0  Subject: additional recruiting\r\ni ' m happy ...

    label_num
0           0
1           0
2           0
3           1
4           0
5           0
6           0
7           1
8           0
9           0
10          1
11          1
12          0
13          1
14          0
15          1
16          0
17          0
18          0
19          0
```

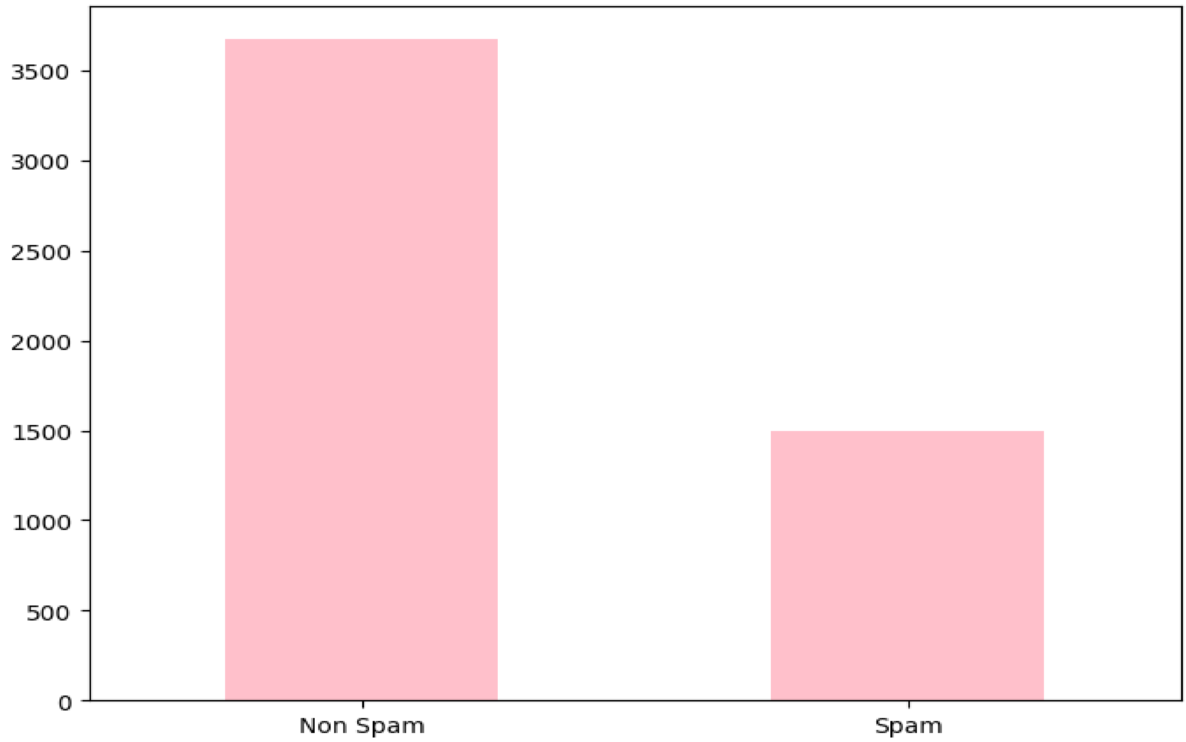
EXPLORATORY DATA ANALYSIS

Describe

	unnamed:0	SP_LABEL	label_num
count	5171.000000	5171.000000	5171.000000
mean	2585.000000	0.289886	0.289886
std	1492.883452	0.453753	0.453753
min	0.000000	0.000000	0.000000
25%	1292.500000	0.000000	0.000000
50%	2585.000000	0.000000	0.000000
75%	3877.500000	1.000000	1.000000
max	5170.000000	1.000000	1.000000

Univariate Analysis

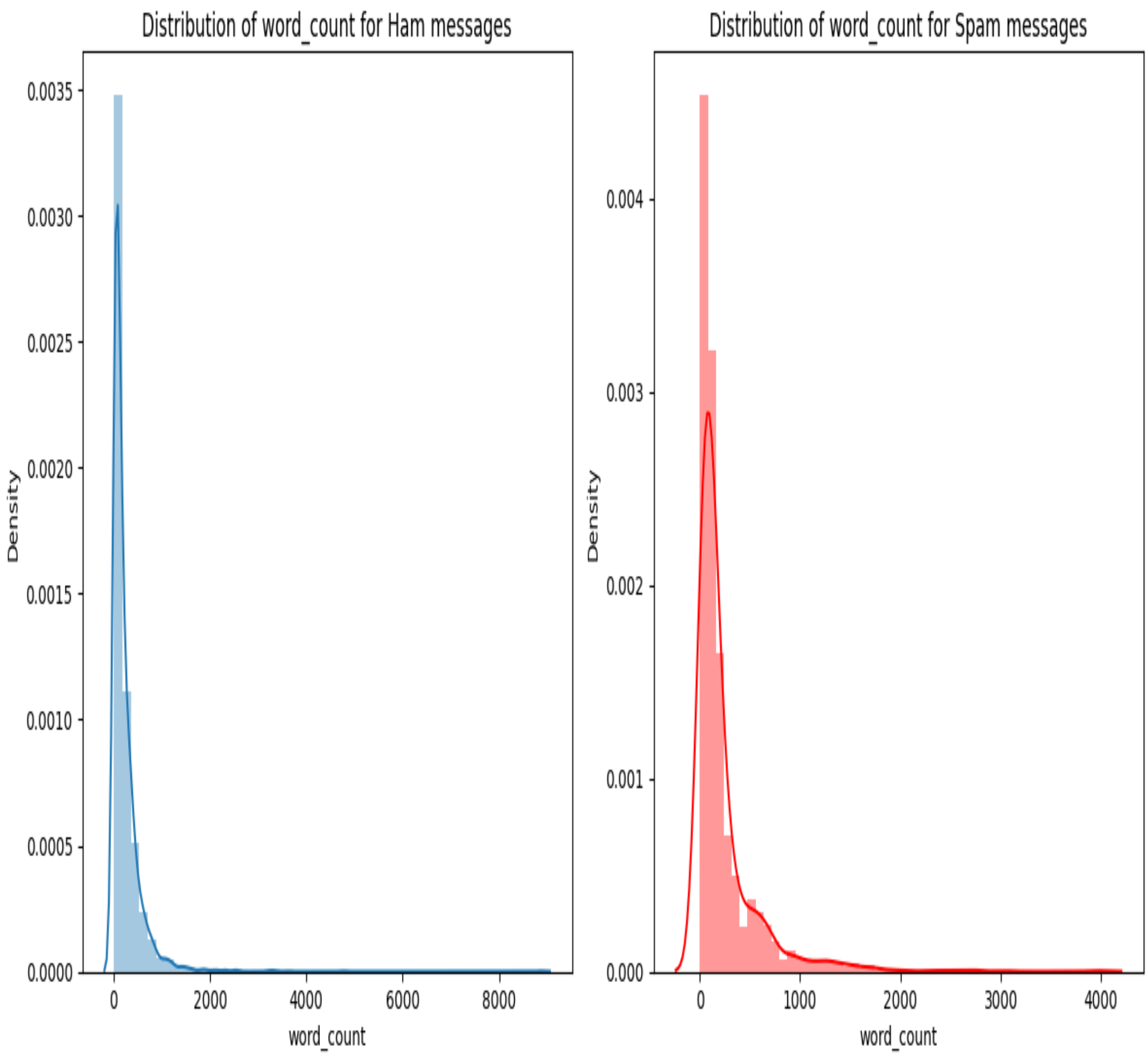
```
([<matplotlib.axis.XTick at 0x7f37e939adc0>,
  <matplotlib.axis.XTick at 0x7f37e939ad90>],
 [Text(0, 0, 'Non Spam'), Text(1, 0, 'Spam')])
```



Scaling the data

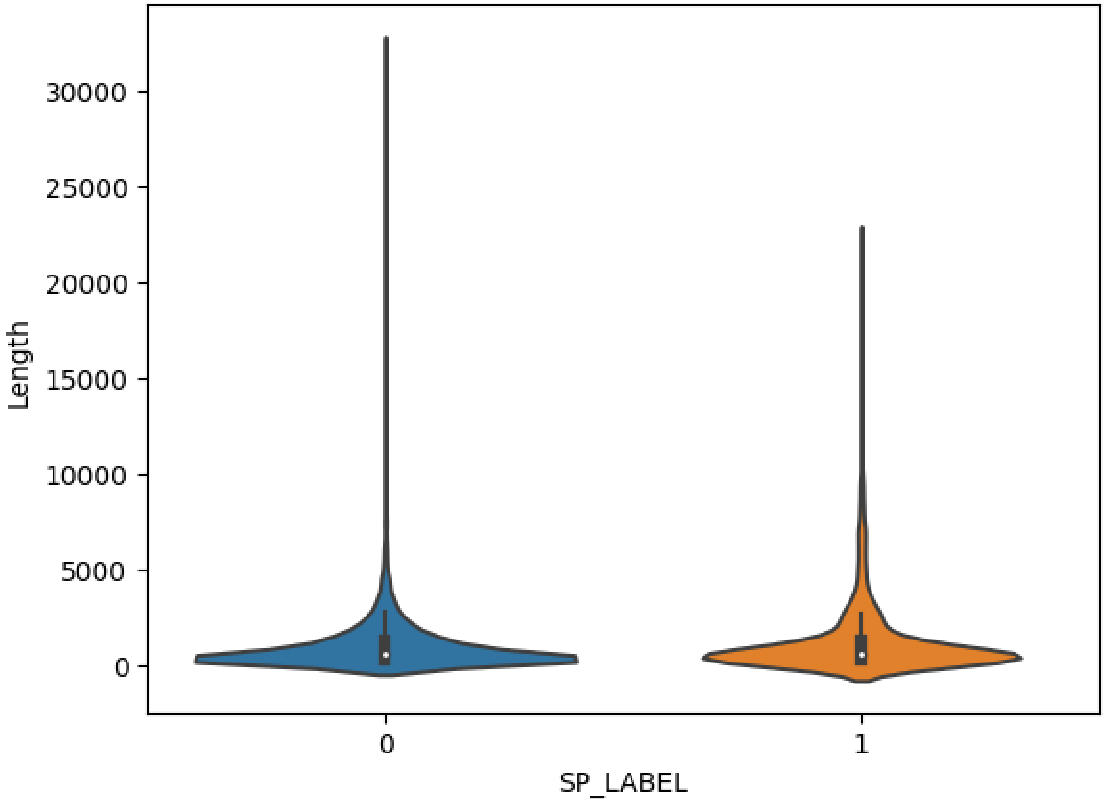
```
Number of Spam records: 1499
Number of Ham records: 3672
```

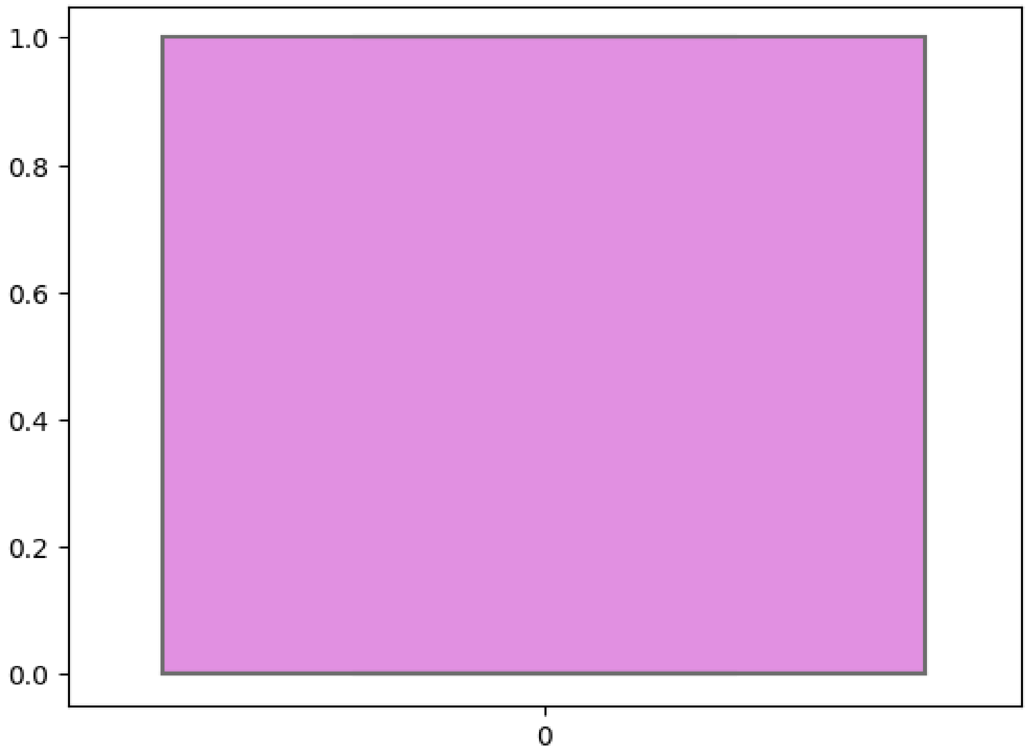
```
<ipython-input-42-ff21c60f164b>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  g = sns.distplot(a=df[df['label_num']==0].word_count)
<ipython-input-42-ff21c60f164b>:10: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  g = sns.distplot(a=df[df['label_num']==1].word_count, color='red')
```

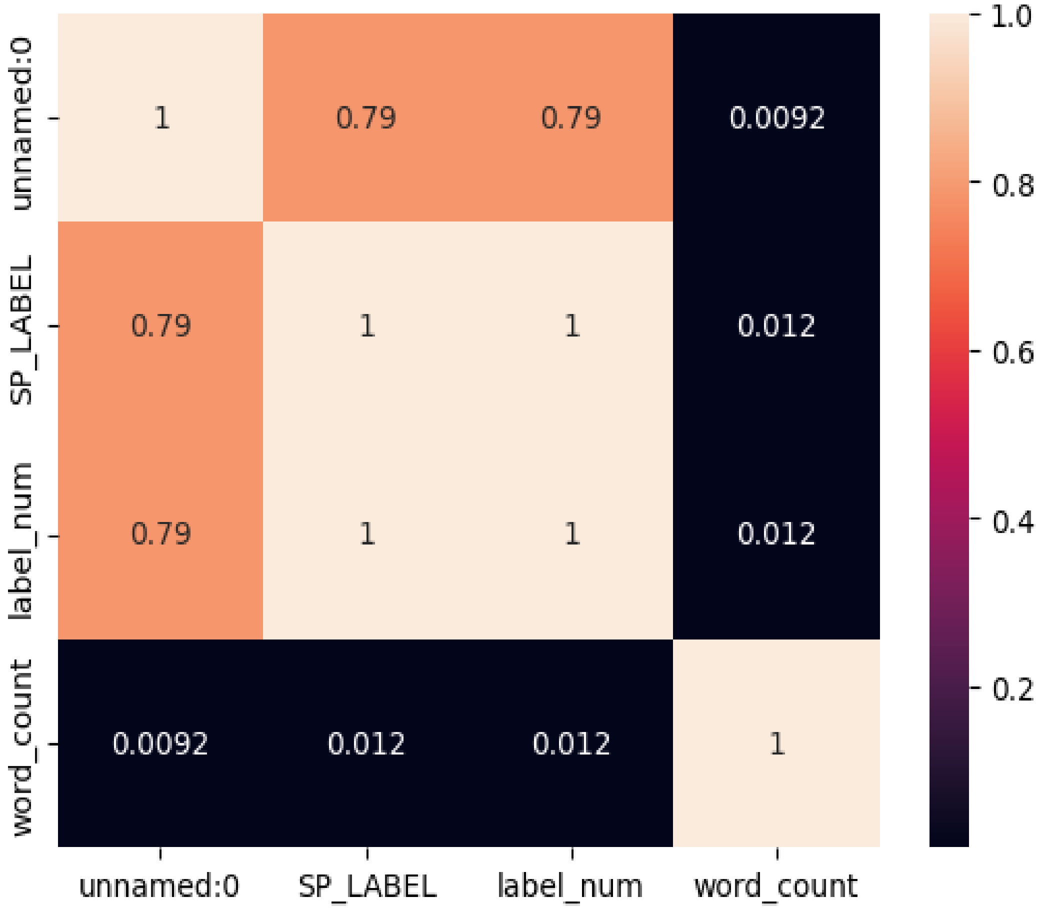


Splitting data into train and test

```
<Axes: xlabel='SP_LABEL', ylabel='Length'>
```


30000
25000
20000
15000
10000
5000
0
Length
0
1
SP_LABEL


1.0
0.8
0.6
0.4
0.2
0.0
0



Decision Tree

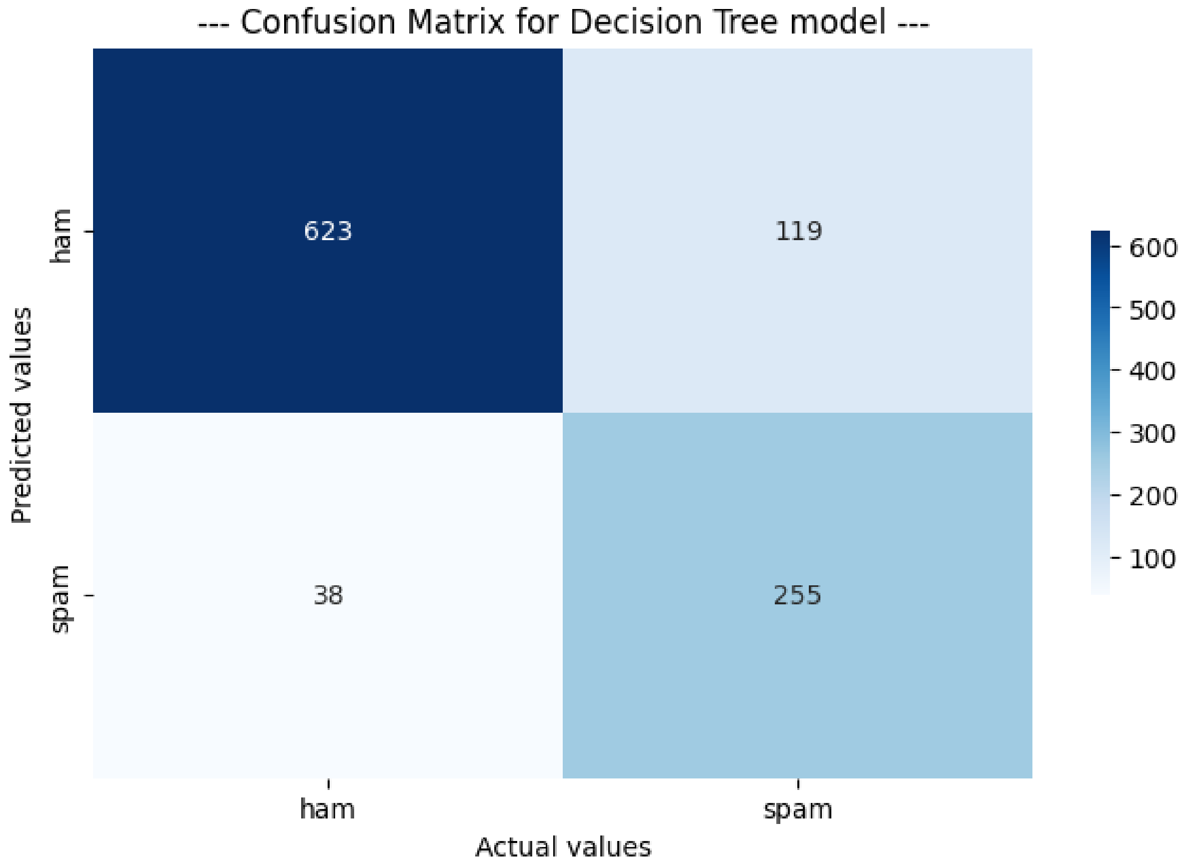
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

RandomForestClasifier

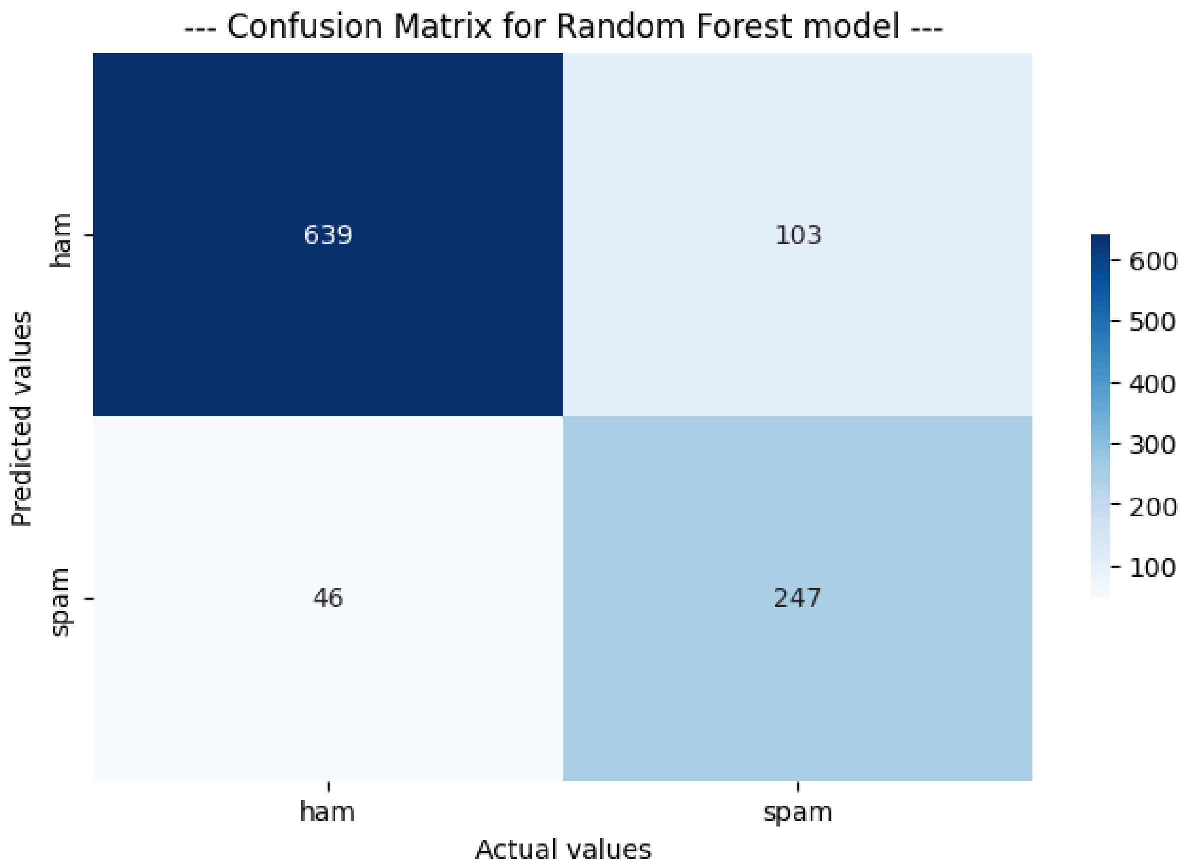
```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
▾ MultinomialNB
MultinomialNB()
```

```
--- Classification report for Decision Tree model ---
              precision    recall  f1-score   support

           0       0.94      0.84      0.89       742
           1       0.68      0.87      0.76       293

    accuracy                           0.85      1035
   macro avg       0.81      0.85      0.83      1035
weighted avg       0.87      0.85      0.85      1035
```




```
--- Classification report for Random Forest model ---
              precision    recall  f1-score   support

           0       0.93      0.86      0.90       742
           1       0.71      0.84      0.77       293

    accuracy                           0.86      1035
   macro avg       0.82      0.85      0.83      1035
weighted avg       0.87      0.86      0.86      1035
```




```
SP_TEXT 0 dtype: int64
```

```
Subject: calpine daily gas nomination\r\n>\r\nricky a . archer\r\nfuel supply\r\n700 louisiana , suite 2700\r\nhouston , texas 77002\r\n713 - 830 - 8659 direct\r\
Subject: \r\n
Subject: \r\nthis week only : f . ree gen . erlc vlag . ra\r\ncover the shipping , and we ' ll send youour product at no cost to prove its\r\neffectiveness .\r\n
Subject: we ' ve found a school for you !\r\n
Subject: you can be smart !\r\n

Subject: re : driscoll ranch # 3 gas pricing and interconnect estimate\r\ncan you help me out on this darren ? mjj\r\n- - - - - - - - - - - - - - - - - - - - -
Subject: jordyn , there is nothing like a dream to create the future .\r\nfor the sake of one good action a hundred evil ones should be forgotten\r\nto accomplish
Subject: from raymond bowen , jr . , exec . v . p . , finance & treasurer\r\nto : all enron employees\r\nfrom : raymond bowen , jr .\r\nevp , finance & treasurer\
Subject: re : indutrial report\r\nrobert ,\r\nthis is the file that i referenced in my last email . please get this file\r\ngoing again . thanks , pat\r\ndaren j
Subject: important online banking alert\r\ndear valued citizensr bank member ,\r\ndue to concerns , for the safety and integrity of the online banking community v
Name: SP_TEXT, Length: 4993, dtype: int64
```

There are 4 numerical variables

The numerical variables are : ['unnamed:0', 'SP_LABEL', 'label_num', 'word_count']

	unnamed:0	SP_LABEL	label_num	word_count
0	605	0	0	67
1	2349	0	0	23
2	3624	0	0	550
3	4685	1	1	48
4	2030	0	0	70

ANN MODEL

(4136, 4710)

```
33/33 [==============================] - 21s 534ms/step
```

```
array([0, 1, 0, ..., 1, 0, 0])
```

Compare the model

```
[[643  99]
[ 45 248]]
Accuracy Score Is:- 86.08695652173914
```

Compare and Tunnig the model

```
[[644  98]
[ 47 246]]
Accuracy Score Is:- 85.99033816425121
```

Integrate With Web Frame Work

Building HTML Pages

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<title> Login Page </title>
<style>
Body {
  font-family: Calibri, Helvetica, sans-serif;
  background-image:url("blacksp.jpg");
background-repeat:no-repeat;


}
h1
{
color:white;
}
</style>
```

```
</head>

<hr></div>

<body>

   <h1>OPTIMIZING </h1>

   <form action="{{url_for('predict')}}" method="post">

<p>



</p>                           <br><br><br><br><br><br>

          <label>Chech the mail: </label>

          <input type="text" placeholder="Enter mail" name="username" required>               

          <button type="submit">Predict</button> <br>



      </div>

   </form>

</body>

</html>
```

Building python code

```
import flask
from flask import Flask, render_template, request
import pickle
import sklearn
from flask_ngrok import run_with_ngrok
import warnings

warnings.filterwarnings('ignore')

app = Flask(__name__)
run_with_ngrok(app)

model = pickle.load(open('rdf.pkl', 'rb'))


@app.route('/', methods=['GET'])
def home():
    return render_template('tkm.html')


@app.route('/', methods=['GET', "POST"])
def predict():
    input_values = [float(x) for x in request.form.values()]
    inp_features = [input_values]
    print(inp_features )
    prediction = model.predict(inp_features)
    if prediction == 1:
        return render_template('tkm.html', prediction_text='Eligible to loan, Loan will
be sanctioned')
    else:
        return render_template('tkm.html', prediction_text='Not eligible to loan')


app.run()
```

Run the web application:

4. ADVANTAGES AND DISADVANTAGES

Advantages:

Machine learning can be a powerful tool for optimizing spam filtering, as it allows for automated identification and classification of spam messages based on patterns and features that are difficult for humans to identify. Here are some advantages of using machine learning for spam filtering:

 Automation: Machine learning algorithms can automatically learn from a large dataset of spam and non-spam messages, allowing for automated classification of incoming messages as spam or not.

 Scalability: Machine learning algorithms can be trained on large datasets of emails, allowing

for scalability to handle large volumes of incoming messages.

 Adaptability: Machine learning algorithms can adapt and improve over time, as they are exposed to new types of spam and non-spam messages. This allows for ongoing improvement of spam filtering accuracy.

 Accuracy: Machine learning algorithms can identify subtle patterns and features that are difficult for humans to detect, resulting in higher accuracy rates than traditional rule-based spam filters.

Disadvantages:

 Machine learning algorithms may incorrectly classify legitimate emails as spam, leading to important messages being missed.

 On the other hand, machine learning algorithms may also incorrectly classify spam emails as legitimate, leading to an increase in unwanted messages.

 Machine learning algorithms require large amounts of data to be trained on, but if the data is biased towards certain types of emails, the algorithm may not be effective in filtering out all types of spam.

 Machine learning algorithms require ongoing maintenance and updates to remain effective, which can be time-consuming and costly.

 Machine learning algorithms require access to personal data, which raises privacy concerns for users who may not want their data used for this purpose.

 Implementing machine learning algorithms for spam filtering requires technical expertise and resources, which may not be available for all organizations.

5. APPLICATION

- One potential application of optimizing spam filtering with machine learning is in email service providers. These providers could use machine learning algorithms to filter out spam emails from their users' inboxes, improving the overall user experience and reducing the risk of phishing attacks.

- To address the challenges mentioned above, email service providers could work to ensure that their machine learning algorithms are trained on diverse datasets to avoid bias, and regularly update and maintain the algorithms to keep up with new types of spam. They could also invest in technical expertise and resources to implement and optimize these algorithms, and be transparent about their data usage to address privacy concerns.

- Overall, optimizing spam filtering with machine learning has the potential to greatly improve email security and user experience, but it requires careful consideration of the challenges and resources needed for successful implementation.

6. CONCLUSION

At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. The finding of trust rank of the mail and classifying those mails as spam and ham mails based on their content.

A quantitative analysis of the use of feature selection algorithms and datasets was conducted. It was verified that the information gain is the most commonly used method for feature selection, although it has been suggested that others may lead to improved results when used with certain machine learning algorithms.

We have develop the machine learning model using python programming language an report are shown above

7. FUTURE SCOPE

- ✓ so far have not been able to remove the requirement of manual checking of the reviews.
- ✓ Hence there is scope for complete automation of spam detection systems with maximum efficiency
- ✓ With grow- ing popularity of online stores, the competition also increases

Also include:

- It increase security and control.
- It reduce IT Administration Costs and Network Resource Cost

8.APPENDIX

Importing libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
```

Read dataset

```
df = pd.read_csv("/content/spam_ham_dataset.csv")
df.head()
```

```
df.info
```

```
df.isna().sum
```

```
df.rename({"label":"SP_LABEL","text":"SP_TEXT_GI"},i
nplace=True,axis=1)
df.tail()
```

```
df.columns=['unnamed:0','SP_LABEL','SP_TEXT','label_num']
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['SP_LABEL']=le.fit_transform(df['SP_LABEL'])
```

```
df.head()
```

```
df.shape
```

```
X=df.loc[:,'SP_TEXT']
y=df.loc[:,'label_num'].values
```

```
X[1]
```

```
###GIVEN DATA IS IMBALANCED,WE BALANCE THE DATA

print("Befor over sampling, counts of label '1':{}".format(sum(y_trai
n==1)))
print("Befor over sampling, counts of label '0' :{} \n".format(sum(y_
train==0))
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer()

###train size 80% & test size 20%
```

```
from imblearn.over_sampling import SMOTE
sm=SMOTE(random_state=2)
X_train_res,y_train_res=sm.fit_resample(X_train,y_train.ravel())
print(" After oversampling, the shape of train_x:{}".format (X_train
_res.shape))
print(" After oversampling, the shape of train_y:{}\n".format (y_tra
in res.shape))
```

```
nltk.download("stopwords")
```

```
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

```
import re
corpus=[]
length=len(df)
```

```
for i in range(0,length):
  SP_TEXT=re.sub("[^a_ZA_Z0-9]"," ",df["SP_TEXT"][i])
  SP_TEXT=SP_TEXT.lower()
  SP_TEXT=SP_TEXT.split()
  pe=PorterStemmer()
  stopword=stopwords.words("english")
  SP_TEXT=[pe.stem(word)for word in SP_TEXT if not word in set (st
opword)]
  SP_TEXT=" ".join(SP_TEXT)
  corpus.append(SP_TEXT)
```

```
corpus
```

```
from sklearn.feature_extraction.text import CountVectorizer

cv=CountVectorizer(max_features=350000)
X=cv.fit_transform(corpus).toarray()
```

```
import pickle
pickle.dump(cv,open('cv1.pkl','wb'))
```

```
from sklearn.naive_bayes import GaussianNB
classifier=GaussianNB()
classifier.fit(X_train,y_train)
```

```
y_pred=classifier.predict(X_test)
from sklearn.metrics import confusion_matrix,accuracy_score
cm1=confusion_matrix(y_test,y_pred)
print("Accuracy:", accuracy_score(y_test,y_pred*100))
```

```
import seaborn as sns
plt.figure(figsize=(6,5))
sns.heatmap(cm1,annot= True, fmt='n',xticklabels=['Not Spam','Spam'
],yticklabels=['Not Spam','Spam'])
```

```
from sklearn.ensemble import RandomForestClassifier
cl=RandomForestClassifier()
cl.fit(X_train,y_train)
```

```
y_pred=cl.predict(X_test)
from sklearn.metrics import confusion_matrix,accuracy_score
cm2=confusion_matrix(y_test,y_pred)
print('Accuracy:',accuracy_score(y_test,y_pred)*100)
```

```
print(df.head(20))
```

EXPLORATORY DATA ANALYSIS:

```
df.describe()
```

```
df["SP_LABEL"].value_counts().plot(kind='bar',figsize=(8,6),color
='pink')
plt.xticks(np.arange(2),('Non Spam','Spam'),rotation=0)
```

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X=sc.fit_transform(X)
```

```
only_spam = df[df['label_num']==1]
print('Number of Spam records: {}'.format(only_spam.shape[0]
))
print('Number of Ham records: {}'.format(df.shape[0]-
only_spam.shape[0]))
```

```
# Creating new feature word_count
df['word_count'] = df['SP_TEXT'].apply(lambda x: len(x.spli
t()))
```

```
plt.figure(figsize=(12, 6))

# 1-row, 2-column, go to the first subplot
plt.subplot(1, 2, 1)
g = sns.distplot(a=df[df['label_num']==0].word_count)
p = plt.title('Distribution of word_count for Ham messages')

# 1-row, 2-column, go to the second subplot
plt.subplot(1, 2, 2)
g = sns.distplot(a=df[df['label_num']==1].word_count, color='red')
p = plt.title('Distribution of word_count for Spam messages')

plt.tight_layout()
plt.show()
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,
random_state=0)
```

```
df_visualizations=df.copy()
df_visualizations["Length"]=df_visualizations["SP_TEXT"].apply(lambda x
:len(x))
sns.violinplot(x="SP_LABEL",y="Length", data=df_visualizations)
```

```
sns.boxplot(df['SP_LABEL'],color='violet')
```

```
sns.heatmap(df.corr(),annot=True)
```

MODEL BUILDING

```
#Model Building
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
model.fit(X_train_res,y_train_res)
```

```
from sklearn.ensemble import RandomForestClassifier
model1=RandomForestClassifier()
model1.fit(X_train_res,y_train_res)
```

```
#Naive Bayee model
from sklearn.naive_bayes import MultinomialNB
model =MultinomialNB()
```

```
model.fit(X_train_res, y_train_res)
```

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=
0.2, random_state=42)
```

```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)

print('--- Classification report for Decision Tree model ---')
print(classification_report(y_test, y_pred))
```

```
# Confusion matrix of Decision Tree model
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8,5))
axis_labels = ['ham', 'spam']
g = sns.heatmap(data=cm, annot=True, cmap="Blues", xticklabels=axi
s_labels, yticklabels=axis_labels, fmt='g', cbar_kws={"shrink": 0.
5})
p = plt.xlabel('Actual values')
p = plt.ylabel('Predicted values')
p = plt.title('--- Confusion Matrix for Decision Tree model ---')
```

```
# Classification report for Random Forest model
rf = RandomForestClassifier(n_estimators=20)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)

print('--- Classification report for Random Forest model ---')
print(classification_report(y_test, y_pred))
```

```
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8,5))
axis_labels = ['ham', 'spam']
g = sns.heatmap(data=cm, annot=True, cmap="Blues", xticklabels=axis
_labels, yticklabels=axis_labels, fmt='g', cbar_kws={"shrink": 0.5}
)
p = plt.xlabel('Actual values')
p = plt.ylabel('Predicted values')
p = plt.title('--- Confusion Matrix for Random Forest model ---')
```

```
categorical = [var for var in df.columns if df[var].dtype=='O']
df[categorical].isnull().sum()
```

```
for var in categorical:

    print(df[var].value_counts())
```

```
numerical = [var for var in df.columns if df[var].dtype!='O']

print('There are {} numerical variables\n'.format(len(numerical))
)

print('The numerical variables are :', numerical)
```

```
df[numerical].head()
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

```
model=Sequential()
```

```
X_train.shape
```

```
model.add(Dense(units=X_train_res.shape[1],activation='relu',kernel_in
itializer="random_uniform"))
```

```
model.add(Dense(units=100,activation="relu",kernel_initializer="random
_uniform"))
```

```
model.add(Dense(units=100,activation="relu",kernel_initializer="random
_uniform"))
```

```
model.add(Dense(units=100,activation="sigmoid"))
```

```
model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accu
racy'])
```

```
import numpy as np
y_pr=np.where(y_pred>0.5,1,0)
```

```
y_pred=model.predict(X_test)
```

```
y_test
```

Performance Testing & Hyperparameter Tuning

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score=accuracy_score(y_test,y_pr)
print(cm)
print("Accuracy Score Is:-",score*100)
```


```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score=accuracy_score(y_test,y_pr)
print(cm)
print("Accuracy Score Is:-",score*100)
```