

**GOVERNMENT ARTS
COLLEGE
CHIDAMBARAM - 608 102**

Optimizing Spam Filtering – Using Machine Learning

TEAM MEMBERS NAME : G.Prem kumar

k.Pradeep

k.Pugazhmani

R.Ragavan

CLASS

: III-BCA COMPUTER
SCIENCE

TABLE OF INDEX

SNO	DESCRIPTION	PAGE NO:
1	Introduction	3
2	Problem Definition & Design Thinking	8
3	Result	16
4	Advantages And Disadvantages	34
5	Application	36
6	Conclusion	37
7	Future Scope	38
8	Appendix	39

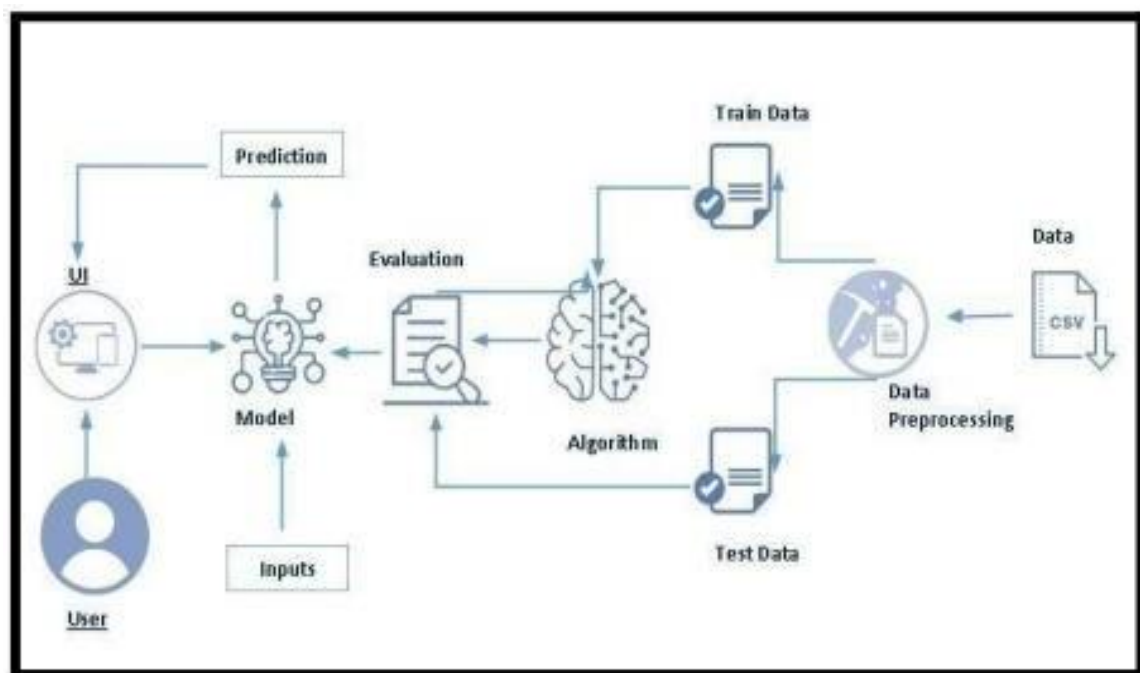
1. INTRODUCTION

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:



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

Name	Date Modified
>  Flask	24-01-2023 19:09
>  Images	24-01-2023 19:09
>  static	24-01-2023 19:09
✓  templates	25-01-2023 11:09
</> index.html	24-01-2023 19:09
</> result.html	25-01-2023 11:09
</> spam.html	24-01-2023 19:09
 app.py	25-01-2023 10:50
 app1.py	25-01-2023 11:09
 cv1.pkl	25-01-2023 10:48
 Procfile	24-01-2023 19:09
 requirements.txt	24-01-2023 19:09
 Spam SMS Classifier - Deployment.py	24-01-2023 19:09
 spam.h5	25-01-2023 09:21

- 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.

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

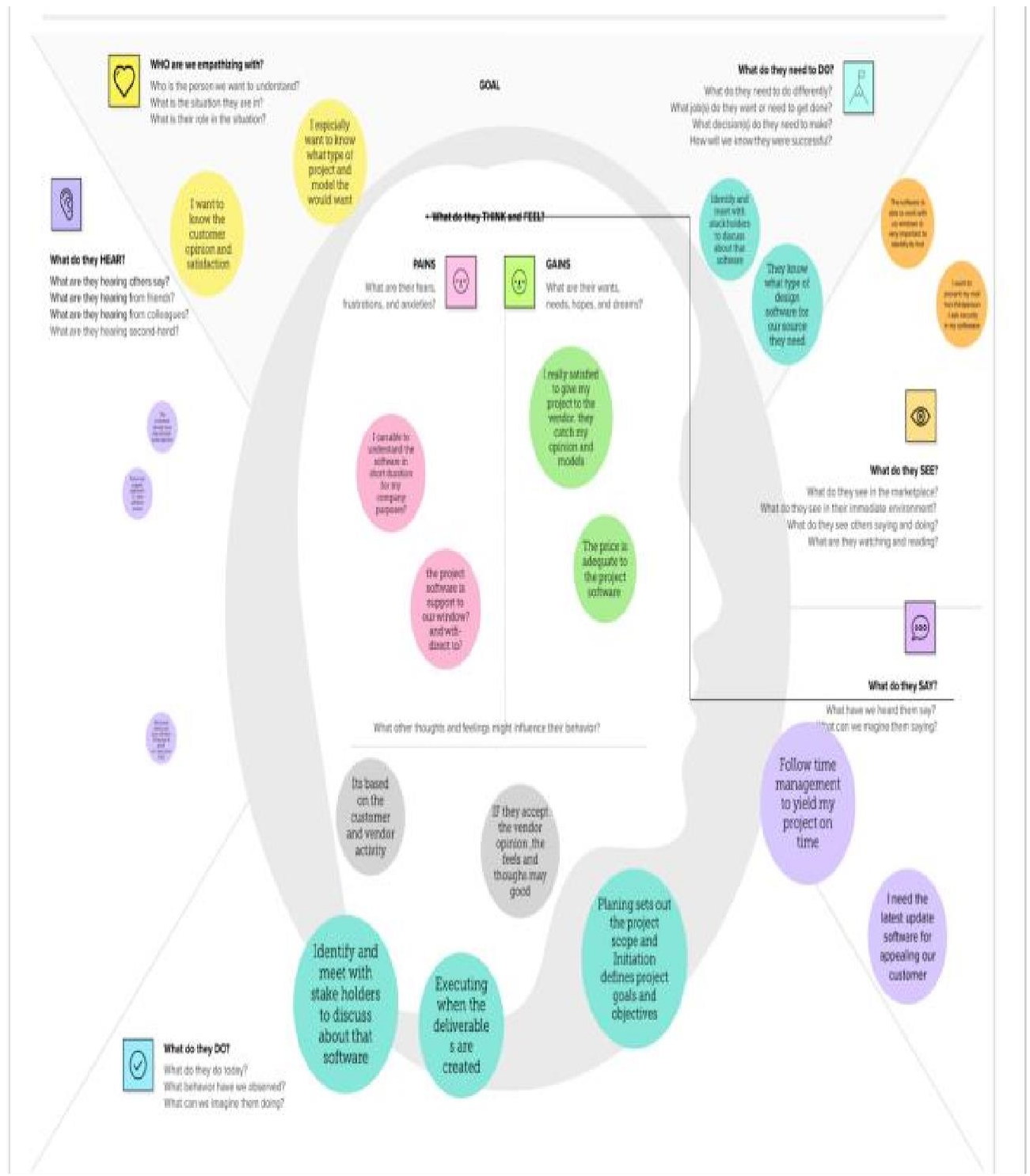
Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile 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

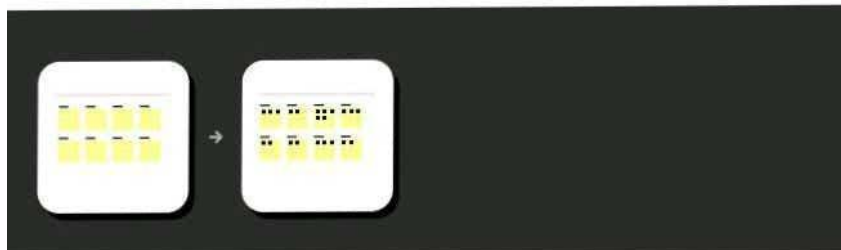
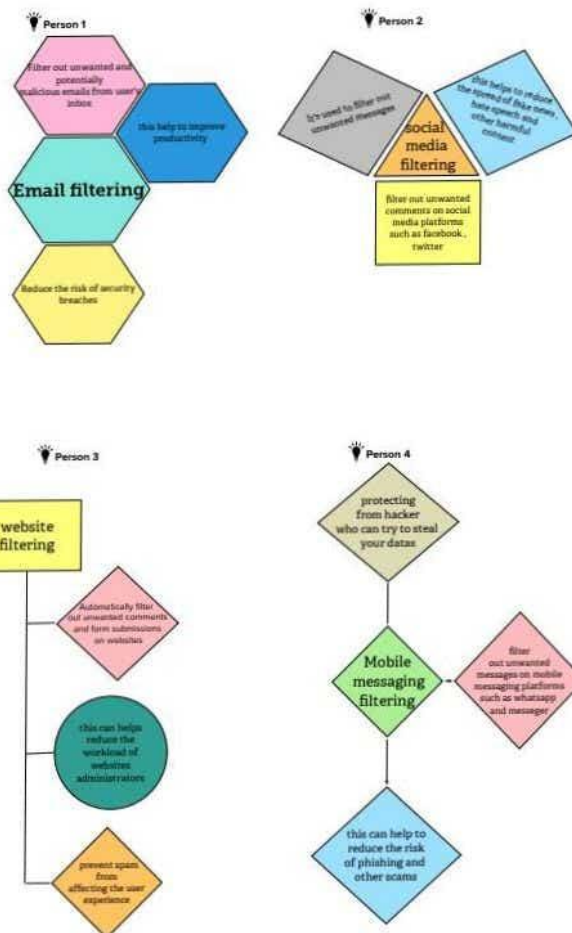
2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP
You can select a sticky note and hit the pencil (switch to sketch) icon to start drawing!

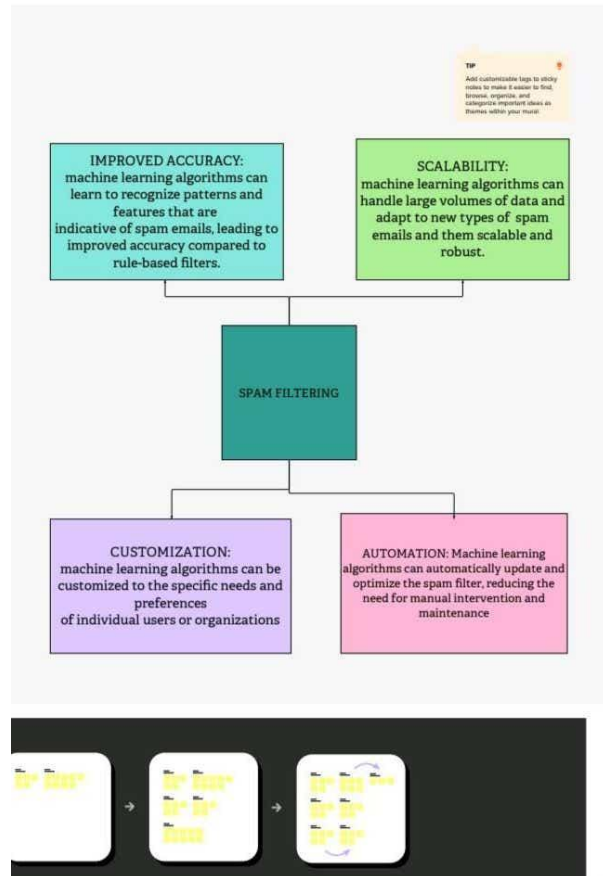


2

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 and break it up into smaller sub-groups.

20 minutes



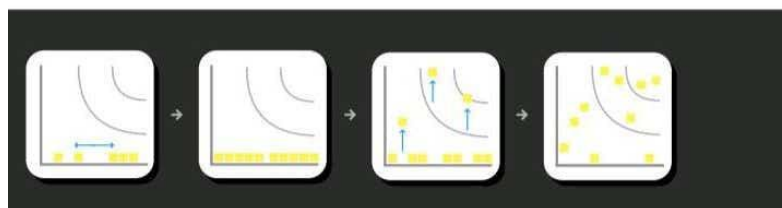
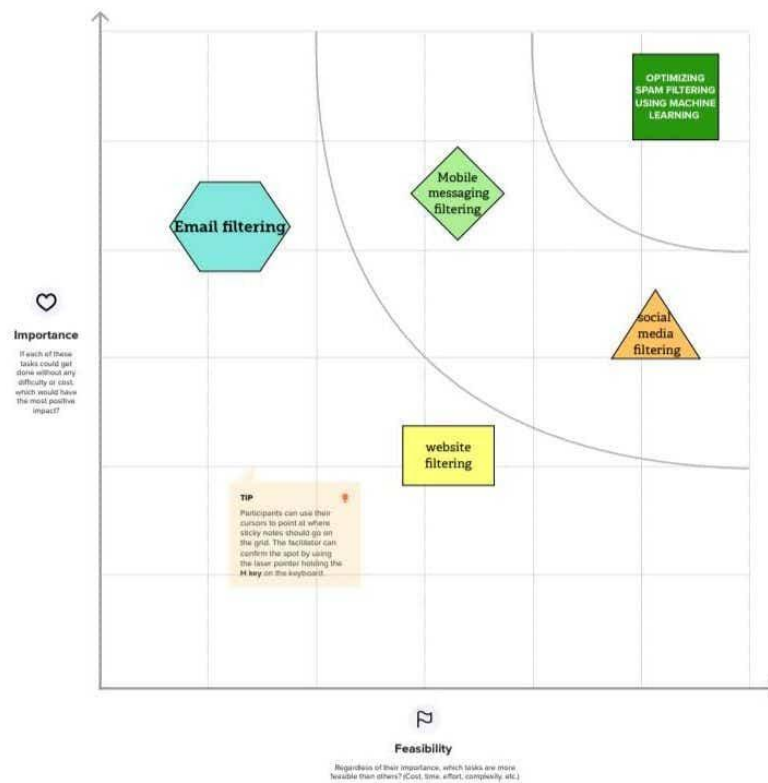
STEP-3: IDEA PRIORITIZATION

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⌚ 20 minutes





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



Share the mural

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



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

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   v1              5572 non-null   object
1   v2              5572 non-null   object
2   Unnamed: 2      50 non-null     object
3   Unnamed: 3      12 non-null     object
4   Unnamed: 4      6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

```
<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

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

Handling Categorical Values

Head

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

Shape:

```
(5572, 5)
```

Handling Imbalance Data

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

```
Befor over sampling, counts of label '1':581  
Befor over sampling, counts of label '0' :3876
```

```
After over sampling, counts of label '1':581  
After over sampling, counts of label '0' :3876
```

Cleaning the text data

Stopwords

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

Corpus

```
corpus
```

```
[',  
',  
'2 21 2005 87121 08452810075 18',  
,  
,  
'3 1 50',  
,  
'0',  
'000 09061701461 341 12',  
'11 08002906030',  
,  
'100 20 000 11 87575 150 6 16 4',  
'1 100 000 01010 4403 187 10',  
,
```

```
3000 ,
[ ] '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 _',
''''''
```

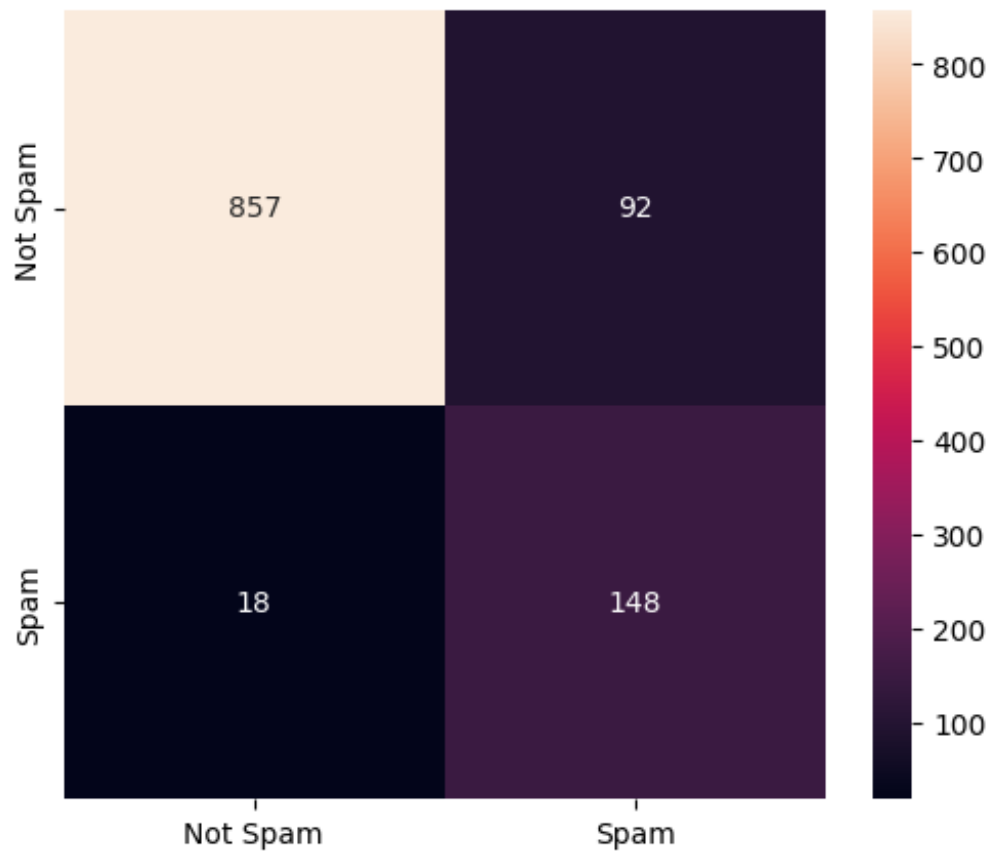
Accuracy

Accuracy: 0.768609865470852

GaussianNB

```
GaussianNB
GaussianNB()
```

Heat map



RandomForestClassifier

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

Accuracy

Accuracy: 97.21973094170404

```

label                                text unnamed:0 \
0      0  Go until jurong point, crazy.. Available only ...  NaN
1      0      Ok lar... Joking wif u oni...  NaN
2      1  Free entry in 2 a wkly comp to win FA Cup fina...  NaN
3      0  U dun say so early hor... U c already then say...  NaN
4      0  Nah I don't think he goes to usf, he lives aro...  NaN
5      1  FreeMsg Hey there darling it's been 3 week's n...  NaN
6      0  Even my brother is not like to speak with me. ...  NaN
7      0  As per your request 'Melle Melle (Oru Minnamin...  NaN
8      1  WINNER!! As a valued network customer you have...  NaN
9      1  Had your mobile 11 months or more? U R entitle...  NaN
10     0  I'm gonna be home soon and i don't want to tal...  NaN
11     1  SIX chances to win CASH! From 100 to 20,000 po...  NaN
12     1  URGENT! You have won a 1 week FREE membership ...  NaN
13     0  I've been searching for the right words to tha...  NaN
14     0      I HAVE A DATE ON SUNDAY WITH WILL!!  NaN

--
18     0  Fine if that's the way u feel. That's the wa...  NaN
19     1  England v Macedonia - dont miss the goals/team...  NaN

unnamed:1 unnamed:2
0      NaN      NaN
1      NaN      NaN
2      NaN      NaN
3      NaN      NaN
4      NaN      NaN
5      NaN      NaN
6      NaN      NaN
7      NaN      NaN
8      NaN      NaN
9      NaN      NaN
10     NaN      NaN
11     NaN      NaN
12     NaN      NaN
13     NaN      NaN
14     NaN      NaN
15     NaN      NaN
16     NaN      NaN
17     M>M      M>M

```

EXPLORATORY DATA ANALYSIS

Describe

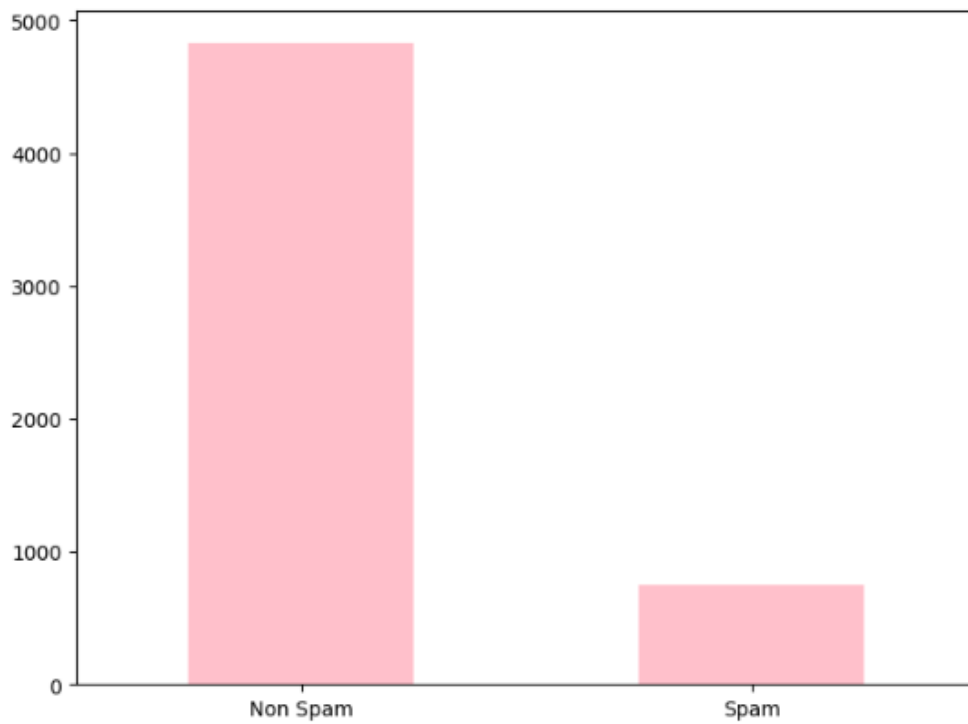
	label
count	5572.000000
mean	0.134063
std	0.340751
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

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: 747
Number of Ham records: 4825

```
<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)
```

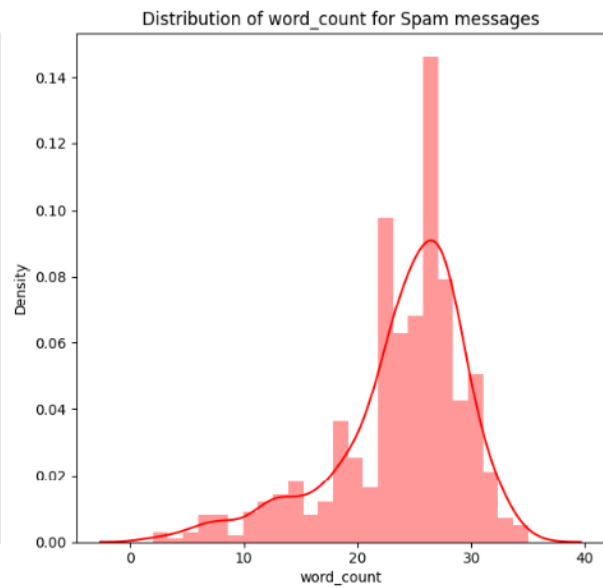
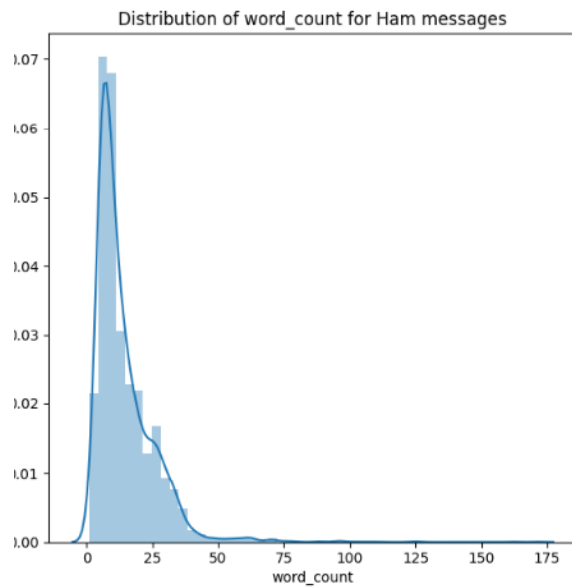
```
<ipython-input-42-ff21c60f164b>:10: UserWarning:
```

```
`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).

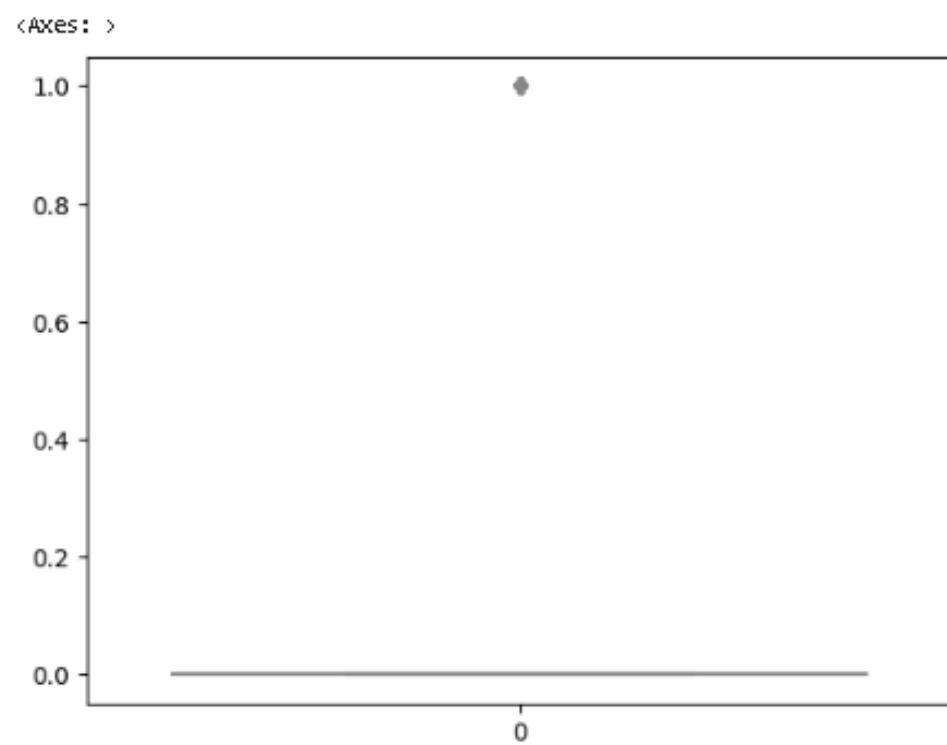
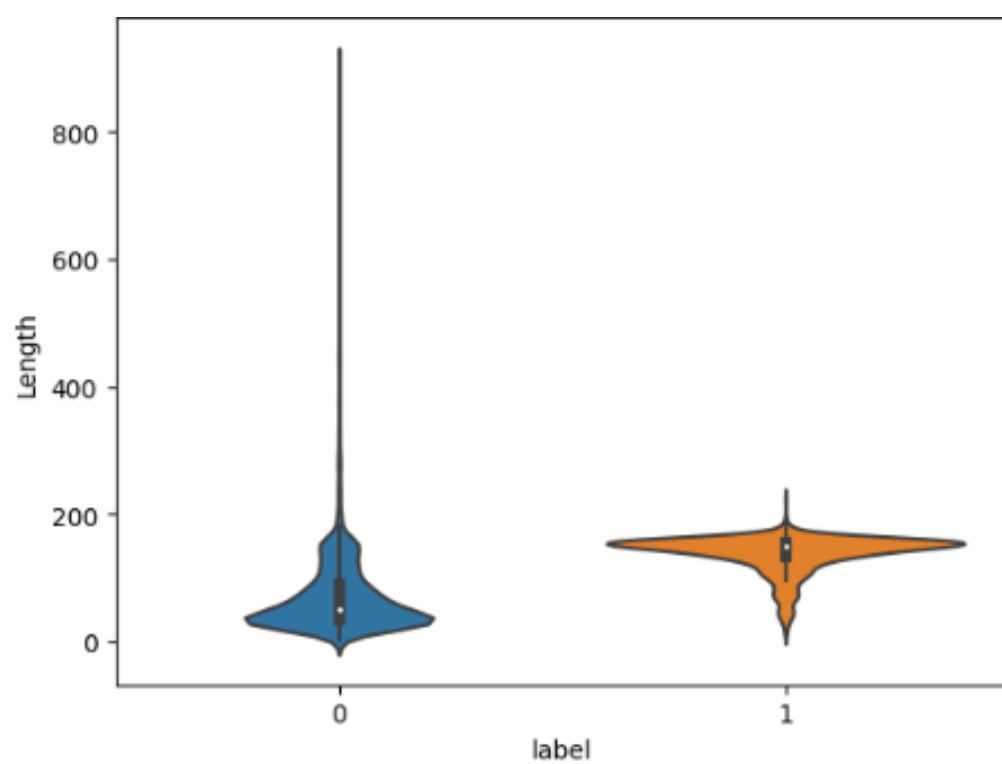
For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

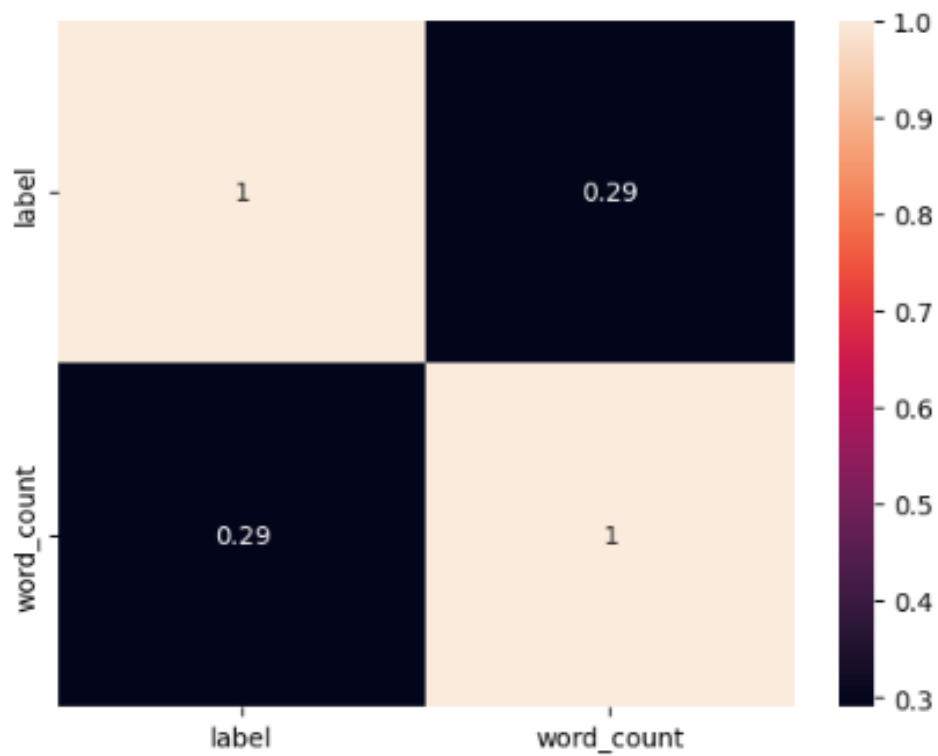
```
g = sns.distplot(a=df[df['label_num']==1].word_count, color='red')
```

Splitting data into train and test

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





Decision Tree

```
▼ DecisionTreeClassifier  
DecisionTreeClassifier()
```

RandomForestClassifier

```
▼ RandomForestClassifier  
RandomForestClassifier()
```

```
▼ MultinomialNB  
MultinomialNB()
```

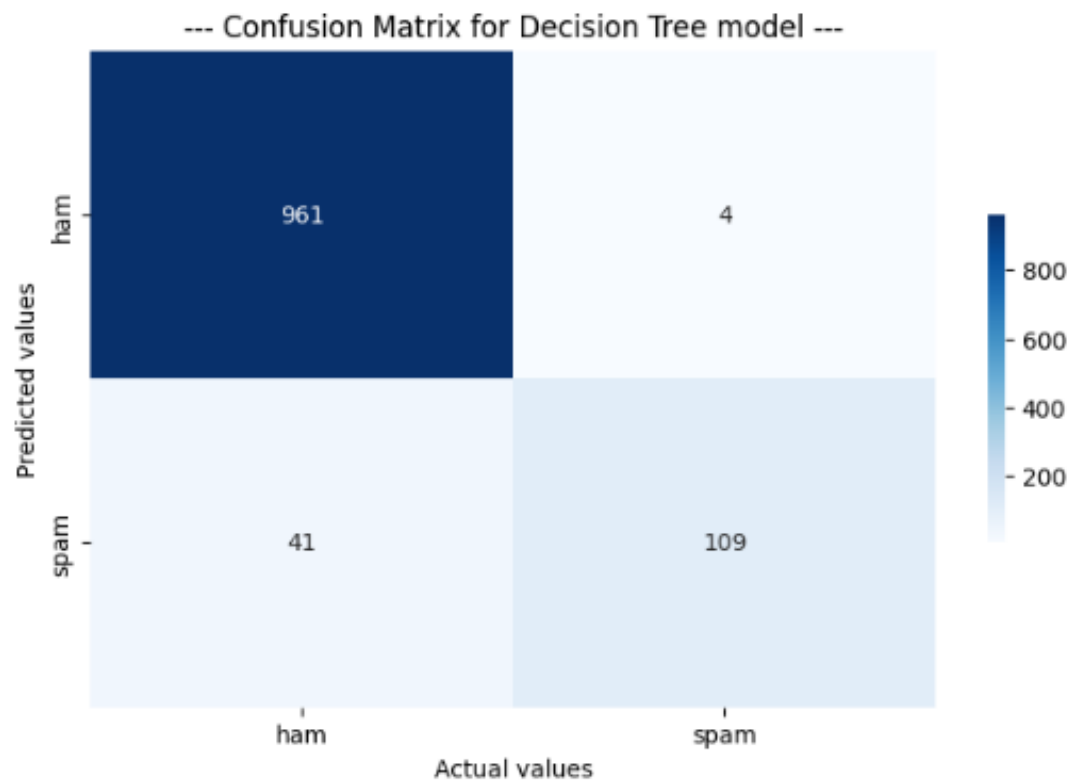
```

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

     0       0.96       1.00       0.98        965
     1       0.96       0.73       0.83        150

 accuracy          0.96          0.96        1115
 macro avg          0.96          0.86          0.90        1115
 weighted avg       0.96          0.96          0.96        1115

```



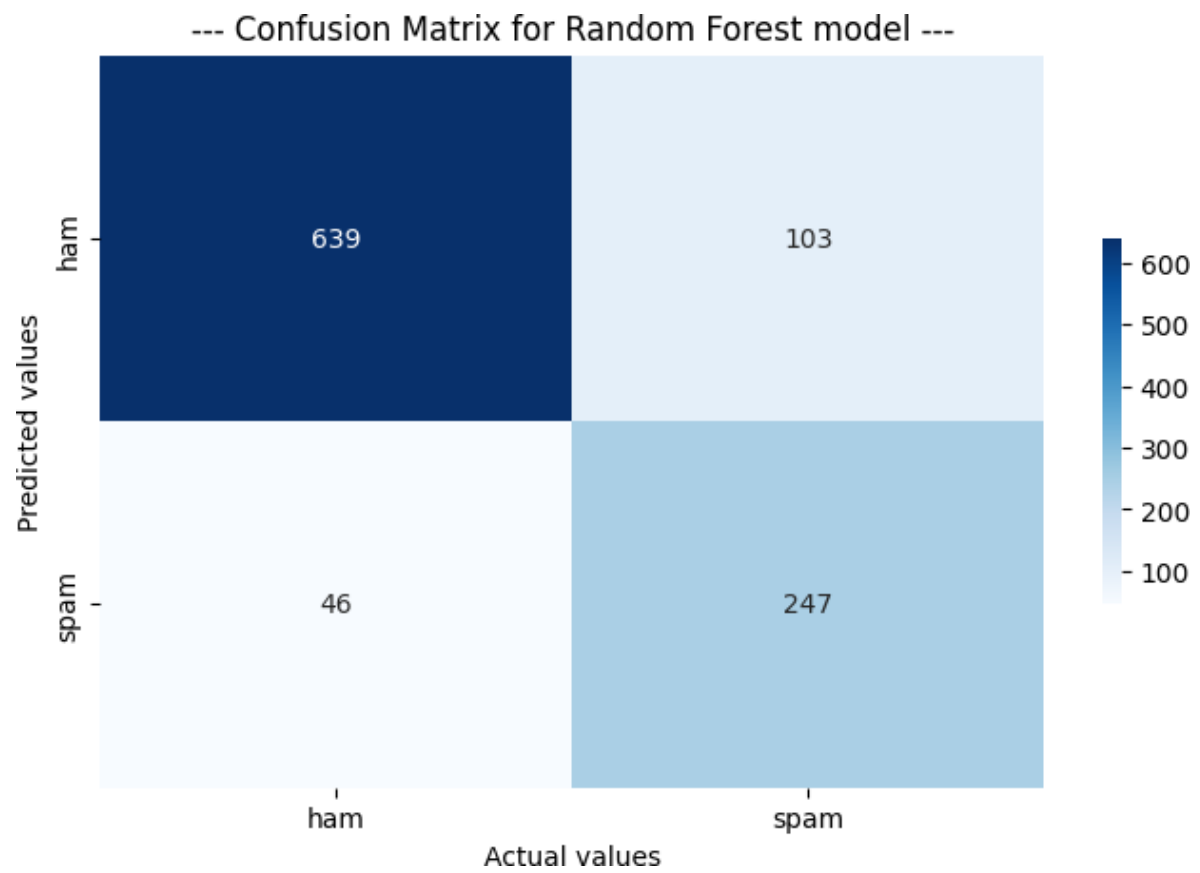
```

--- 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          0.86        1035
 macro avg          0.82          0.85          0.83        1035
 weighted avg       0.87          0.86          0.86        1035

```



```
text 0 dtype: int64
```

Subject: calpine daily gas nomination\r\n\r\nricky a . archer\r\n\r\nfuel supply\r\n\r\n700 louisiana , suite 2700\r\n\r\nhouston , texas 77002\r\n\r\n713 - 830 - 8659 direct\r\n\r\nSubject: \r\n\r\nSubject: \r\n\r\nthis week only : f . ree gen . eric vlag . ra\r\n\r\ncover the shipping , and we ' ll send you our product at no cost to prove its\r\n\r\nineffectiveness .\r\n\r\nSubject: we ' ve found a school for you !\r\n\r\nSubject: you can be smart !\r\n\r\nSubject: re : driscoll ranch # 3 gas pricing and interconnect estimate\r\n\r\ncan you help me out on this darren ? mjj\r\n\r\n- - - - -
Subject: jordyn , there is nothing like a dream to create the future .\r\n\r\nfor the sake of one good action a hundred evil ones should be forgotten\r\n\r\nto accomplish
Subject: from raymond bowen , jr . , exec . v . p . , finance & treasurer\r\n\r\nto : all enron employees\r\n\r\nfrom : raymond bowen , jr .\r\n\r\nnevp , finance & treasurer
Subject: re : industrial report\r\n\r\nrobert ,\r\n\r\nthis is the file that i referenced in my last email . please get this file\r\n\r\nongoing again . thanks , pat\r\n\r\ndarren j
Subject: important online banking alert\r\n\r\ndeared valued citizens\r\n\r\nbank member ,\r\n\r\ndue to concerns , for the safety and integrity of the online banking community
Name: SP_TEXT, Length: 4993, dtype: int64

There are 2 numerical variables

The numerical variables are : ['label_num', 'word_count']

	label	word_count
0	0	20
1	0	6
2	1	28
3	0	11
4	0	13

ANN MODEL

(4136, 4710)

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

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

Compare the model

```
[[[959 6]
 [ 34 116]]
Accuracy Score Is:- 96.41255605381166
```

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>

<body>

OPTIMIZING

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

$\langle p \rangle$

[illegible]

<label>Check the mail: </label>

```
<input type="text" placeholder="Enter mail" name="username"  
required>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;  
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
```

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

</body>

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.csv")df.head(5)
df.head()
```

```
df.info
```

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

```
df.rename({"label": "label", "text": "text"}, inplace=True, axis=1)
df.tail()
```

```
df.columns=['label', 'text', 'unnamed:0', 'unnamed:1', 'unnamed:2']
```

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

```
df.head()
```



```
df.shape
```

```
X=df.loc[:, 'text']  
y=df.loc[:, 'label'].values
```

```
x[1]
```

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

```
from sklearn.feature_extraction.text import CountVectorizer  
cv=CountVectorizer(max_features=35000)
```

```
###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_train_res
```

```
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):
    text=re.sub("[^a-zA-Z0-9]"," ",df["text"][i])
    text=text.lower()
    text=text.split()
    pe=PorterStemmer()
    stopword=stopwords.words("english")
    text=[pe.stem(word) for word in text if not word in set (stopword)]
    text=" ".join(text)
    corpus.append(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["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']==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['text'].apply(lambda x: len(x.split()))
```

```
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']==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']==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["text"].apply(lambda x
:len(x))
sns.violinplot(x="label",y="Length", data=df_visualizations)
```

```
sns.boxplot(df['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 Bayes 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=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 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=['accuracy'])
```

```
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)
```