GOVERNMENT OF TAMILNADU

Naan Muthalvan - Project-Based Experiential Learning

Optimizing Spam Filtering With Machine Learning

Submitted by

TEAM ID- NM2023TMID22894

S. SARANYA - (20326ER032)

P. SABITHA - (20326ER030)

M. SARANYA - (20326ER031)

N. SHANMUGA PRIYA – (20326ER033)

Under the guidance of

Mrs. J. SUKANYA, MCA., M.Phil.,

Assistant Professor

PG and Research Department of Computer Science

M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN

(Affiliated To Mother Teresa Women's University, Kodaikanal)

Reaccredited with "A" Grade by NAAC

DINDIGUL-624001.

APRIL - 2023

M.V.MUTHIAH GOVERNMENT ARTS COLLEG FOR WOMEN
(Affiliated to Mother Teresa Women's University, Kodaikanal)
Reaccredited with "A" Grade by NAAC
Dindigul - 624 001

PG & RESEARCH DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this is a bonafide record of the project entitled, '**OPTIMIZING SPAM FILTERING WITH MACHINE LEARING' by S.SARANYA (20326ER032) , P.SABITHA (20326ER030) , M.SARANYA (20326ER031) and N.SHANMUGA PRIYA (20326ER033)**. This is submitted in partial fulfillment for the award of the degree of **Bachelor of Science in Computer Science in M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN,DINDIGUL** during the period of December 2022 to April 2023.

Project Mentor(s) **Head of the Department**

Submitted for viva-voce Examination held on 11.04.2023

TABLE OF THE CONTENT

1. INTRODUCTION

1.1 OVERVIEW

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.

1.2 PURPOSE

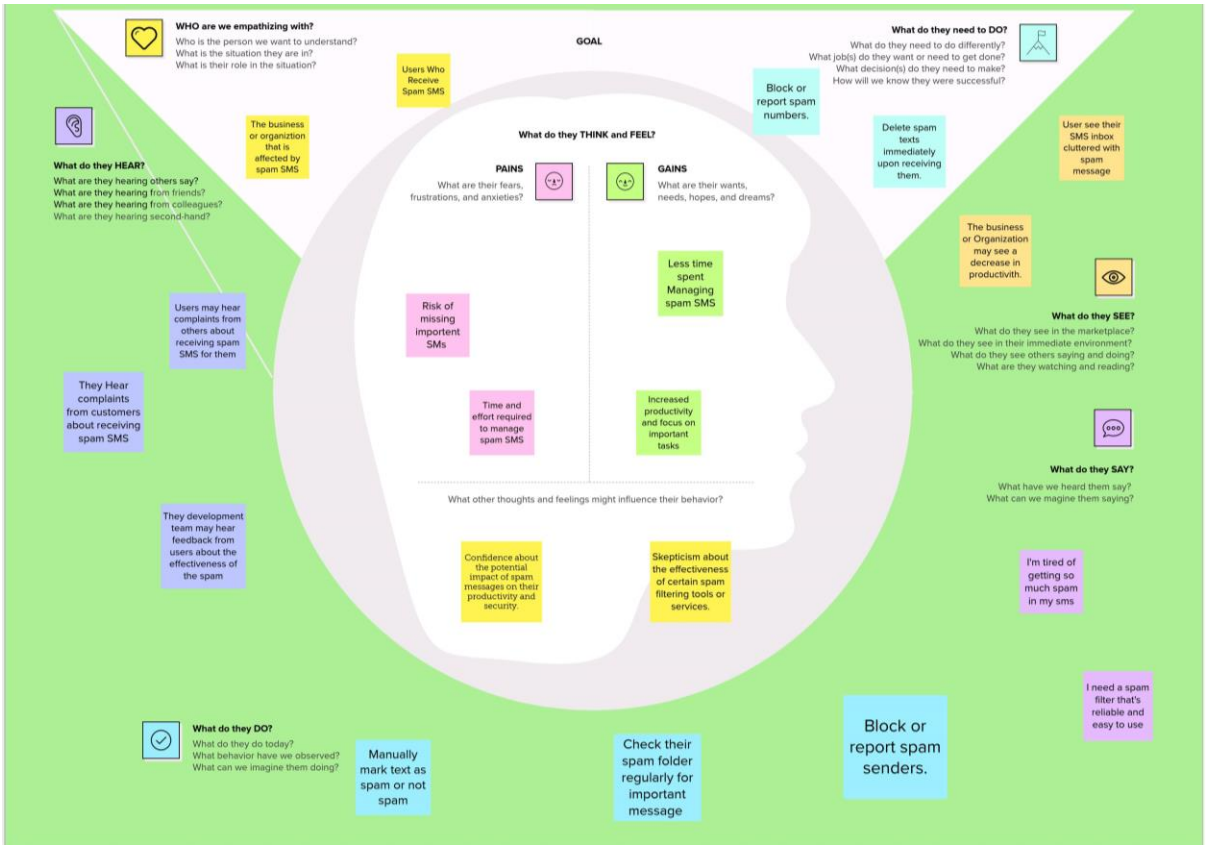
Spam SMS messages are a persistent problem for mobile phone users. These unwanted messages can be a source of annoyance and may even contain malicious content. Machine learning techniques can be used to effectively filter out spam SMS messages, reducing the amount of unwanted messages that users receive. In this project, we will explore the use of machine learning algorithms to classify SMS messages as either spam or not spam. We will use a dataset of SMS messages that have been previously labeled as spam or not spam, and train our machine learning model on this data. Our goal is to create a model that can accurately classify SMS messages as spam or not spam, based on their content. This will involve pre-processing the data, selecting appropriate features, and training and testing several different machine learning algorithms. We will evaluate the performance of each algorithm using various metrics and select the best-performing model. Ultimately, our project aims to demonstrate the effectiveness of machine learning techniques in solving real-world problems, and to provide a practical solution to the problem of spam SMS messages.

2. DEFINE PROBLEM AND PROBLEM UNDERSTANDING

In this milestone, we will see the define problem and problem understanding.

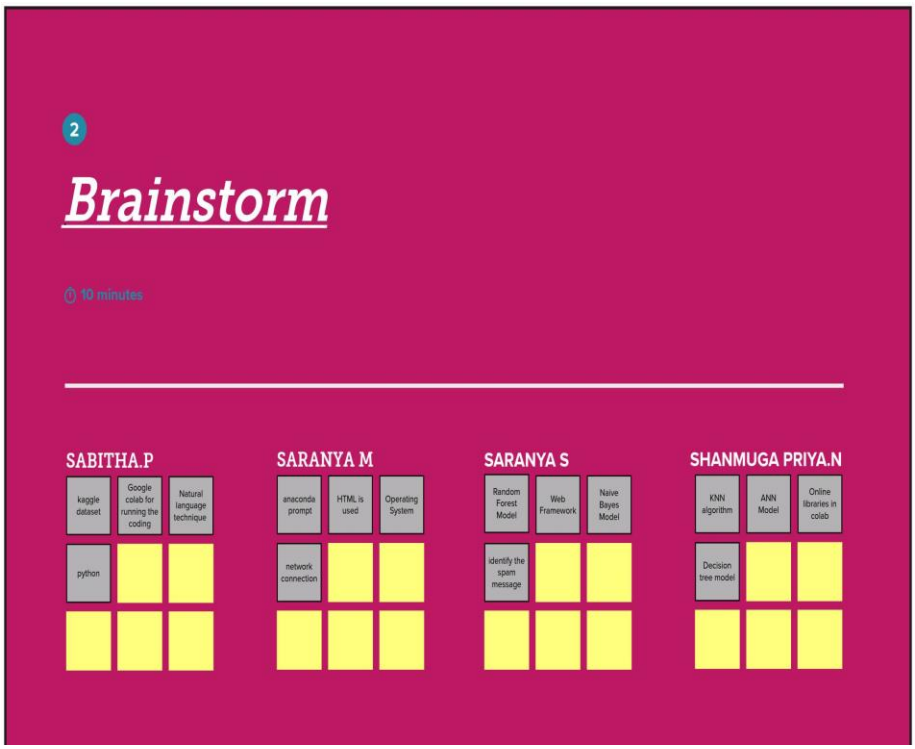
2.1 Empathy map

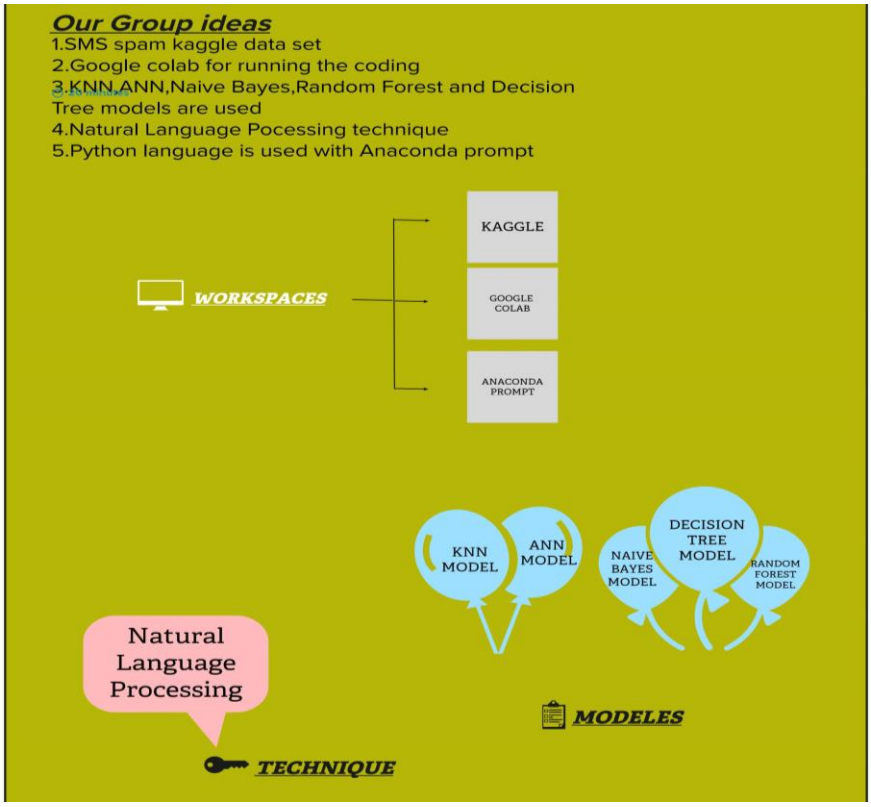
Empathy in this case can refer to the ability of the machine learning algorithm to understand the context and nuances of the data it is analysing. This includes understanding the factors that may impact the placement of students, such as their academic performance, background, and the current job market.



2.2 Ideation and Brainstorm

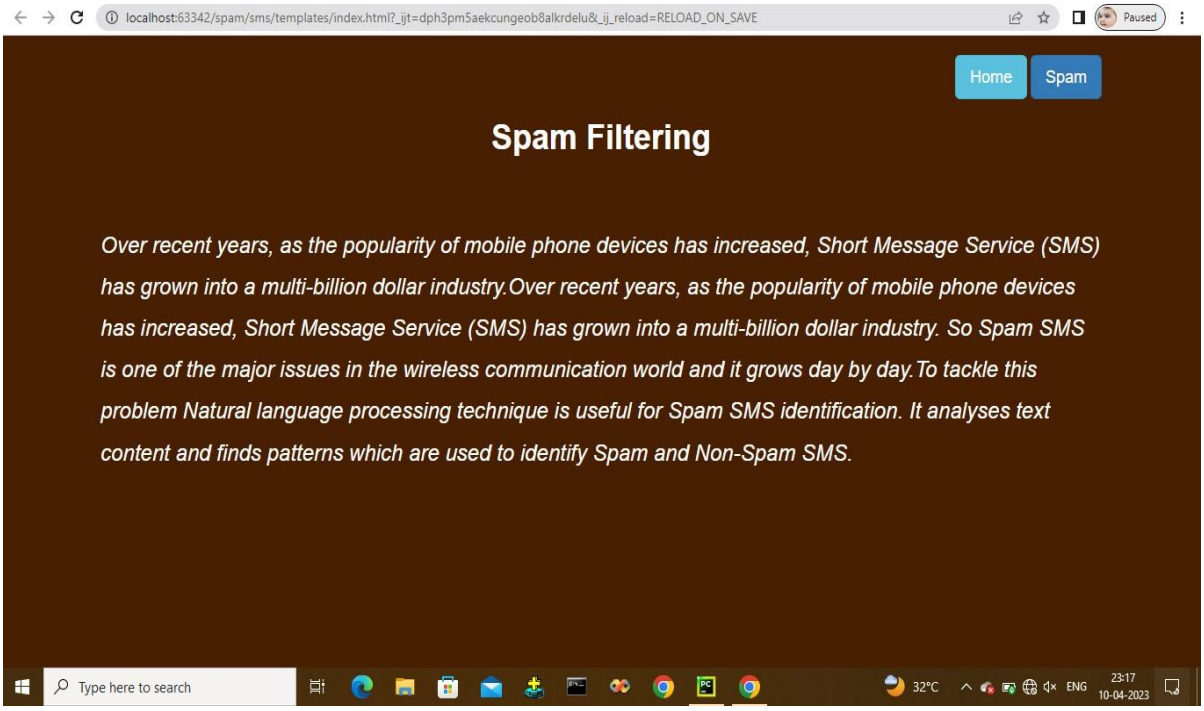
Brainstorm Map for Identifying Patterns and Trends in Campus Placement Data using Machine Learning

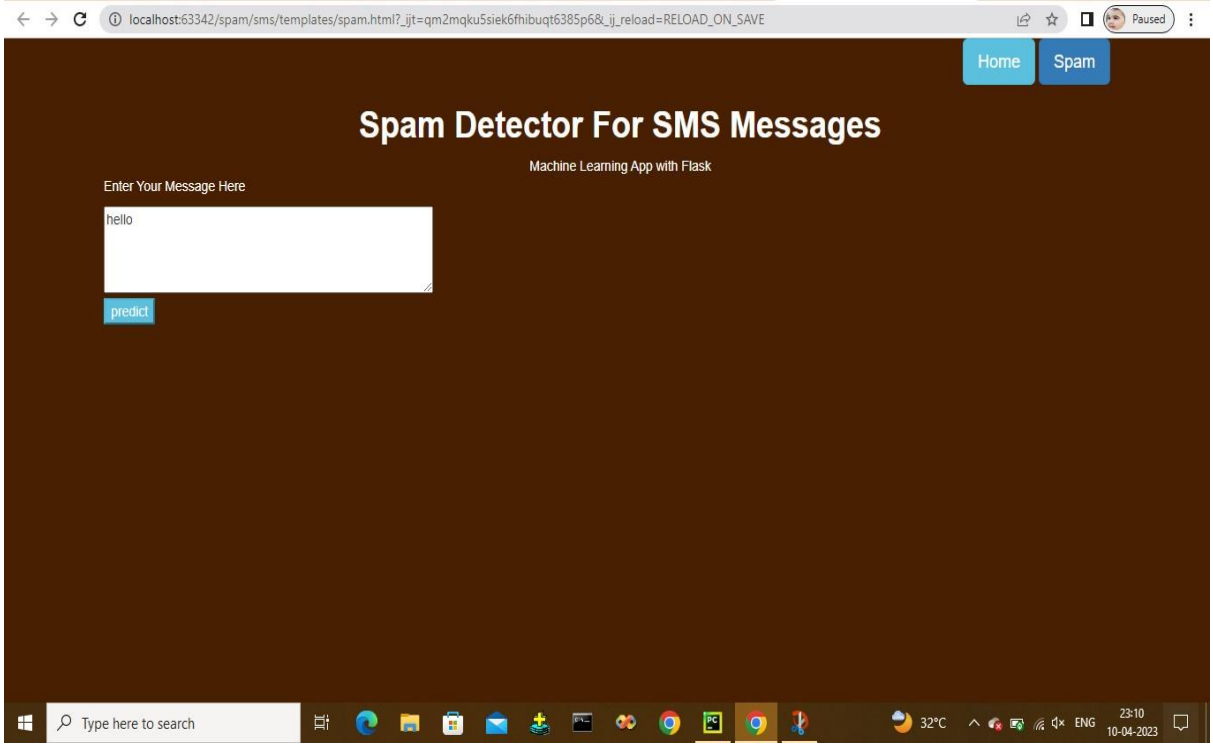


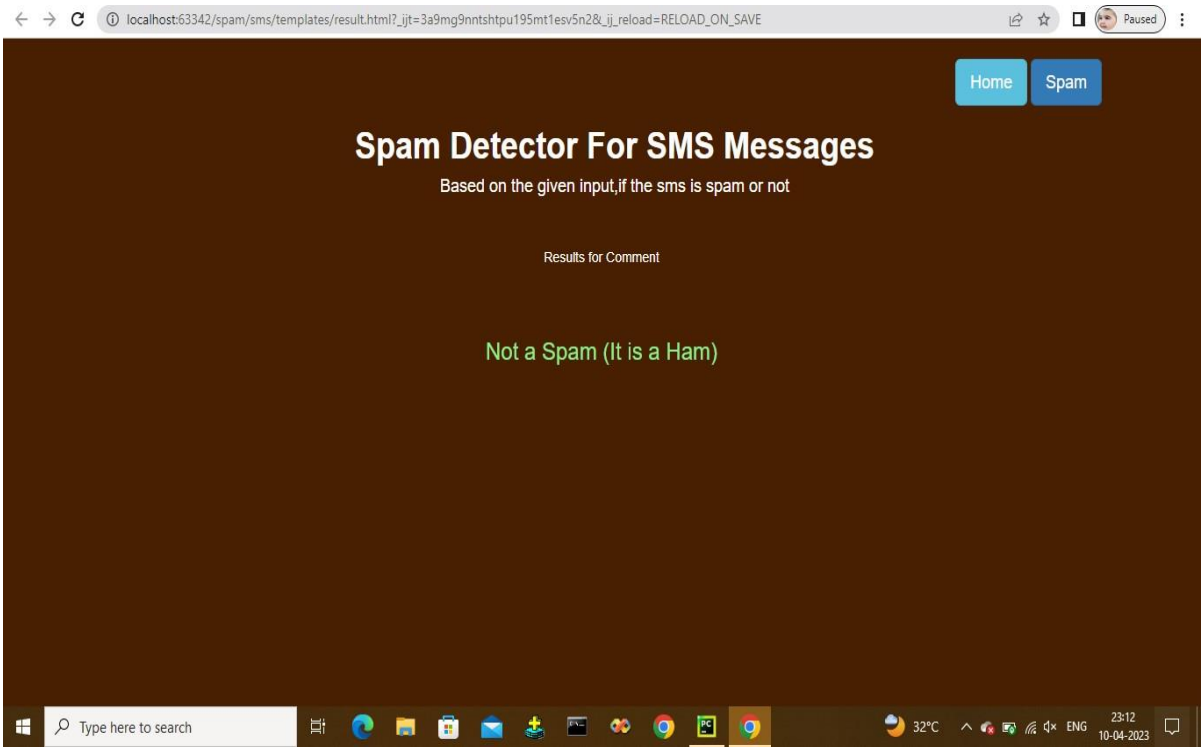

Our Group ideas
1.SMS spam kaggle data set
2.Google colab for running the coding
3.KNN,ANN,Naive Bayes,Random Forest and Decision Tree models are used
4.Natural Language Pocessing technique
5.Python language is used with Anaconda prompt
WORKSPACES
KAGGLE
GOOGLE COLAB
ANACONDA PROMPT
KNN MODEL
ANN MODEL
NAIVE BAYES MODEL
DECISION TREE MODEL
RANDOM FOREST MODEL
Natural Language Processing
TECHNIQUE
MODELES


Prioritize
20 minutes
Natural language technique
KNN algorithm
Operating System
HTML is used
Google colab for running the coding
ANN Model
kaggle dataset
python
Online libraries in colab
Naive Bayes Model
Decision tree model
Web Framework
Random Forest Model
anaconda prompt
identify the spam message

3. RESULT


localhost:63342/spam/sms/templates/index.html?_ijt=dph3pm5aekcungeob8alkrdelu&_ij_reload=RELOAD_ON_SAVE
Paused
Home
Spam
Spam Filtering
Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry.Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.To tackle this problem 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.
Type here to search
32°C
ENG
23:17
10-04-2023


localhost:63342/spam/sms/templates/spam.html?_ijt=qm2mqku5siek6fhibuqt6385p6&_ij_reload=RELOAD_ON_SAVE
Paused
Home
Spam
Spam Detector For SMS Messages
Machine Learning App with Flask
Enter Your Message Here
hello
predict
Type here to search
32°C
ENG
23:10
10-04-2023


localhost:63342/spam/sms/templates/result.html?_ijt=3a9mg9nntshtpu195mt1esv5n2&_ij_reload=RELOAD_ON_SAVE
Paused
Home
Spam
Spam Detector For SMS Messages
Based on the given input,if the sms is spam or not
Results for Comment
Not a Spam (It is a Ham)
Type here to search
32°C
ENG
23:12
10-04-2023

4. ADVANTAGES AND DISADVANTAGES

Advantages:

- **Accuracy:**

  Machine learning algorithms can learn from large datasets, resulting in higher accuracy rates compared to rule-based filtering methods.

- **Adaptability:**

  Machine learning algorithms can adapt to changes in spamming techniques and new types of spam messages, making them more effective over time.

- **Efficiency:**

  Once a machine learning algorithm is trained, it can process messages in real-time, which is crucial for filtering spam from high-volume sources like SMS and social media.

- **Scalability:**

  Machine learning algorithms can be easily scaled to handle large volumes of messages, making them ideal for use in high-traffic environments.

Disadvantages:

- **Data requirements:**

  Machine learning algorithms require large amounts of training data to perform effectively, which can be difficult to obtain, particularly for new or rapidly evolving spamming techniques.

- **Overfitting:**

  Machine learning algorithms can sometimes overfit the training data, leading to poor generalization and reduced accuracy on new data.

5. APPLICATIONS

- **Personalization:**

  Machine learning can be used to personalize the filtering of spam messages based on the recipient's preferences, behavior, and interests. This can enhance the user experience and increase the accuracy of filtering.

- **Fraud detection:**

  Machine learning can be used to detect fraudulent SMS messages such as phishing scams or fake promotional offers.

- **Sentiment analysis:**

  Machine learning can be used to analyze the sentiment of SMS messages, which can help filter out messages that contain negative or inappropriate content.

- **Language translation:**

  Machine learning can be used to automatically translate SMS messages from one language to another, which can be useful for international messaging.

- **Automated responses:**

  Machine learning can be used to generate automated responses to SMS messages based on the content of the message.

- **Analytics:**

  Machine learning can be used to analyze SMS data to gain insights into customer behavior, preferences, and trends. This can be useful for businesses to improve their marketing strategies and customer engagement.

6. CONCLUSION

Machine learning can greatly improve spam filtering by accurately identifying and categorizing spam SMS. By using a combination of supervised and unsupervised learning techniques, such as natural language processing and deep learning, machine learning algorithms can effectively identify and filter out spam SMS in real-time.

However, it is important to note that no spam filtering system is perfect, and some spam SMS may still slip through the cracks. Therefore, it is important to continually update and refine the machine learning algorithms to keep up with new spamming techniques.

Additionally, privacy concerns must be taken into consideration when implementing machine learning-based spam filtering systems. It is important to ensure that user data is protected and not misused, while still providing an effective spam filtering service. Machine learning can be a powerful tool in the fight against spam, but it must be used responsibly and ethically.

7. FUTURE SCOPE

The field of spam filtering with machine learning has already made significant progress in recent years, but there is still a lot of potential for future developments. Some possible directions for optimizing spam filtering with machine learning in the future include:

- **Deep learning approaches:**

  While traditional machine learning techniques like decision trees, naive Bayes, and SVMs have been effective in spam filtering, deep learning methods like neural networks, convolutional neural networks, and recurrent neural networks have shown great potential for improving accuracy and handling complex data types. As the field of deep learning continues to advance, these techniques will likely become even more powerful for spam filtering.

- **Incorporating more data sources:**

  Currently, most spam filters rely primarily on analyzing the content of sms to determine whether they are spam or not. However, there are many other sources of data that could be used to improve spam filtering accuracy, such as metadata, sender reputation, network traffic patterns, and social media activity. As machine learning models become better at integrating and analyzing these diverse data sources, they will be able to make more informed decisions about whether an email is spam.

- **Adapting to evolving spam tactics:**

  Spammers are constantly coming up with new tactics to try to bypass spam filters, such as using image-based spam or obfuscating text. Machine learning models need to be able to adapt to these changing tactics in order to remain effective. One approach to this is to use reinforcement learning, which allows the model to continually learn and adapt based on feedback from its environment.

- **Interpretable machine learning:**

  As machine learning models become more complex, it becomes more difficult to understand how they are making decisions. This is a problem for spam filtering, where it is important to be able to explain to users why a particular email was classified as spam. Developing interpretable machine learning techniques that allow users to understand the reasoning behind the model's decisions will be an important area of research in the future.

- **Privacy concerns:**

  Spam filtering requires analyzing the content of sms, which raises privacy concerns. One way to address this is to use techniques like differential privacy, which allows data to be analyzed in a way that preserves individual privacy.This systems are designed with privacy in mind will be important as these systems become more widespread

8. APPENDIX

8.1 SOURCE CODE

```
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

import csv

from sklearn import datasets

df=pd.read_csv("/content/sample_data/spam.csv",encoding="latin")

df.head()

df.info()

df.isna().sum()

df.rename({"v1":"label","v2":"text"},inplace=True,axis=1)

df.tail()

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

df['label']=le.fit_transform(df['label'])

dataset = datasets.load_wine()

from sklearn.model_selection import train_test_split

X = dataset['data']

y = dataset['target']

x_train,x_test,y_train,y_test = train_test_split(X,y,test_size = 0.20, random_state = 0)
```

```
print("Before OverSampling, counts of label '1':{}".format(sum(y_train==1)))
print("Before OverSampling, counts of label '0':{} \n".format(sum(y_train==0)))
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.shape))
print("After OverSampling, counts of label '1':{}".format(sum(y_train_res==1)))
print("After OverSampling, counts of label '0':{}".format(sum(y_train_res==0)))
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=35000)
X=cv.fit_transform(corpus).toarray()
import pickle
pickle.dump(cv,open('cv1.pk1','wb'))
df.describe()
df.shape
df["label"].value_counts().plot(kind="bar",figsize=(12,6))
plt.xticks(np.arange(2),('Non spam','spam'),rotation=0);
sc=StandardScaler()
X=sc.fit_transform(X)
x_bal=pd.DataFrame(X_bal,columns=names)
dataset = datasets.load_wine()
from sklearn.model_selection import train_test_split
X = dataset['data']
y = dataset['target']
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size = 0.20, random_state = 0)
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)
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB()
model.fit(x_train_res,y_train_res)
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_initializer="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=1,activation="sigmoid"))
model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accuracy'])
generator=model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch=len(x_train_res)//6)
generator=model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch=len(x_train_res)//6)
y_pred=model.predict(x_test)
y_pred
y_pr=np.where(y_pred>0.5,1,0)
y_test
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
```

```
print('Accurracy Score Is:-' ,score*100)
def new_review(new_review):
 new_review=new_review
 new_review=re.sub('[^a-zA-Z]',' ',new_review)
 new_review=new_review.lower()
 new_review=new_review.split()
 ps=PorterStemmer()
 all_stopwords=stopwords.words('english')
 all_stopwords.remove('not')
 new_review=[ps.stem(word) for word in new_review if not word in set(all_stopwords)]
 new_review='  '.join(new_review)
 new_corpus=[new_review]
 new_x_test=cv.transform(new_corpus).toarray()
 print(new_x_test)
 new_y_pred=loaded_model.predict(new_x_test)
 print( new_y_pred)
 new_x_pred=np.where(new_y_pred>0.5,1,0)
 return new_y_pred
new_review=new_review(str(input("Enter new review...")))
import numpy as np # scientific computation
import pandas as pd # loading dataset file
import matplotlib.pyplot as plt # visualization
import nltk # preprocessing our text
from nltk.corpus import stopwords # removing all the stop words
```

```
from nltk.stem.porter import PorterStemmer # stemming of words
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy score is naive bayes|:-' ,score*100)
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is:-',score*100)
cm1=confusion_matrix(y_test,y_pred1)
score1= accuracy_score(y_test, y_pred1)
print(cm1)
print('accuracy score is:-' ,score*100)
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
print('accuracy score is:-',score*100)
model.save('spam.h5')
```

PYCHARM CODING

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>Home</title>
<linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<style>
Body
{
background: linear-gradient(109.6deg,rgba(209,0,116,1)11.2%,rgba(110,44,107,1)91.1%);
background-size: cover;
}
h3.big
{
line-height: 1.8;
color: white;
}
 </style>
</head>
```

```
<body>

<br>

<div class="container">

<div class="row">

<div class="col-md-12 bg-light text-right">

<a href="/home" class="btn btn-info btn-lg">Home</a>

<a href="/spam" class="btn btn-primary btn-lg">Spam</a>

</div>

</div>

<center>

 <h1><strong> Spam Filtering </strong></h1>

 </center>

 <h3 class="big"><em>
```

Over recent years, as the popularity of mobile phone devices has increased, Short Message.Service (SMS) has grown into a multi-billion dollar industry.Over recent years, as the popularity of mobile phone devices has increased, Short Message.Service (SMS) has grown into a multi-billion dollar industry. So Spam SMS is one of the major issues in the wireless communication world and it growsday by day.To tackle this problem 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.

```
</em></h3><br>

</div>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

</body>

</html>
```

```
</em></h3>

</div>

</body>

</html>
```

Spam.html

```
<!DOCTYPE html>

<html>

<head>

<title>Home</title>

<linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<style>

body

{

background: linear-gradient(109.6deg,rgba(209,0,116,1)11.2%,rgba(110,44,107,1)91.1%);

background-size: cover;

}

</style>

</head>

<body>

<div class="container">

<div class="row">

<div class="col-md-12 bg-light text-right">

<a href="/home" class="btn btn-info btn-lg">Home</a>
```

```
<a href="/spam" class="btn btn-primary btn-lg">Spam</a>
</div>
</div>
<header>
<div class="container">
<div id="brandname">
Machine Learning App with Flask
</div>
<h2>Spam Detector For SMS Messages</h2>
</div>
</header>
<div class="ml-container">
<form action="{{ url_for('predict')}}" method="POST">
<p>Enter Your Message Here</p>
<!-- <input type="text" name="comment"/> -->
<textarea name="message" rows="4" cols="50"></textarea>
<br/>
<input type="submit" class="btn-info" value="predict">
</form>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</body>
</html>
```

result.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Output</title>
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<style>
Body
{
background: linear-gradient(109.6deg,rgba(209,0,116,1)11.2%,rgba(110,44,107,1)91.1%);
background-size: cover;
}
 h3.big
{
 line-height: 1.8;
 color: white;
 }
</style>
</head>
<body>
<br>
 <div class="container">
<div class="row">
```

```
<div class="col-md-12 bg-light text-right">
<a href="/home" class="btn btn-info btn-lg">Home</a>
<a href="/spam" class="btn btn-primary btn-lg">Spam</a>
</div>
</div>
<header>
<div class="container">
<div id="brandname">
ML App
</div>
<h2>Spam Detector For SMS Messages</h2>
</div>
</header>
<p style="color:blue;font-size:20;text-align: center;"><b>Results for Comment</b></p>
<div class="results">
{% if prediction == 1%}
<h2 style="color:red;">Spam</h2>
{% elif prediction == 0%}
<h2 style="color:blue;">Not a Spam (It is a Ham)</h2>
{% endif %}
</div>
</body>
</html>
```

Python Coding

```
from flask import Flask,render_template,request
import pickle
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from tensorflow.keras.models import load_model

loaded_model = load_model('spam.h5')
cv = pickle.load(open('cv1.pk1','rb'))
app = Flask(__name__)
@app.route('/')
def home():
 return render_template('home.html')
@app.route('/spam',methods=['POST','GET'])
def prediction():
 return render_template('spam.html')
@app.route('/predict',methods=['POST'])
def predict():
 if request.method == 'POST':
  message = request.form['message']
  data = message
```

```
new_review = str(data)
 print(new_review)
 new_review = re.sub('[a-zA-Z]',' ', new_review)
 new_review = new_review.lower()
 new_review = new_review.split()
 ps = PorterStemmer()
 all_stopwords = stopwards.words('english')
 all_stopwords.remove('not')
 new_review = [ps.stem(word) for word in new_review if not word in set(all_stopwords)]
 new_review = ' '.join(new_review)
 new_corpus = [new_review]
 new_x_test = cv.transform(new.corpus).toarray()
 print(new_x_test)
 new_y_pred = loaded_model.predict(new_x_test)
 new_x_pred = np.where(new_y_pred>0.5,1,0)
 print(new_x_pred)
 if new_review[0][0]==1:
  return render_template('result.html',prediction="Spam")
 else:
  return render_template('result.html',prediction="Not a spam")
if__name__=="__main__"
  #app.run(host='0.0.0.0',port=8000,debug=True)
  port=int(os.environ.get('PORT',5000))
  app.run(debug=False)
```