Project Report on

SPAM IDENTIFICATION

SUBMITTED BY

ABHISHEK SHUKLA (47)

ARYAN THESIA (67)

ANIL YADAV (55)

GAUTAM SHUKLA (14)

UNDER GUIDANCE OF

MS. APARNA MAJARE

DEPARTMENT OF

ELECTRONICS AND TELECOMMUNICATION ENGINEERING

SHREE L. R. TIWARI COLLEGE OF ENGINEERING,

KANAKIA PARK, MIRA ROAD (E) – 401 107

UNIVERSITY OF MUMBAI

Academic Year 2023–2024

Department of Electronics and Telecommunication Engineering

Shree L R Tiwari College of Engineering,

Mira Road (E) – 401 107.

CERTIFICATE

This is to certify that the project entitled **"SPAM IDENTIFICATION"** is a bonafide work of the following students

ABHISHEK SHUKLA [47]

ARYAN THESIA [67]

ANIL YADAV [55]

GAUTAM SHUKLA [14]

submitted to the department of **Electronics and Telecommunication** in partial fulfillment of the requirement for the award of the internship **certificate.**

(Mr./Mrs.)
Mentor

_____

Mrs. Aboli Moharil
HOD

_____

SHREE L.R. TIWARI
College of Engineering

INDEX

INTRODUCTION

ABSTRACT

We delve deep into modern content-based e-mail spam filtering methods, focusing on machine learning-driven filters and their adaptations. Our discussion covers key concepts, methods, notable advancements, and recent innovations in this field. Initial analysis reveals the basics of e-mail spam filtering and the role of feature engineering. We conclude by exploring techniques, methodologies, evaluation criteria, and insights from recent progress, paving the way for future research.
Keywords: SVM Classifier, Spam Email Classification, Data Mining, Machine Learning, Deep Learning, etc.

GENERAL OVERVIEW

In present times the commercial or bulk e-mails have become a really major problem. Spam nowadays is a waste of storage space, time and bandwidth for communication. From many years the problem caused by spam or fraud mails is increasing. In recent studies, 77% of all mail is spam that comes around a value of 15 billion emails per day and costs Internet users about $ 300 million per year. Today for email filtering, knowledge Engineering and Machine Learning are two most successful approaches. In knowledge engineering approach the hard and fast rule is specifying a set of principles according to which email is classified as spam or ham. Application of this method, doesn't shows any promising results because the rules should be necessary. Constantly updating the rules and methods just causes waste of time and requires more maintenance. As compared to knowledge Engineering, Machine learning is more appropriate approach. It does not have to specify any rules. A set of pre-classified e-mail messages is used here in place of set of rules. Machine learning approaches have a wide range of Importance and a lot of algorithms can be used for e-mail filtering and classification. These include Support Vector Machine, Naïve Bayes.

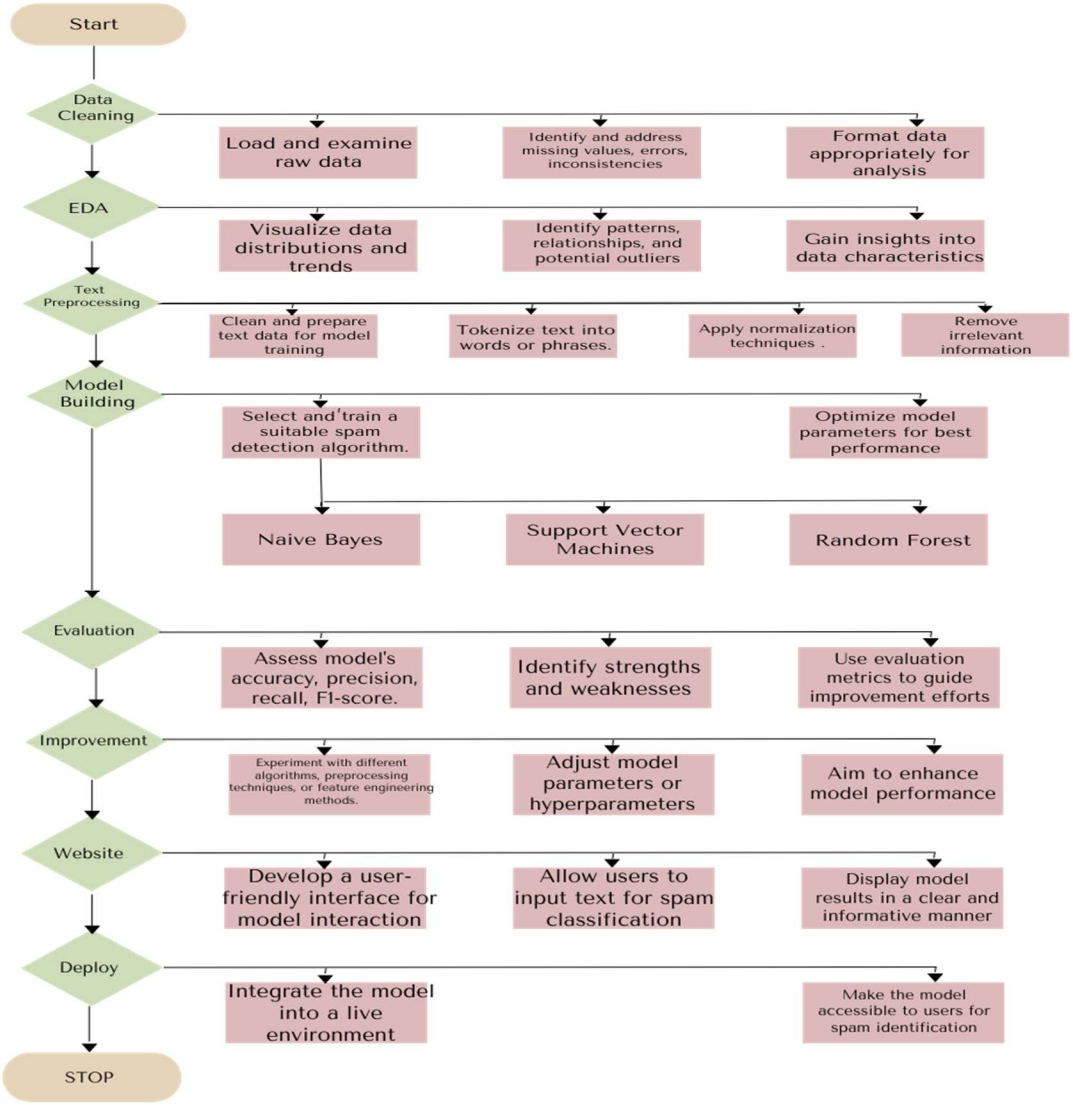
LITERATURE REVIEW

1.Inuwa-Dutse et al. (2018) proposed a real-time spam detection system for Twitter, using a combination of ML classifiers, such as Support Vector Machine (SVM), Random Forest (RF), Multi-Layer Perception (MLP), Gradient Boosting, and Maximum Entropy. They used a Honeypot dataset, as well as a manually and automatically annotated spam dataset (SPD) to evaluate their system. They achieved an accuracy of 97.71%, a precision of 99%, a recall of 97%, and an F-score of 98%. However, they noted that their system faced limitations with dealing with lengthy tweets, which could affect the spamming activity detection.

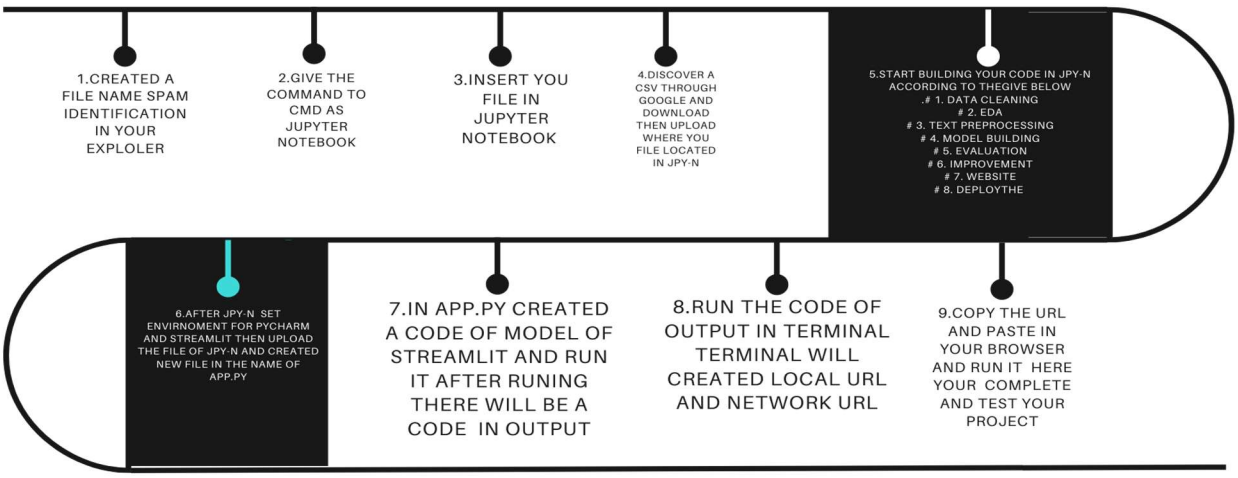
2.Aiyar and Shetty (2018) applied ML models, such as SVM, RF, and Naive Bayes (NB) to detect hate speech in YouTube comments. They used N-grams based features to represent the comments, and obtained an F1-score of 0.97. They suggested that better word representation techniques, such as word embeddings, could improve the performance of their system.

3.Alharthi et al. (2018) worked on sentiment analysis of Arabic tweets, using Long Short-Term Memory (LSTM) models. They collected over 10,000 tweets via the Twitter API, and annotated them manually with positive, negative, or neutral labels. They achieved an accuracy of 0.97, but noted that the system classification depended on the tweet length, and that shorter tweets were more difficult to classify.

FLOWCHART



TIMELINE



SOFTWARE USED

1.Command Prompt (CMD): CMD is a command-line interpreter in Windows OS, used for executing commands and scripts. It can be used to navigate files, run certain programs, or perform basic tasks related to spam filtering configurations or management.

2.Jupyter Notebook: Jupyter Notebook is an open-source web application that allows creating and sharing documents containing live code, equations, visualizations, and narrative text. It's used in spam identification for developing and testing machine learning models or algorithms.

3.Visual Studio Code: VS Code is a lightweight but powerful source-code editor with support for various programming languages. It's used in spam identification for writing, editing, and debugging code related to spam filtering algorithms or scripts.

4.Python 3: Python 3 is a widely used high-level programming language known for its simplicity and readability. In spam identification, Python 3 is used for developing machine learning models, data analysis, and scripting tasks for spam detection systems.

5.PyCharm: PyCharm is an integrated development environment (IDE) for Python development. It provides tools for coding assistance, debugging, and intelligent code analysis, aiding developers working on spam identification algorithms or scripts.

6.Streamlit: Streamlit is an open-source Python library used for building web applications for data science and machine learning projects. In spam identification, Streamlit can be utilized to create user-friendly interfaces for displaying spam identification results or interacting with spam detection models.

CODE AND FUNCTION

1. For training the algorithm dataset from Kaggle is used which is shown below

```
[1]: import numpy as np
     import pandas as pd

[2]: df = pd.read_csv('spam.csv',encoding='latin1')

[3]: df.sample(5)
```

[3]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
2939	spam	You have 1 new message. Please call 08712400200.	NaN	NaN	NaN
1759	ham	Do u ever get a song stuck in your head for no...	NaN	NaN	NaN
1304	ham	Your right! I'll make the appointment right now.	NaN	NaN	NaN
1072	spam	Dear U've been invited to XCHAT. This is our f...	NaN	NaN	NaN
165	ham	I place all ur points on e cultures module alr...	NaN	NaN	NaN

```
[4]: df.shape
[4]: (5572, 5)
```

- A Python code snippet and its output, displaying a sample of data from a CSV file labeled as spam or ham.
- The code imports numpy and pandas libraries, reads the spam.csv file, and prints the shape and a random sample of the DataFrame.

2.DATA CLEANING:-

```
[6]: # drop last 3 cols
     df.drop(columns=['Unnamed: 2','Unnamed: 3','Unnamed: 4'],inplace=True)
[7]: df.sample(5)
```

[7]:

	v1	v2
4102	spam	U have a secret admirer who is looking 2 make ...
812	spam	Congratulations ur awarded either â£500 of CD ...
3896	ham	No. Thank you. You've been wonderful
5480	ham	Have you seen who's back at Holby?!
4777	ham	U R THE MOST BEAUTIFUL GIRL IVE EVER SEEN. U R...

```
[8]: # renaming the cols
     df.rename(columns={'v1':'target','v2':'text'},inplace=True)
     df.sample(5)
```

[8]:

	target	text
1442	ham	Its ok., i just askd did u knw tht no?
5497	spam	SMS SERVICES. for your inclusive text credits,...
2963	spam	Do you ever notice that when you're driving, a...
1842	ham	\Are you comingdown later?\""
3699	ham	Oh...i asked for fun. Haha...take care. Ì_

Python code and output for displaying text messages labeled as ham or spam.

```
[9]: from sklearn.preprocessing import LabelEncoder
     encoder = LabelEncoder()
[10]: df['target'] = encoder.fit_transform(df['target'])
[11]: df.head()
```

[11]:

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

```
[12]: # missing values
      df.isnull().sum()
[12]: target    0
      text      0
      dtype: int64
[13]: # check for duplicate values
      df.duplicated().sum()
[13]: 403
[14]: # remove duplicates
      df = df.drop_duplicates(keep='first')
[15]: df.duplicated().sum()
[15]: 0
[16]: df.shape
[16]: (5169, 2)
```

1.Python code and output for using LabelEncoder to transform text data.
2. Python code for handling missing and duplicate values in a DataFrame.

3.EDA

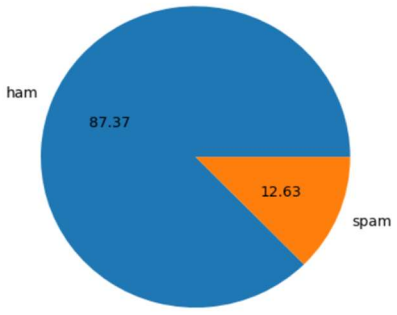
```
[17]: df.head()
```

[17]:

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

```
[18]: df['target'].value_counts()
[18]: target
      0    4516
      1     653
      Name: count, dtype: int64
```

```
[19]: import matplotlib.pyplot as plt
      plt.pie(df['target'].value_counts(), labels=['ham','spam'],autopct="%0.2f")
      plt.show()
```



1.Python code and its output, displaying the use of LabelEncoder to transform text data.
2.A pie chart generated using Python, displaying the distribution of ham and spam messages.

```
[24]: df['num_characters'] = df['text'].apply(len)
[25]: df.head()
```

[25]:

	target	text	num_characters
0	0	Go until jurong point, crazy.. Available only ...	111
1	0	Ok lar... Joking wif u oni...	29
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155
3	0	U dun say so early hor... U c already then say...	49
4	0	Nah I don't think he goes to usf, he lives aro...	61

```
[26]: # num of words
      df['num_words'] = df['text'].apply(lambda x:len(nltk.word_tokenize(x)))
[27]: df.head()
```

[27]:

	target	text	num_characters	num_words
0	0	Go until jurong point, crazy.. Available only ...	111	24
1	0	Ok lar... Joking wif u oni...	29	8
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37
3	0	U dun say so early hor... U c already then say...	49	13
4	0	Nah I don't think he goes to usf, he lives aro...	61	15

```
[28]: df['num_sentences'] = df['text'].apply(lambda x:len(nltk.sent_tokenize(x)))
[29]: df.head()
```

[29]:

	target	text	num_characters	num_words	num_sentences
0	0	Go until jurong point, crazy.. Available only ...	111	24	2
1	0	Ok lar... Joking wif u oni...	29	8	2
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2
3	0	U dun say so early hor... U c already then say...	49	13	1
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1

1.Python code and its output displaying a DataFrame with text messages and their character counts.

2. Python code and its output, displaying the use of LabelEncoder to transform text data.

3. Python code and a DataFrame displaying text messages and their respective character and word counts.

```
[30]: df[['num_characters','num_words','num_sentences']].describe()
```

[30]:

	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.977945	18.455794	1.965564
std	58.236293	13.324758	1.448541
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	60.000000	15.000000	1.000000
75%	117.000000	26.000000	2.000000
max	910.000000	220.000000	38.000000

```
[31]: # ham
      df[df['target'] == 0][['num_characters','num_words','num_sentences']].describe()
```

[31]:

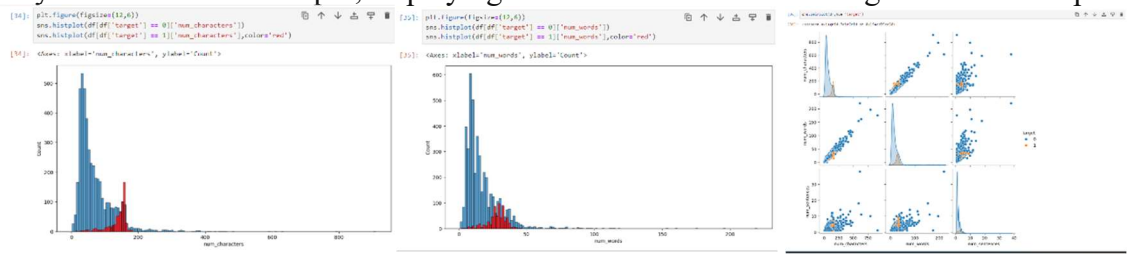
	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.459256	17.123782	1.820195
std	56.358207	13.493970	1.383657
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

```
[32]: #spam
      df[df['target'] == 1][['num_characters','num_words','num_sentences']].describe()
```

[32]:

	num_characters	num_words	num_sentences
count	653.000000	653.000000	653.000000
mean	137.891271	27.667688	2.970904
std	30.137753	7.008418	1.488425
min	13.000000	2.000000	1.000000
25%	132.000000	25.000000	2.000000
50%	149.000000	29.000000	3.000000
75%	157.000000	32.000000	4.000000
max	224.000000	46.000000	9.000000

Python code and its output, displaying statistical data of text messages labeled as “spam”.



1.Python code output showing a histogram of word counts for two groups of texts, with blue bars for target 0 and red bars for target 1.

2.Python code output showing a correlation heatmap of four numeric variables, with annotations and a color scale.

3.A pairplot graph showing the relationship between the number of characters, words, and sentences in two groups of texts.

4. DATA PREPROCESSING:-

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

ps = PorterStemmer()
stop_words = set(stopwords.words('english'))

def transform_text(text):
    text = text.lower()
    text = nltk.word_tokenize(text)

    y = []
    for i in text:
        if i.isalnum():
            y.append(i)

    text = y[:]
    y.clear()

    for i in text:
        if i not in stop_words and i not in string.punctuation:
            y.append(ps.stem(i))

    return " ".join(y)

# Example usage
transformed_text = transform_text("I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.")
print(transformed_text)
```

```
gon na home soon want talk stuff anymor tonight k cri enough today
```

```
transformed_text= ("I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.")
df['text'][10]
```

```
"I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today."
```

```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
ps.stem('loving')
```

```
'love'
```

```
df['transformed_text'] = df['text'].apply(transform_text)
```

```
df.head()
```

	target	text	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

```
from wordcloud import WordCloud
wc = WordCloud(width=500,height=500,min_font_size=10,background_color='white')
```

```
spam_wc = wc.generate(df[df['target'] == 1]['transformed_text'].str.cat(sep=" "))
```

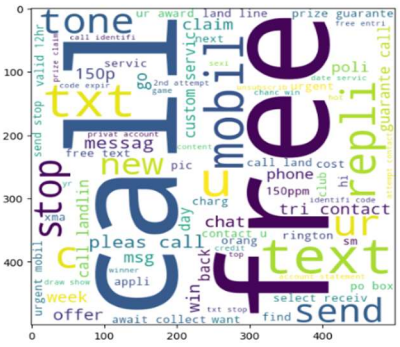
```
plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

```
<matplotlib.image.AxesImage at 0x1f4f03d44d0>
```

Python code for text processing and visualization

```
plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

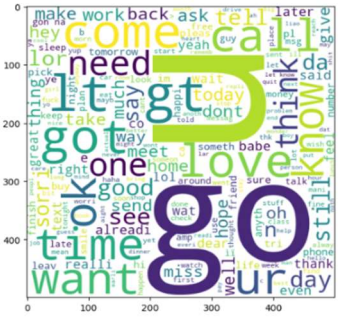
```
<matplotlib.image.AxesImage at 0x1f4f03d44d0>
```




```
ham_wc = wc.generate(df[df['target'] == 0]['transformed_text'].str.cat(sep=" "))
```

```
plt.figure(figsize=(15,6))
plt.imshow(ham_wc)
```

```
<matplotlib.image.AxesImage at 0x1f4f03f2660>
```



Word cloud of terms related to mobile texting or messaging.

```
df.head()
```

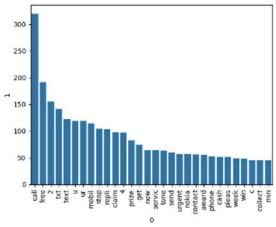
	target	text	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

```
spam_corpus = []
for msg in df[df['target'] == 1]['transformed_text'].tolist():
    for word in msg.split():
        spam_corpus.append(word)
```

```
len(spam_corpus)
```

```
9939
```

```
import seaborn as sns
from collections import Counter
sns.barplot(x=pd.DataFrame(Counter(spam_corpus).most_common(30))[0], y=pd.DataFrame(Counter(spam_corpus).most_common(30))[1])
plt.xticks(rotation='vertical')
plt.show()
```

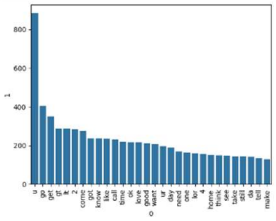


```
ham_corpus = []
for msg in df[df['target'] == 0]['transformed_text'].tolist():
    for word in msg.split():
        ham_corpus.append(word)
```

```
len(ham_corpus)
```

```
35404
```

```
from collections import Counter
sns.barplot(x=pd.DataFrame(Counter(ham_corpus).most_common(30))[0], y=pd.DataFrame(Counter(ham_corpus).most_common(30))[1])
plt.xticks(rotation='vertical')
plt.show()
```



```
# Text Vectorization
# using Bag of Words
df.head()
```

	target	text	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

A data table, and a bar graph visualizing the frequency of words.

5. MODEL BUILDING

```
[57]: from sklearn.feature_extraction.text import TfidfVectorizer
      from sklearn.model_selection import train_test_split
      from sklearn.naive_bayes import MultinomialNB
      from sklearn.metrics import accuracy_score, precision_score
      import pandas as pd
      import seaborn as sns
      import matplotlib.pyplot as plt
      import pickle

[58]: # Assuming you have a DataFrame 'df' with columns 'transformed_text' and 'target'
      tfidf = TfidfVectorizer(max_features=3000)
      X = tfidf.fit_transform(df['transformed_text']).toarray()

[59]: X = tfidf.fit_transform(df['transformed_text']).toarray()

[60]: y = df['target'].values

[61]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)

[62]: mnb = MultinomialNB()
      mnb.fit(X_train, y_train)

[62]: ▾ MultinomialNB
      MultinomialNB()

[63]: # Save the model using pickle
      with open('model.pkl', 'wb') as model_file:
          pickle.dump(mnb, model_file)

[64]: # Later, when you want to use the model to make predictions:
      # Load the saved model
      with open('model.pkl', 'rb') as model_file:
          mnb = pickle.load(model_file)
```

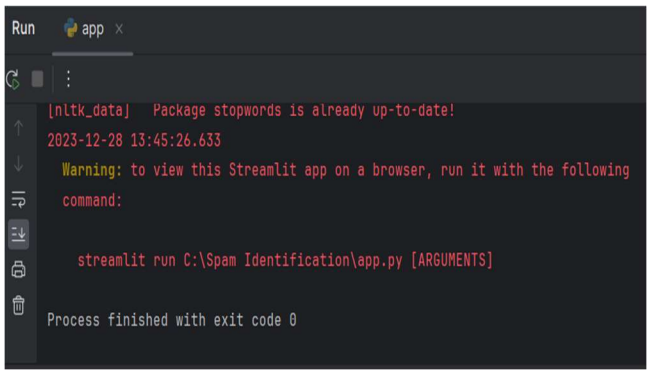
```
# Later, when you want to use the model to make predictions:

[65]: # Load the saved model
      with open('model.pkl', 'rb') as model_file:
          mnb = pickle.load(model_file)
      if isinstance(mnb, MultinomialNB):
          y_pred = mnb.predict(X_test)
          accuracy = accuracy_score(y_test, y_pred)
          precision = precision_score(y_test, y_pred)
          print("Accuracy:", accuracy)
          print("Precision:", precision)
      else:
          print("Error: Model is not a MultinomialNB instance.")

      Accuracy: 0.9709864603481625
      Precision: 1.0

[66]: # Similarly, you can do the same for the vectorizer
      with open('vectorizer.pkl', 'wb') as vectorizer_file:
          pickle.dump(tfidf, vectorizer_file)

[67]: # Later, to load the vectorizer:
      with open('vectorizer.pkl', 'rb') as vectorizer_file:
          tfidf = pickle.load(vectorizer_file)
```

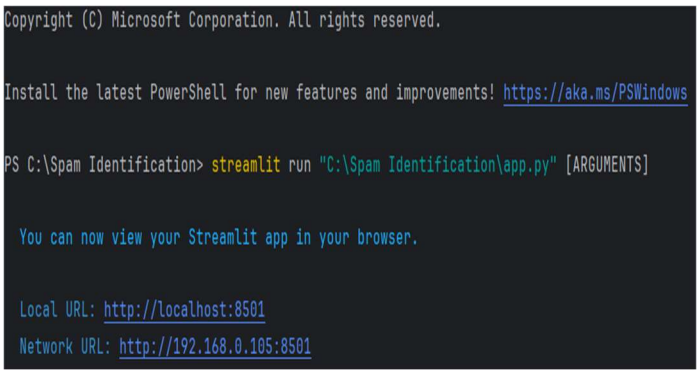
training and saving.

2.Python code for loading a saved model and vectorizer to make predictions.

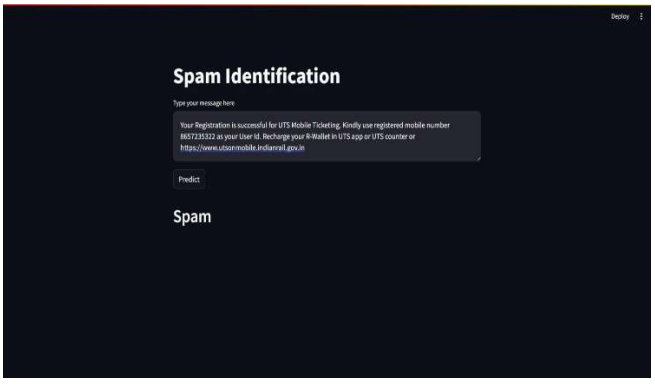
OUTPUT AND RESULT

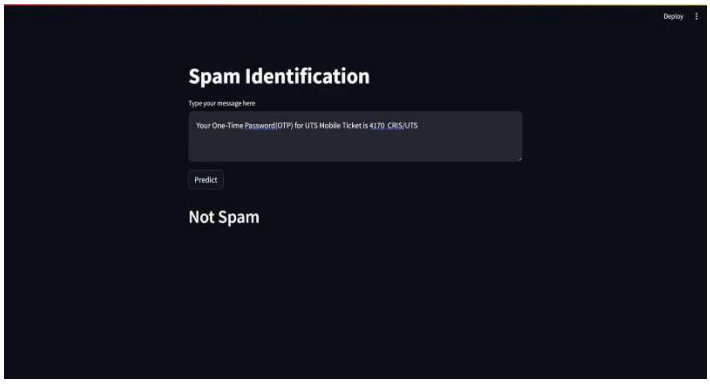
1.URL AND STREAMLIT RUN





2.TESTING





ADVANTAGE & DISADVANTAGE

ADVANTAGE:-

1.Effectively filters unwanted emails
2.Enhances cybersecurity
3.Protects personal information
4.Reduces malware and phishing threats
5.Increases user productivity

DISADVANTAGE:-

1.Possibility of false positives
2.Evolving tactics of spammers
3.Over-filtering legitimate emails
4.Resource-intensive for large-scale operations
5.Language and cultural barriers

FUTURE SCOPE

1.Integration of AI for smarter filtering
2.Improved anomaly detection algorithms
3.Real-time adaptation to new spamming techniques
4.Development of anti-spoofing techniques
5.Behavioral analysis for personalized filtering

CONCLUSION

Email has been the most important medium of communication nowadays, through internet connectivity any message can be delivered to all aver the world. More than 270 billion emails are exchanged daily, about 57% of these are just spam emails. Spam emails, also known as non-self, are undesired commercial or malicious emails, which affects or hacks personal information like bank ,related to money or anything that causes destruction to single individual or a corporation or a group of people. Besides advertising, these may contain links to phishing or malware hosting websites set up to steal confidential information. Spam is a serious issue that is not just annoying to the end-users but also financially damaging and a security risk. Hence this system is designed in such a way that it detects unsolicited and unwanted emails and prevents them hence helping in reducing the spam message which would be of great benefit to individuals as well as to the company .In the future this system can be implemented by using different algorithms and also more features can be added to the existing syste

REFERENCE

1. Shukor Bin Abd Razak, Ahmad Fahrulrazie Bin Mohamad “Identification of Spam Email Based on Information from Email Header” 13th International Conference on Intelligent Systems Design and Applications (ISDA), 2013
2. Trivedi, Shrawan Kumar. "A study of machine learning classifiers for spam detection." Computational and Business Intelligence (ISCBI), 2016 4th International Symposium on. IEEE, 2016. [10] You, Wanqing, et al. "Web Service-Enabled Spam Filtering with Naïve Bayes Classification." 2015 IEEE First International Conference on Big Data Computing Service and Applications (BigDataService). IEEE, 2015.
3. Sahın, Esra, Murat Aydos, and Fatih Orhan. "Spam/ham e-mail classification using machine learning methods based on bag of words technique." 2018 26th Signal Processing and Communications Applications Conference (SIU). IEEE, 2018
4. Mujtaba, Ghulam, et al. "Email classification research trends: Review and open issues." IEEE Access 5 (2017).
5. Ravinder Kamboj, “A rule based approach for spam detection” ,Computer Science and Engineering Department, Thapar University, India, July 2010.