

CHAPTER 1

INTRODUCTION:

The Internet has become a common thing in our lives. The same message sends multiple times which affects the organization financially and also irritates the receiving user. In this project, a Spam Mail Detection system is proposed will classify the given email as spam or ham email. Spam filtering mainly focuses on the content of the message. The classification algorithm classifies the given email based on the content. Feature extraction and selection plays a vital role in the classification. In spam mail detection, email data is collected through the dataset.

To obtain the accurate results, data needs to be pre-processed by removing stop words and word tokenization. Pre-processing of data is done by using TF-IDF Vectorizer module. SVM algorithm is used to detect the given email is spam or harm. In recent times, unwanted industrial bulk emails known as spam has become an enormous drawback on the net. The person causing the spam messages is noted because the sender. Such an individual gathers email addresses from completely different websites, chatrooms, and viruses. Spam prevents the user from creating full and sensible use of your time, storage capability and network information measure. The massive volume of spam mails flowing through the pc networks have damaging effects on the memory house of email servers, communication information measure, central processing unit power and user time. The menace of spam email is on the rise on yearly basis and is to blame for over seventy-seven of the entire international email traffic.

Users United Nations agency receive spam emails that they failed to request realize it terribly irritating. It's conjointly resulted to much loss to several users United Nations agency have fallen victim of web scams and different dishonest practices of spammers United Nations agency send emails pretence to be from honorable firms with the intention to influence people to disclose sensitive personal info like passwords, Bank Verification variety (BVN) and mastercard numbers.

CHAPTER 2

RELATED WORK:

To effectively handle the threat expose by email spams, leading email suppliers like Gmail, Yahoo mail and Outlook have utilized the mixture of various machine learning (ML) techniques like Neural Networks in its spam filters. These techniques have the capacity to be told and establish spam mails and phishing messages by analyzing many such messages throughout a massive assortment of computers. Since machine learning have the capability to adapt to variable conditions, Gmail and Yahoo mail spam filters do over simply checking junk emails victimization pre-existing rules. They generate new rules themselves supported what they need to learn as they continue in their spam filtering operation.

The machine learning model utilized by Google have currently advanced to the purpose that it will observe and separate spam and phishing emails with regarding 99% accuracy. The implication of this is often that one out of m messages reach evading their email spam filter. Statistics from Google discovered that between 50-70 % of emails that Gmail receives area unit direct mail. Google's detection models have conjointly incorporated tools referred to as Google Safe Browsing for distinctive websites that have malicious URLs.

The phishing-detection performance of Google are increased by introduction of a system that delay the delivery of some Gmail messages for a short while to hold out further comprehensive scrutiny of the phishing messages since easier to observe after they are analyzed and put together. The aim of delaying the delivery of a number of these suspicious emails is to conduct a deeper examination whereas a lot of messages arrive in due course of your time and therefore the algorithms are updated in real time. solely regarding zero.05 % of emails are unit plagued by this deliberate delay.

CHAPTER 3

METHODOLOGY:

The era of creation of this product includes models i.e., object-oriented model, Prototype model, waterfall model etc. for making the correct system. water model, the oldest model of creation of correct system. The product model used by our framework is that the cascade model.

Cascade model could be a precise and successive way to contend with the merchandise improvement. This incorporates framework coming up with and displaying that sets up requirements for all the framework parts and distribution some set of those conditions to programming. Framework building and examination incorporate requirement gathering at the framework level with modest amount of top-ranking arrange. Examination info building consolidate would like assortment at the key business level and at the business space level.

CHAPTER 4

EXISTING SYSTEM:

- Email Spam Classifier based on Machine Learning Techniques had done by using SVM, KNN, Naive Bayes and Decision tree algorithms etc.
- SVM had an average accuracy of 99.6%.
- It had good accuracy when compared to the other algorithms in proposed system.

CHAPTER 5

PROPOSED SYSTEM:

- Email Spam Classifier is used to classify email data into spam and ham emails.
- This method is performed by using Support Vector Machine (SVM) algorithm.
- In this method, dataset is divided into two sets based on labels and given as input to algorithm.
- The accuracy of 99% on training data and 98.2% on test data is obtained through the proposed system.

CHAPTER 6

PROBLEM DEFINATION:

6.1 How Gmail, Yahoo and Outlook emails spam filters work?

Different spam filtering formulas are used by Gmail, Outlook.com and Yahoo Mail to deliver solely the valid emails to their users and strain the illegitimate messages. Conversely, these filters additionally typically mistakenly block authentic messages. it's been according that concerning twenty p.c of authorization based mostly emails sometimes fail to urge to the inbox of the expected recipient. the e-mail suppliers have designed varied mechanisms to be used in email anti-spam filter to curtail the risks posed by phishing, email-borne malware and ransomware to email users. The mechanisms area unit wont to decide the danger level of every incoming email. samples of such mechanisms embody satisfactory spam limits, sender policy frameworks, whitelists and blacklists, and recipient verification tools. These mechanisms may be utilized by single or multiple users. once the satisfactory spam thresholds is simply too low it will cause a lot

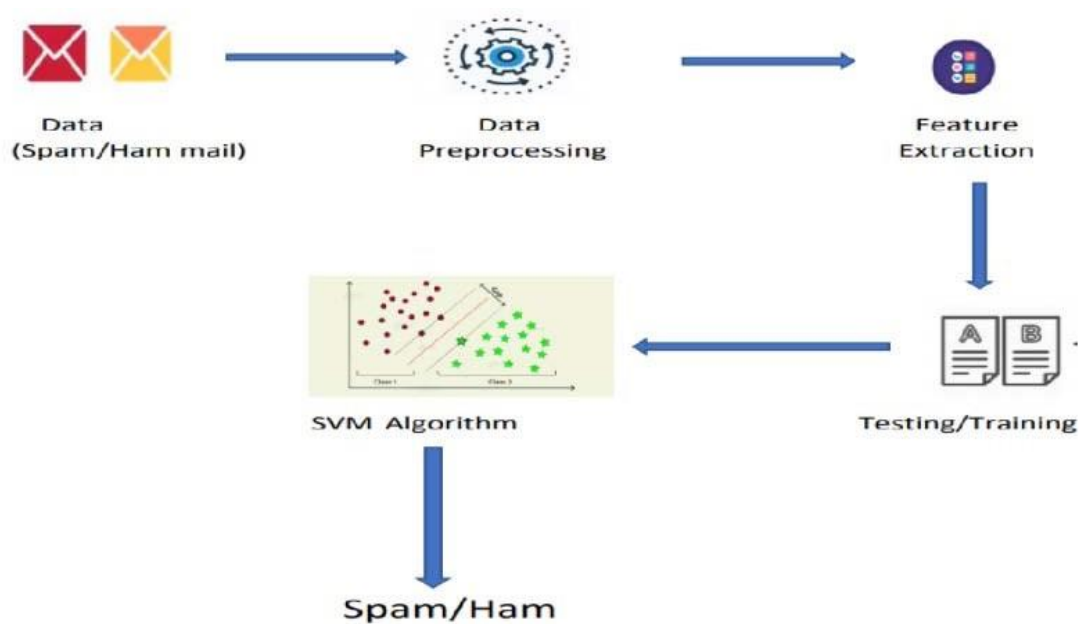
of spam evading the spam filter and getting into the users' inboxes. in the meantime having a awfully high threshold will cause some vital emails being isolated unless the administrator redirects them. This section discusses the operations of Gmail, Yahoo and Outlook emails anti-spam filters.

6.2 Email spam filtering method:

An email message is created from 2 major elements that area unit the header and also the body. The header is that the space that have broad data concerning the content of the e-mail. It includes the topic, sender and receiver. The body is that the heart of the e-mail. It will embody data that doesn't have a pre-defined information. Examples embody website, audio, video, analog information, images, files, and hypertext mark-up language markup. the email header is comprised of fields like sender's address, the recipient's address, or timestamp that indicate once the message was sent by negotiant servers to the Message Transport Agents (MTAs) that operate as associate workplace for organising mails. The header line typically starts with a “From” associated it goes through some modification whenever it moves from one server to a different through an mediate server. Headers enable the user to look at the route the e-mail passes through, and also the time taken by every server to treat the mail. The on the market data ought to submit to some process before the classifier will build use of it for filtering.

CHAPTER 7

ARCHITECTURE DIAGRAM



MODULES

- Dataset Collection
- Data Cleaning
- Modules used in the code.
- Preprocessing of dataset.
- Feature Extraction
- Model Training
- Testing Model

Dataset collection:

Informational index assortment: Information assortment can assist you with tracking down ways of following previous occasions utilizing information examination to record them. This permits you to foresee the way and make prescient models utilizing AI devices to anticipate future changes. Since the prescient model is just pretty much as great as the data acquired, the most effective way to gather information is to further develop execution. The data of got to be faultless (garbage, open air squander) and bought to incorporate data about the work you are doing. For instance, a non-performing advance may not profit from the sum got, yet may profit from gas costs over the long run. In this module, we gather data from the Kaggle data set. These figures contain data on yearly contrasts.

Data Cleaning:

Data Cleanliness is a significant piece of all AI exercises. The data cleanliness of this module is excepted for the arrangement of information for the annihilation and transformation of wrong, inadequate, misdirecting data. You can utilize it to look for data. Discover what cleaning you can do.

Feature Extraction:

This is done to lessen the quantity of capacities in the informational index, which will accelerate preparing and increment proficiency. In AI, picture acknowledgment, and picture handling, mining starts at the front line of estimated, useful data (ascribes) pointed toward guaranteeing, adjusting, following, and normalizing data, and now and again prompting more prominent clearness.

Take out the properties related with aspect decrease On the off chance that the calculation's feedback is excessively enormous, it won't be handled, and assuming it is suspected to be excessively huge (like estimating one foot and meter, or rehashing the picture displayed in pixels), it tends to be switched. properties (likewise called vector properties). Characterize the initial segment, called highlight choice. The chose things ought to contain data about the data got so they can fill the ideal role utilizing this portrayal rather than complete data.

Model training: An illustration of this preparation is the informational collection used to prepare the ML calculation. It comprises of significant info definitions that influence information inspecting and yield. The preparation model is utilized to utilize the information through the result and result change calculations. The aftereffects of this connection will be utilized to alter the layout. This strategy for assault is designated "matching model". Information preparing definition or informational collection approval is significant for demonstrating. Plan language preparing is a method for giving data about the ML calculation and assist with deciding and become familiar with the best significance of every one of its highlights. There are many kinds of AI, the majority of which are controlled and uncontrolled.

Testing Model:

In this module, we test an AI machine planned utilizing research information Quality protection is needed to make the product framework work appropriately. All chance settled upon? Does the program fill in true to form? All program testing standards should be remembered for the specialized detail. What's more, programming testing can uncover every one of the defects and shortcomings that have happened during improvement. Once the application is delivered, you don't need your clients to come to your home together. Various kinds of tests just take into account recognition of blunders during activity.

CHAPTER 9

FUTURE SCOPE:

However, varied endeavors are actualized towards grappling the problem of spam SMS utilizing body, social and innovative measures, the arrangement planned aren't finished arrangements. 1) Achieving precise grouping, with zero % (0%) misclassification of Ham SMS as spam and spam SMS as Ham. 2) The endeavors would be applied to stand phishing SMS that conveys the phishing assaults and now-days that is more and more matter of concern. The framework we tend to area unit making are going to be operating simply on windows. As increasing utilization of cellular phone step by step, there's a requirement to recollect this workplace as AN application for cellphone too.

Tools:



Installation and Software Requirements:

In this project, we have used Anaconda which has various inbuilt software's like Spyder, PyCharm, Jupyter, and much more...

For this specific project, I had used Jupyter notebook to run the codes

Anaconda is a distribution of the Python programming languages for scientific computing, that aims to simplify package management and deployment.

Languages Used:



In this project, I have used python with machine learning Python is an interpreted high-level general-purpose programming language.

Python is an interactive and object-oriented scripting language. Python is designed to be highly readable. It supports functional and structured programming methods as well as OOP. It can be used as a scripting language or can be compiled to bytecode for building large applications. It provides very high-level dynamic data types and supports dynamic type checking. It supports automatic garbage collection.

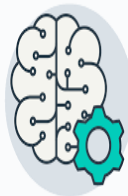
About Machine Learning!!

Machine Learning:



Machine Learning is an application of Artificial Intelligence (AI) which enables a program/software to learn from the experiences and improve itself at a task without being explicitly programmed. For example, how would you write a program that can identify fruits based on their various properties, such as color, shape, size, or any other Machine Learning today has all the attention it needs. Machine Learning can automate many tasks, especially the

ones that only humans can perform with their innate intelligence. Replicating this intelligence to machines can be achieved only with the help of machine learning.



Unsupervised Learning:

Unsupervised learning is a machine learning technique in which models are not supervised using a training dataset. Instead, models themselves find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things. Unsupervised learning cannot be directly applied to a regression or classification problem because, unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of the dataset, group that data according to similarities, and represent that dataset in a compressed format.

Jupyter Final Project Last Checkpoint: 12/20/2021 (autosaved) Python 3

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

In [2]: #Loading the data from csv file to pandas dataframe
df = pd.read_csv('spamham.csv')

In [3]: df.sample(5)

Out[3]:
```

	Category	Message
531	spam	PRIVATE! Your 2003 Account Statement for 07815...
4145	ham	That's a shame! Maybe cld meet for few hrs tomo?
5405	ham	So how many days since then?
4610	ham	Y de asking like this.
329	ham	Cool, text me when you're parked

```
In [4]: #Checking no. of rows and columns in dataframe
df.shape

Out[4]: (5572, 2)

In [5]: # 1. Data cleaning
# 2. EDA
# 3. Text Preprocessing
# 4. Model building
# 5. Evaluation
# 6. Improvement
# 7. Website
# 8. Deploy
```

1.Data Cleaning

CODE

```
In [6]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   Category    5572 non-null   object
1   Message     5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB

In [7]: from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()

In [8]: df['Category'] = encoder.fit_transform(df['Category'])

In [9]: #printing first 5 rows of dataframe
df.head()

Out[9]:
```

	Category	Message
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...

```
In [10]: # missing values
df.isnull().sum()

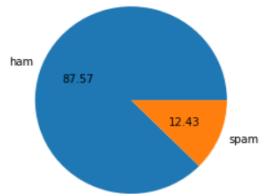
Out[10]: Category    0
Message          0
dtype: int64
```

SCREENSHOTS

```
In [15]: df['Category'].value_counts()
```

```
Out[15]: 0    4516  
         1     641  
         Name: Category, dtype: int64
```

```
In [16]: import matplotlib.pyplot as plt  
plt.pie(df['Category'].value_counts(), labels=['ham', 'spam'], autopct="%0.2f")  
plt.show()
```



```
In [17]: import nltk
```

```
In [18]: !pip install nltk
```

```
Requirement already satisfied: nltk in c:\users\siva rama krishna\anaconda3\lib\site-packages (3.5)  
Requirement already satisfied: click in c:\users\siva rama krishna\anaconda3\lib\site-packages (from nltk) (7.1.2)  
Requirement already satisfied: joblib in c:\users\siva rama krishna\anaconda3\lib\site-packages (from nltk) (0.17.0)  
Requirement already satisfied: tqdm in c:\users\siva rama krishna\anaconda3\lib\site-packages (from nltk) (4.50.2)  
Requirement already satisfied: regex in c:\users\siva rama krishna\anaconda3\lib\site-packages (from nltk) (2020.10.15)
```

```
In [19]: nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to C:\Users\Siva Rama  
[nltk_data] Krishna\AppData\Roaming\nltk_data...  
[nltk_data] Package punkt is already up-to-date!
```

```
Out[19]: True
```

```
In [23]: df[['num_characters', 'num_words', 'num_sentences']].describe()
```

```
Out[23]:
```

	num_characters	num_words	num_sentences
count	5157.000000	5157.000000	5157.000000
mean	79.103936	18.560016	1.965290
std	58.382922	13.403671	1.439549
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	61.000000	15.000000	1.000000
75%	118.000000	26.000000	2.000000
max	910.000000	220.000000	38.000000

```
In [24]: #Label Encoding  
#Label spam mail as 0; Non-spam mail as 1;  
df.loc[df['Category'] == 'spam', 'Category', ] = 0  
df.loc[df['Category'] == 'ham', 'Category', ] = 1
```

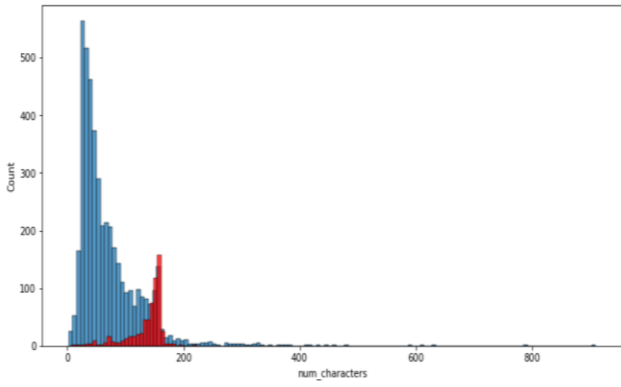
```
In [25]: # ham  
df[df['Category'] == 0][['num_characters', 'num_words', 'num_sentences']].describe()
```

```
Out[25]:
```

	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.869353	17.267272	1.822852
std	56.708301	13.585433	1.374848
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	53.000000	13.000000	1.000000
75%	91.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

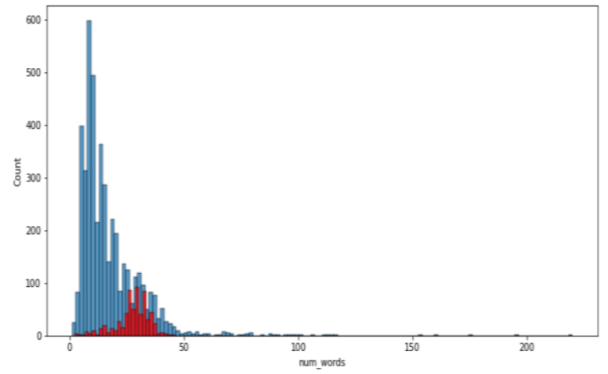
```
In [28]: #num_characters vs count
plt.figure(figsize=(12,6))
sns.histplot(df[df['Category'] == 0]['num_characters'])#spam
sns.histplot(df[df['Category'] == 1]['num_characters'],color='red')#ham
```

Out[28]: <AxesSubplot:xlabel='num_characters', ylabel='Count'>



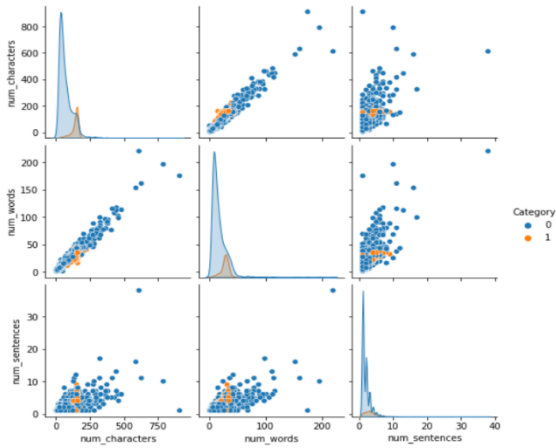
```
In [29]: #num_words vs Count
plt.figure(figsize=(12,6))
sns.histplot(df[df['Category'] == 0]['num_words'])
sns.histplot(df[df['Category'] == 1]['num_words'],color='red')
```

Out[29]: <AxesSubplot:xlabel='num_words', ylabel='Count'>



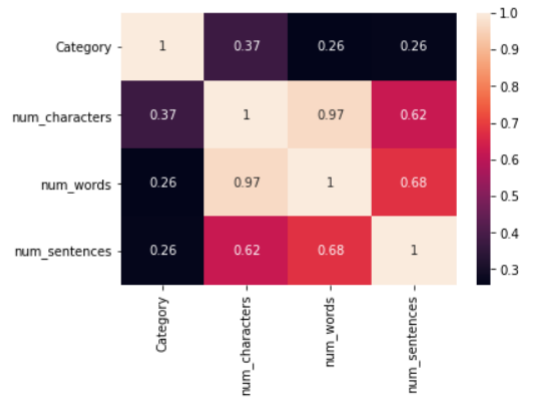
```
In [30]: sns.pairplot(df,hue='Category')
```

Out[30]: <seaborn.axisgrid.PairGrid at 0x28843c6bb50>



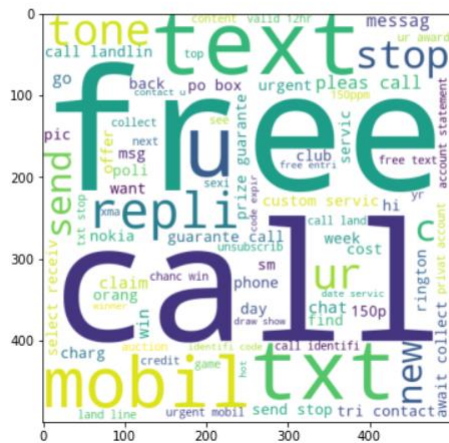
```
In [31]: sns.heatmap(df.corr(),annot=True)
```

Out[31]: <AxesSubplot:~>



```
In [44]: plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

Out[44]: <matplotlib.image.AxesImage at 0x28845c96970>



```
In [46]: plt.figure(figsize=(15,6))
plt.imshow(ham_wc)
```

Out[46]: <matplotlib.image.AxesImage at 0x2884618d8e0>




```
In [48]: spam_corpus = []
for msg in df[df['Category'] == 1]['transformed_Message'].tolist():
    for word in msg.split():
        spam_corpus.append(word)
```

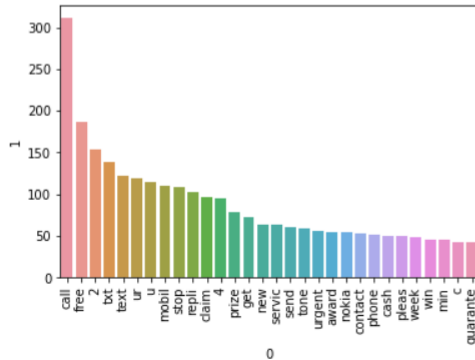
```
In [49]: len(spam_corpus)
```

```
Out[49]: 9781
```

```
In [50]: from collections import Counter
sns.barplot(pd.DataFrame(Counter(spam_corpus).most_common(30))[0],pd.DataFrame(Counter(spam_corpus).most_common(30))[1])
plt.xticks(rotation='vertical')
plt.show()
```

C:\Users\Siva Rama Krishna\anaconda3\lib\site-packages\seaborn_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

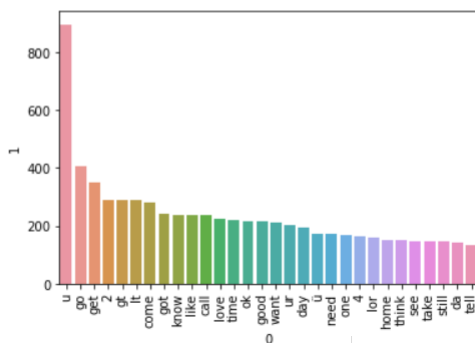
warnings.warn()



```
In [53]: from collections import Counter
sns.barplot(pd.DataFrame(Counter(ham_corpus).most_common(30))[0],pd.DataFrame(Counter(ham_corpus).most_common(30))[1])
plt.xticks(rotation='vertical')
plt.show()
```

C:\Users\Siva Rama Krishna\anaconda3\lib\site-packages\seaborn_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

warnings.warn()



4. Model Building

```
In [55]: from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
cv = CountVectorizer()
tfidf = TfidfVectorizer(max_features=3000)
```

```
In [56]: X = tfidf.fit_transform(df['transformed_Message']).toarray()
```

```
In [57]: X.shape
```

```
Out[57]: (5157, 3000)
```

```
In [58]: y = df['Category'].values
```

```
In [59]: from sklearn.model_selection import train_test_split
```

```
In [60]: X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)
```

```
In [61]: from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score
gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()
```

```
In [62]: gnb.fit(X_train,y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test,y_pred1))
print(confusion_matrix(y_test,y_pred1))
```

```
In [62]: gnb.fit(X_train,y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test,y_pred1))
print(confusion_matrix(y_test,y_pred1))
print(precision_score(y_test,y_pred1))
```

```
0.8672480620155039
[[785 120]
 [ 17 110]]
0.4782608695652174
```

```
In [63]: mnb.fit(X_train,y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred2))
print(confusion_matrix(y_test,y_pred2))
print(precision_score(y_test,y_pred2))
```

```
0.9709302325581395
[[905  0]
 [ 30 97]]
1.0
```

```
In [64]: bnb.fit(X_train,y_train)
y_pred3 = bnb.predict(X_test)
print(accuracy_score(y_test,y_pred3))
print(confusion_matrix(y_test,y_pred3))
print(precision_score(y_test,y_pred3))
0.9835589941972921
```

```
0.9835271317829457
[[903  2]
 [ 15 112]]
0.9824561403508771
```

Out[64]: 0.9835589941972921

```
In [66]: # tfidf --> MNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
```

```
In [67]: svc = SVC(kernel='sigmoid', gamma=1.0)
knc = KNeighborsClassifier()
mnb = MultinomialNB()
dtc = DecisionTreeClassifier(max_depth=5)
lrc = LogisticRegression(solver='liblinear', penalty='l1')
rfc = RandomForestClassifier(n_estimators=50, random_state=2)
abc = AdaBoostClassifier(n_estimators=50, random_state=2)
bc = BaggingClassifier(n_estimators=50, random_state=2)
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
gbdt = GradientBoostingClassifier(n_estimators=50, random_state=2)
xgb = XGBClassifier(n_estimators=50, random_state=2)
```

```
In [68]: clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB' : mnb,
    'DT' : dtc,
    'LR' : lrc,
    'RF' : rfc,
    'AdaBoost' : abc,
    'BgC' : bc,
    'ETC' : etc,
    'GBDT' : gbdt,
    'xgb' : xgb
}
```

```
In [69]: def train_classifier(clf,X_train,y_train,X_test,y_test):
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    precision = precision_score(y_test,y_pred)

    return accuracy,precision
```

```
In [70]: train_classifier(svc,X_train,y_train,X_test,y_test)
```

Out[70]: (0.9757751937984496, 0.9811320754716981)

```
In [71]: accuracy_scores = []
precision_scores = []

for name,clf in clfs.items():

    current_accuracy,current_precision = train_classifier(clf, X_train,y_train,X_test,y_test)

    print("For ",name)
    print("Accuracy - ",current_accuracy)
    print("Precision - ",current_precision)

    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
For SVC
Accuracy - 0.9757751937984496
Precision - 0.9811320754716981
For KN
Accuracy - 0.9127906976744186
Precision - 1.0
For NB
Accuracy - 0.9709302325581395
Precision - 1.0
For DT
Accuracy - 0.936046511627907
Precision - 0.8210526315789474
For LR
Accuracy - 0.9583333333333334
Precision - 0.9375
```

CONCLUSION

- Spam email is one of the most demanding and troublesome internet issues in today's world of communication and technology.
- It is almost impossible to think about e-mail without considering the issue of spam.
- Spammers by generating spam mails are misusing this communication facility and thus affecting organizations and many email users.
- The machine learning model used by Google have now advanced to the point that it can detect and filter out spam and phishing emails with about 99.9 percent accuracy.
- The implication of this is that one out of a thousand messages succeed in evading their email spam filter.

REFERENCES

- Ahmed Khorsi, "An Overview of Content-based Spam Filtering Techniques", Informatica, vol. 31, no. 3, October 2007, pp 269-277.
- Alistair McDonald, "Spam Assassin: A Practical Guide to Integration and Configuration", 1st Edition, Packet publishers, 2004.
- Ian H. Witten, Eibe Frank, "Data Mining – Practical Machine Learning Tools and Techniques," 2nd Edition, Elsevier, 2005.
- Deepika Mallampati, Nagaratna P. Hegde "A Machine Learning Based Email Spam Classification Framework Model" in IJITEE, ISSN: 2278-3075, Vol.9 Issue.4, Feb 2020.
- Javatpoint, "Machine Learning Tutorial" 2017 <https://www.javatpoint.com/machine-learning>.
- Spam Assassin, "Spam and Ham Dataset", Kaggle, 2018. <https://www.kaggle.com/veleon/ham-and-spam-dataset>
- Apache, "open-source Apache Spam Assassin Dataset", 2019. <https://spamassassin.apache.org/old/publiccorpus/>