FLIP ROBO

A project on Spam Email classification

Submitted by:

Sankalp Mahapatra

Internship-29

ACKNOWLEDGMENT

I would like to express my sincere thanks of gratitude to my SME as well as “Flip Robo Technologies” team for letting me work on “Email spam detection classifier” project. Their suggestions and directions have helped me in the completion of this project successfully. This project also helped me in doing lots of research wherein I came to know about so many new things.

Finally, I would like to thank my family and friends who have helped me with their valuable suggestions and guidance and have been very helpful in various stages of project completion.

TABLE OF CONTENTS:

1. INTRODUCTION

1.1 Business Problem Framing

You were recently hired in a Start-up Company and was asked to build a system to identify spam emails. We will explore and understand the process of classifying Emails as Spam or Not Spam by build Machine Learning and NPL model to detect the HAM and SPAM mails. The model will detect the unsolicited and unwanted emails and thus we can prevent them from creeping into user's inbox and therefore, increase the user Experience.

1.2 CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

As we know how a machine translates language, or how voice assistants respond to questions, or how mail gets automatically classified into spam or not spam, all these tasks are done through Natural Language Processing (NLP), which processes text into useful insights that can be applied to future data. In the field of artificial intelligence, NLP is one of the most complex areas of research due to the fact that text data is contextual. It needs modification to make it machine-interpretable and requires multiple stages of processing for feature extraction.

Classification problems can be broadly split into two categories: binary classification problems, and multi-class classification problems. Binary classification means there are only two possible label classes, e.g. a patient's condition is cancerous or it isn't, or a financial transaction is fraudulent or it is not. Multi-class classification refers to cases where there are more than two label classes. An example of this is classifying the sentiment of a movie review into positive, negative, or neutral.

There are many types of NLP problems, and one of the most common types is the classification of strings. Examples of this include the classification of movies/news articles into different genres and the automated classification of emails into a spam or not spam. We shall be looking into this last example in more detail for this project.

1.3 REVIEW OF LITERATURE

In recent times, unwanted commercial / promotional bulk emails also known as spam has become a huge problem on the internet and for our mail inbox. An individual / organization sending the spam messages are referred to as the spammers. Such a person gathers email addresses from different websites, chatrooms, and other sources to send the mail to bulk audience. Spam prevents the user from making full and good use of time, storage capacity and network bandwidth. The huge volume of spam mails flowing through the computer networks have destructive effects on the memory space of email servers, communication bandwidth, CPU power and user time. The menace of spam email is on the increase on yearly basis and is responsible for over 80% of the whole global email traffic (Source google).

Users who receive spam emails that they did not request find it very irritating. It is also resulted to untold financial loss to many users who have fallen victim of internet scams and other fraudulent practices of spammers who send emails pretending to be from reputable companies with the intention to persuade individuals to disclose sensitive personal information like passwords, Bank Verification Number (BVN) and credit card numbers.

1.4 MOTIVATION FOR THE PROBLEM UNDERTAKEN

Motivation for this project has been undertaken because it is a project which is assigned to me during my internship at Flip Robo Technologies. This project will help Start-up companies to detect and filter the SPAM mails in their Email inbox and therefore, increase the user experience and save their server from unwanted mails, phishing mails or other viruses.

2. ANALYTICAL PROBLEM FRAMING

2.1 MATHEMATICAL/ ANALYTICAL MODELING OF THE PROBLEM

Throughout the project multiple mathematical and analytical models have been used, first we have checked the ratio of spam and ham emails in our dataset. The shape of our data set is 2893 rows and 3 columns. Then we have used regular expressions to clean the message column which contained body of the email. Then we have used TfidfVectorizer, to transforms text to feature vectors that can be used as input to estimator.

```
1 df.info()
```

```
<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
```

2.2 DATA SOURCES AND THEIR FORMATS

The data was provided to us from the FlipRobo Technologies as a part of our Internship assignment. The data was provided in CSV format and there were 3 attributes and 5572 rows in the data set.

```
#Let's load the CSV file

df=pd.read_csv("https://raw.githubusercontent.com/Sankalpmahapatra10/-Email-Spam-Classifier-project/main/spam.csv", encoding
df.head()
```

	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

```
df.tail()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

2.3 DATA PREPROCESSING DONE

After loading all the data we will proceeded with the data pre-processing. Following Steps were followed during data pre-processing:

Removing unwanted attribute from Dataset :

It's quite hard to find whether a mail is a spam or not just by looking at the subject. So we started by replacing the null values.

```
#Let's check if there are null values in our dataset

df.isnull().sum()
```

```
v1              0
v2              0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

```
#Lets delete thee null values in the Unnamed:2,Unnamed: 3 and Unnamed: 4 columns
df.drop(columns=['Unnamed: 2','Unnamed: 3', 'Unnamed: 4'], inplace = True)
```

```
#Let's check the columns in our dataset

df.columns
```

```
Index(['v1', 'v2'], dtype='object')
```

```
#Let's re check that we have sucessfully imputed all the null values

df.isnull().sum()
```

```
v1    0
v2    0
dtype: int64
```

Adding additional attribute :

In order to analyse the data in a better way while doing pre-processing, we have added an attribute ‘Length’ which shows length of the message against it. This was done just to compare the length of text before and after preprocessing and to get idea about the memory optimization.

```
df['num_words'] = df['text'].apply(lambda x : len(nltk.word_tokenize(x)))
```

```
df.head()
```

	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

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

```
df.head()
```

	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

Converting all the messages to lower case:

All messages in the ‘message’ attribute was converted to small case since keeping words in large case does not make sense as same word with small and large case conveys same meaning.

```
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[:]  # Copying list we have to clone it
    y.clear()

    for i in text:
        if i not in stopwords.words('english') and i not in string.punctuation:
            y.append(i)
    text = y[:]
    y.clear()

    for i in text:
        y.append(ps.stem(i))

    return " ".join(y)
```

```
transform_text('Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat
```

```
'go jurong point crazi avail bugi n great world la e buffet cine got amor wat'
```

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

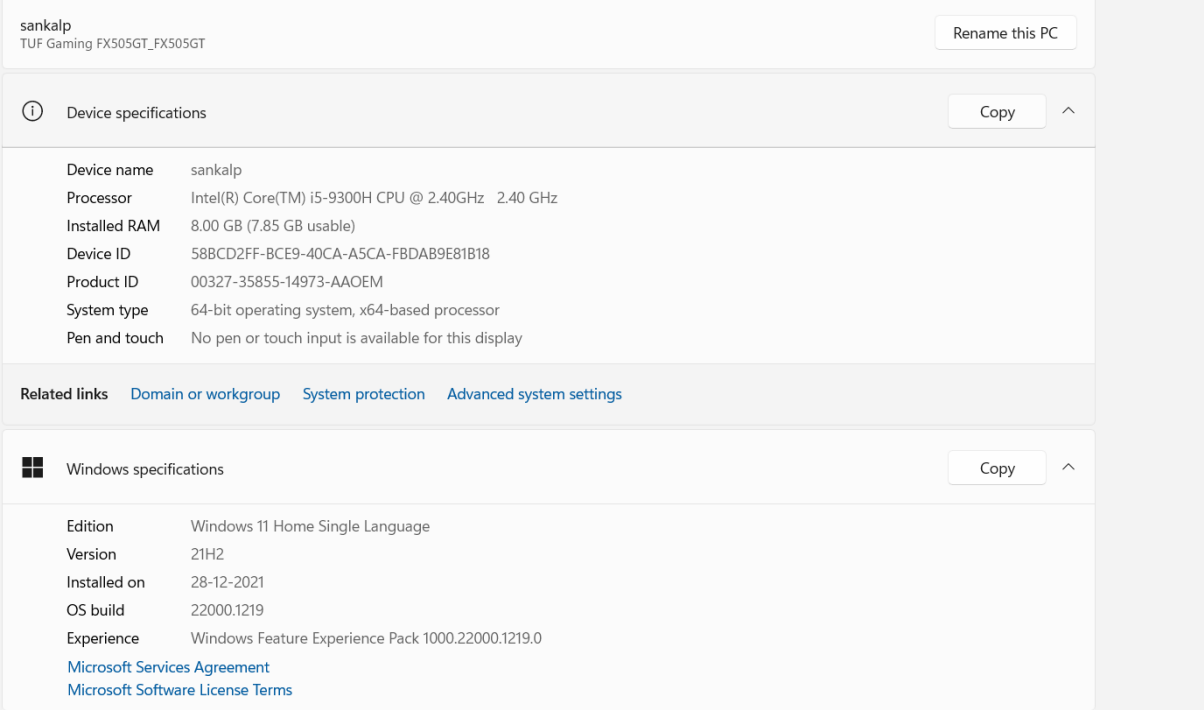
```
df.head()
```

	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

DATA INPUTS- LOGIC- OUTPUT RELATIONSHIPS

We have analysed the words that were present in the spam and ham mails, based on the words present and the data we already have which says if the mail is ham or spam, we are going to train the model to predict the same.

HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED



LIBRARIES:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns



import nltk, re
import string
from nltk.corpus import stopwords
from collections import Counter
from nltk.corpus import stopwords
from wordcloud import WordCloud
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
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
```

MODEL/S DEVELOPMENT AND EVALUATION

IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS)

As the target column was Bivariant data and the algorithm that we choose depends on this target variable. So, we have chosen classification analysis for this project.

TESTING OF IDENTIFIED APPROACHES (ALGORITHMS)

We have used the following algorithms

```
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
```

```
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)
```

RUN AND EVALUATE SELECTED MODELS

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

```
def train_classifier(clf, X_train, y_train):
    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
```

```

train_classifier(svc, X_train, y_train)
```

```
(0.9758220502901354, 0.9747899159663865)
```

```
accuracy_scores = []
precision_scores = []

for name, clf in clfs.items():
    current_accuracy, current_precision = train_classifier(clf, X_train, y_train)
    print("For", name)
    print("Accuracy", current_accuracy)
    print("Precision", current_precision)


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

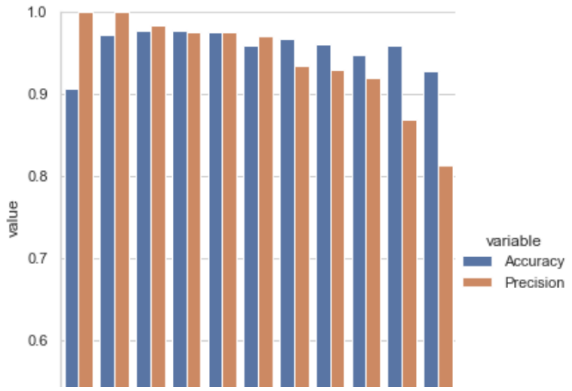
```
For SVC
Accuracy 0.9758220502901354
Precision 0.9747899159663865
For KN
Accuracy 0.9052224371373307
Precision 1.0
For NB
Accuracy 0.9709864603481625
Precision 1.0
For DT
Accuracy 0.9274661508704062
Precision 0.8118811881188119
For LR
Accuracy 0.9584139264990329
Precision 0.9702970297029703
For RF
Accuracy 0.9758220502901354
Precision 0.9829059829059829
For AdaBoost
Accuracy 0.960348162475822
Precision 0.9292035398230089
For BgC
Accuracy 0.9584139264990329
Precision 0.8682170542635659
For ETC
Accuracy 0.9748549323017408
```

```
For ETC
Accuracy 0.9748549323017408
Precision 0.9745762711864406
For GBDT
Accuracy 0.9468085106382979
Precision 0.9191919191919192
For xgb
Accuracy 0.9671179883945842
Precision 0.9333333333333333
```

```
performance_df
```

	Algorithm	Accuracy	Precision
1	KN	0.905222	1.000000
2	NB	0.970986	1.000000
5	RF	0.975822	0.982906
0	SVC	0.975822	0.974790
8	ETC	0.974855	0.974576
4	LR	0.958414	0.970297
10	xgb	0.967118	0.933333
6	AdaBoost	0.960348	0.929204
9	GBDT	0.946809	0.919192
7	BgC	0.958414	0.868217
3	DT	0.927466	0.811881

```
sns.catplot(x = 'Algorithm', y='value',
            hue = 'variable', data=performance_df1, kind = 'bar', height=5)
plt.ylim(0.5,1.0)
plt.xticks(rotation='vertical')
plt.show()
```



Model Evaluation

Model Improvement

```
svc = SVC(kernel='sigmoid', gamma=1.0, probability=True)
mnb = MultinomialNB()
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
from sklearn.ensemble import VotingClassifier
```

```
voting  = VotingClassifier(estimators=[('svm',svc), ('nb', mnb), ('et',etc)], voting='soft') # Weithage is equal for all alg
```

```
voting.fit(X_train, y_train)
```

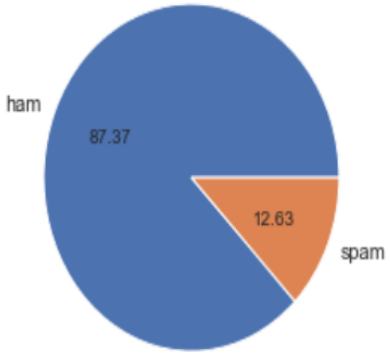
```
VotingClassifier(estimators=[('svm',
                              SVC(gamma=1.0, kernel='sigmoid',
                                  probability=True)),
                             ('nb', MultinomialNB()),
                             ('et',
                              ExtraTreesClassifier(n_estimators=50,
                                                   random_state=2))],
```

```
y_pred = voting.predict(X_test)
print("Accuracy", accuracy_score(y_test, y_pred))
print("Precision", precision_score(y_test,y_pred))
```

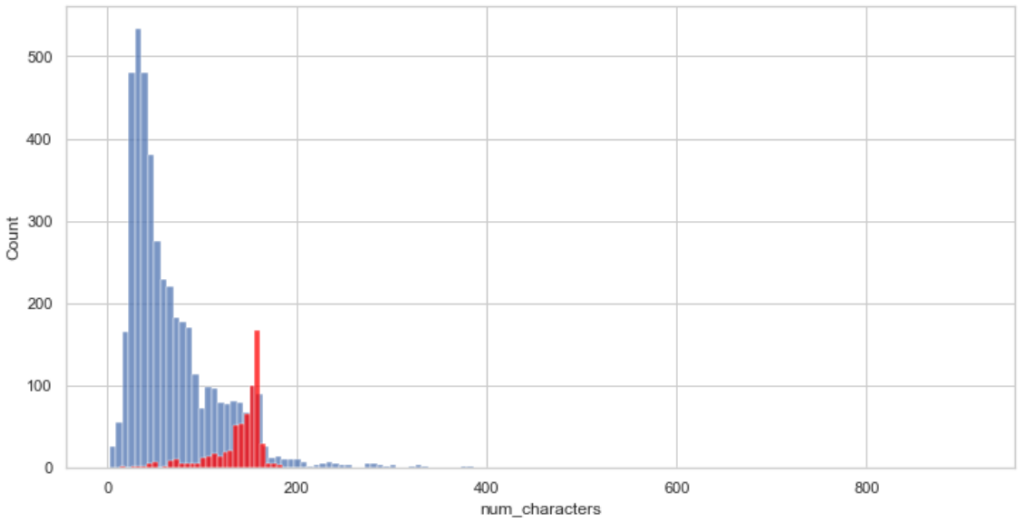
```
Accuracy 0.9816247582205029
Precision 0.9917355371900827
```

Visualizations

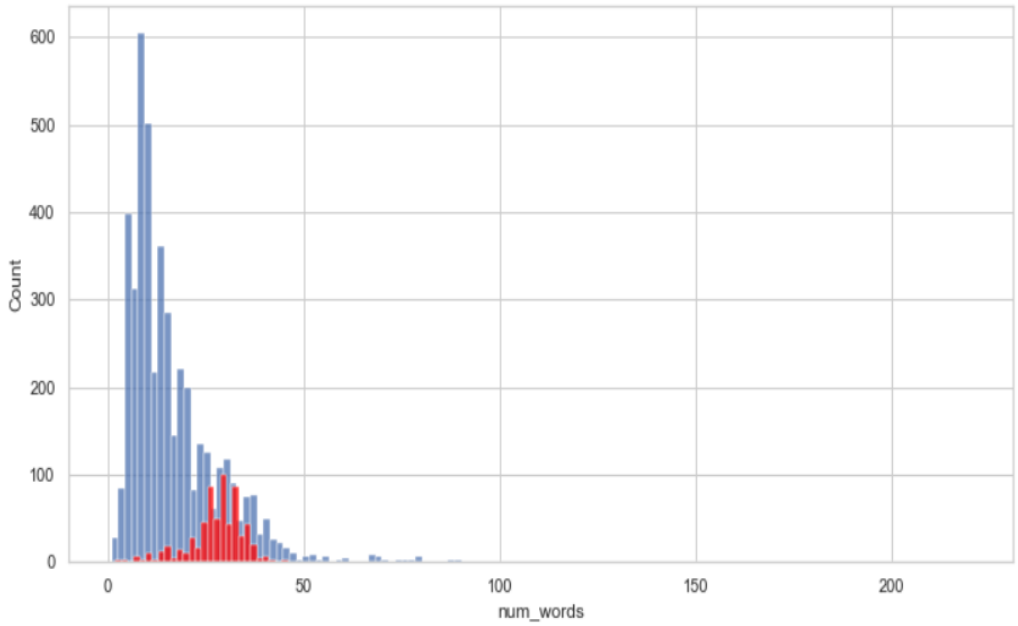
```
plt.pie(df['target'].value_counts(), labels= ['ham', 'spam'], autopct='%0.2f')
plt.show()
```



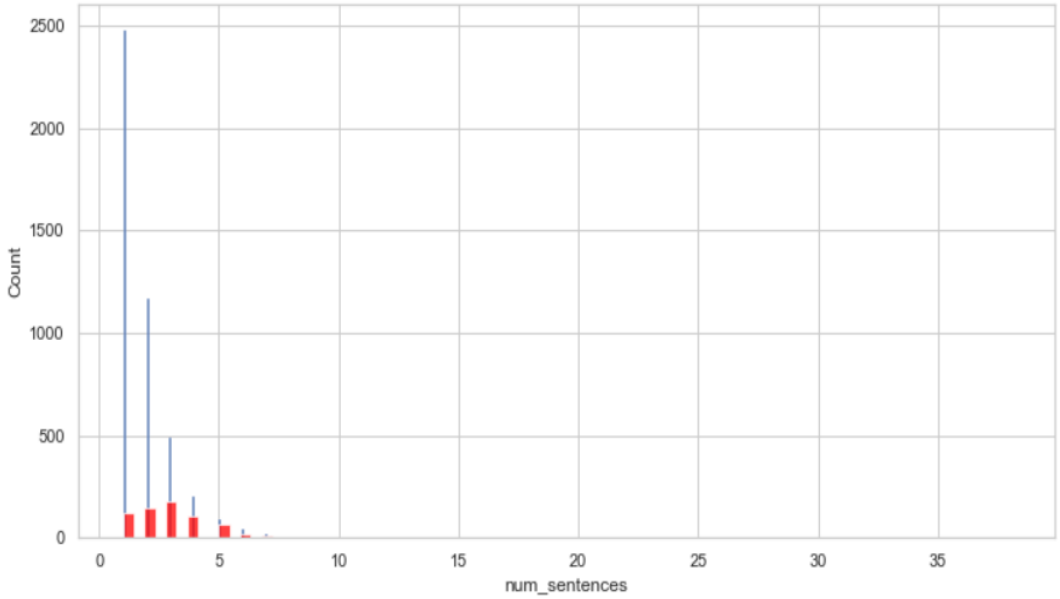

```
plt.figure(figsize =(12,6))
sns.histplot(df[df['target'] == 0]['num_characters']);
sns.histplot(df[df['target'] == 1]['num_characters'], color = 'red');
```



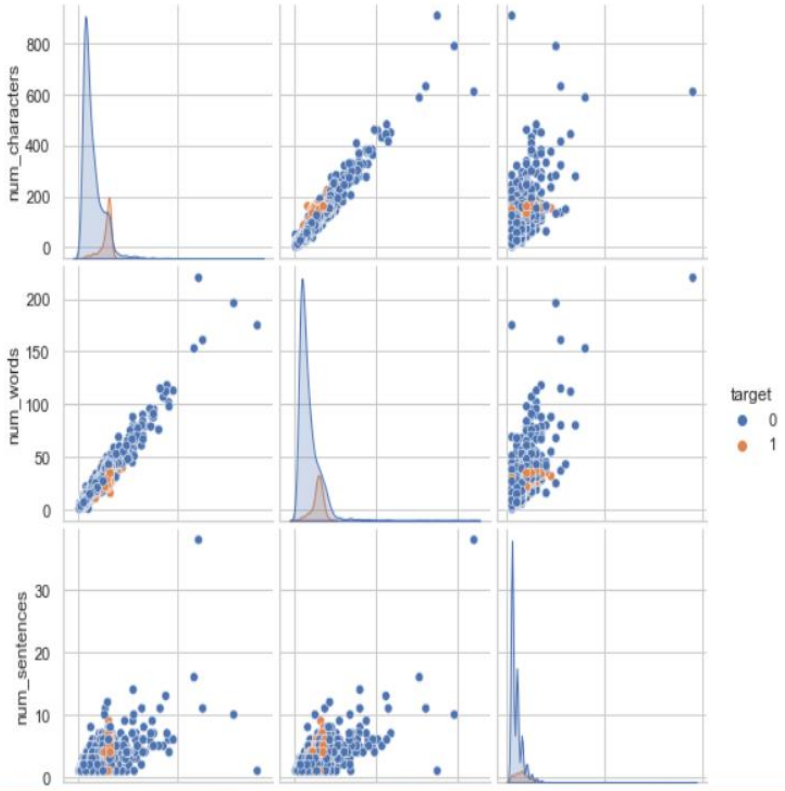
```
plt.figure(figsize =(12,6))
sns.histplot(df[df['target'] == 0]['num_words']);
sns.histplot(df[df['target'] == 1]['num_words'], color = 'red');
```



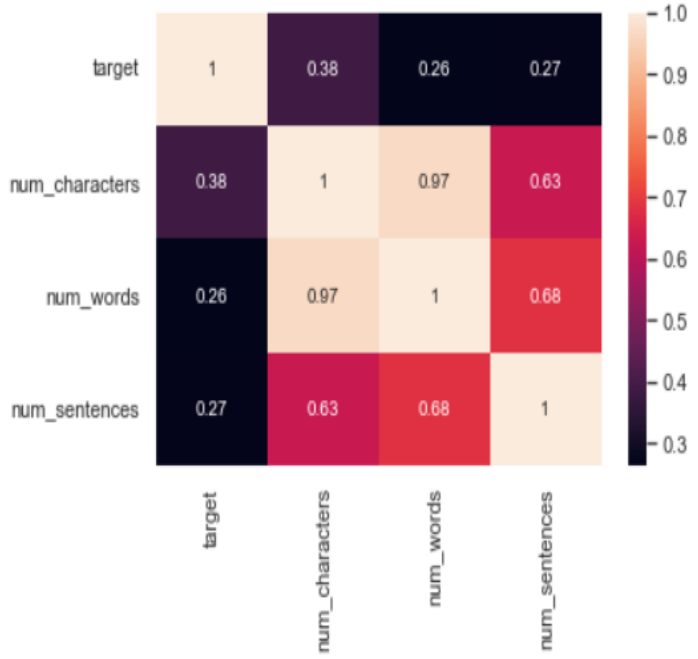

```
plt.figure(figsize =(12,6))
sns.histplot(df[df['target'] == 0]['num_sentences']);
sns.histplot(df[df['target'] == 1]['num_sentences'], color = 'red');
```



```
1 sns.pairplot(df,hue='target');
```




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



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

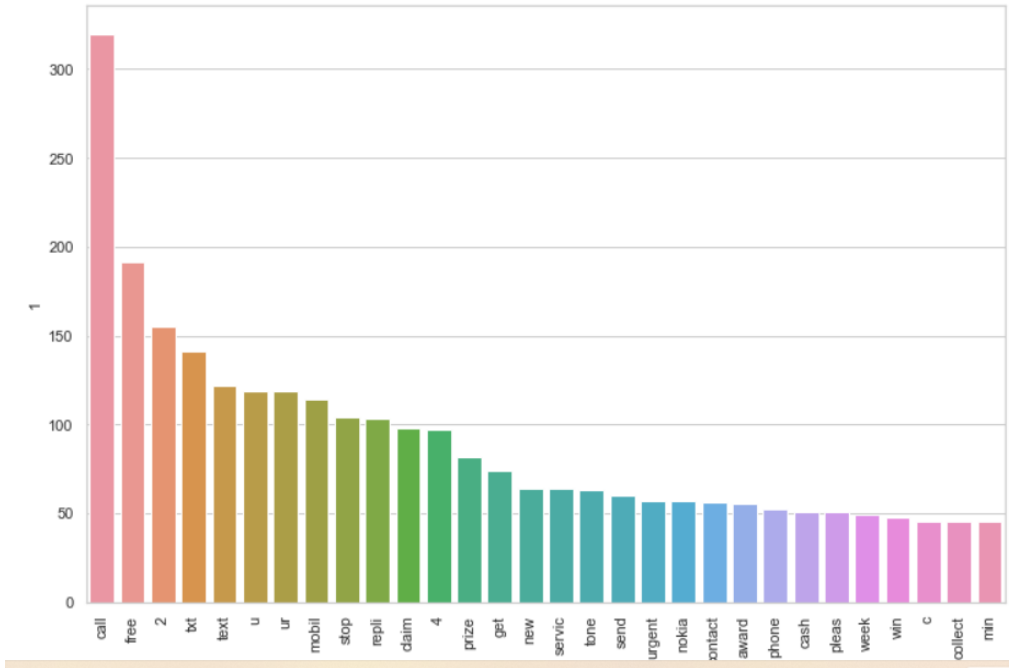
```
spam_wc = wc.generate(df[df['target']==1]['transformed_text'].str.cat(sep = " "))
```

```
plt.figure(figsize=(12,8))
plt.imshow(spam_wc);
```

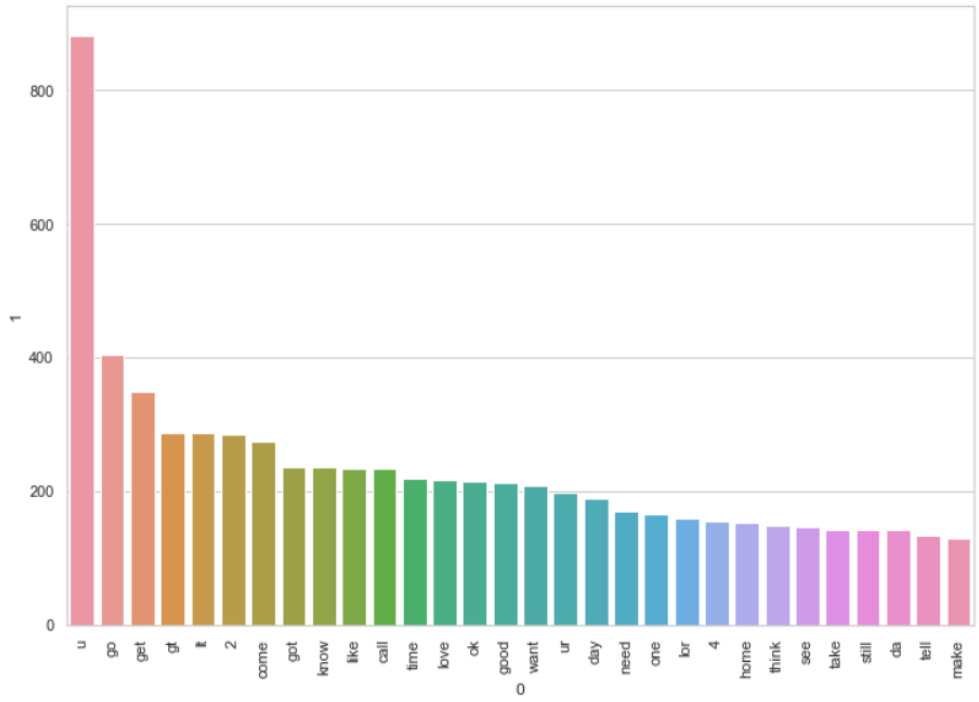

```
ham_wc = wc.generate(df[df['target']==0]['transformed_text'].str.cat(sep = " "))
```

```
plt.figure(figsize=(12,8))
plt.imshow(ham_wc);
```


```
1 from collections import Counter
2 plt.figure(figsize=(12,8))
3 sns.barplot(pd.DataFrame(Counter(spam_corpus).most_common(30))[0], pd.DataFrame(Counter(spam_corpus).most_common(30))[1])
4 plt.xticks(rotation = 'vertical');
```



```
plt.figure(figsize=(12,8))
sns.barplot(pd.DataFrame(Counter(ham_corpus).most_common(30))[0], pd.DataFrame(Counter(ham_corpus).most_common(30))[1])
plt.xticks(rotation = 'vertical');
```



CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

From the whole evaluation we found out that the spam emails can be classified and can be stopped doing harm to the users.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

I found visualisation a very useful technique to infer insights from dataset.
The ROC AUC plot gives large info about the false positive rate and True positive rate at various thresholds.

We are able to classify the emails as spam or non-spam. With high number of emails lots if people using the system it will be difficult to handle all possible mails as our project deals with only limited amount of corpus

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

Since the data contained less number of '1' target labels. The trained model will be limited in scope for this label. More data of spam can definitely improve the model's performance on identification of Spam mails.