IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

SMS SPAM DETECTION

[1]V.Iswarya Kumari,[2]J.Sirisha, [3]T.Lakshmi Prasanna, [4]M.Jaya Tanuja, [5]M.Rishitha

[1,3,4,5]B. Tech Students, Department of Information Technology
[2]Assistant Professor, Department of Information Technology
Prasad V Potluri Siddhartha Institute of Technology, Vijayawada, Andhra Pradesh, India.

Abstract: In our everyday lives, the conversation has been going through SMS (Short Message Service) for various purposes. Many organizations communicate our request processing through SMS, which is easy to communicate. Many fraudulent URLs and short links are sent through SMS, and we don't know if they are safe or not. If we click on the links and give them our information, they may misuse it and threaten us with money. So, we can't predict them by seeing whether they are spam or not. To address this problem, we have come across machine learning algorithms and a natural language toolkit that help tokenize and preprocess every word. To come up with a user interface that makes better use of availability and can predict spam and not spam. In this paper, we are going to check all the classification algorithms and choose the most accurate one to build and save the model so we can implement it in the user interface.

***Index Terms*: Machine Learning Algorithm, Natural Language Toolkit, SMS Filtering, Naïve Bayes**

I. INTRODUCTION

In the communication between organizations and people, we transfer information through messages. Daily, we communicate with lots of people, so we will have different sector contacts, so we don't bother about the safety of our information. Some members can impersonate any other person or any other organization while communicating professionally and giving exciting offers. If we observe daily, we will be getting numerous messages giving different offers. These are the types of messages that are reward-based. Another group of fraud messages tells us to give personal details, like housing information and bank details, to do KYC, which is a threat to our bank account and can result in losing money. We also get many important messages through SMS[1] sometimes we ignore those, thinking they are spam, leading us to process our work again. We need a user interface where we can easily predict whether they are spam or not. This analysis and text classification can be done with the help of machine learning algorithms[9], some of which are Naive Bayes and Support Vector Machines. We prepare a dataset of spam and non-spam messages, train and test it with a suitable classification algorithm that gives certain accurate solutions, and then implement it with a user interface.[11]

1.1 Existing System

Over the years, many reviews and implementations have been done on SMS spam filtering with various algorithms and techniques. Some of the algorithms could be time-consuming, and some techniques require more lines of code. Some algorithms like KNN, which don't process the images as much, are used. The pre-processing techniques need to be updated, and there could be an appropriate user interface to make users make use of it.

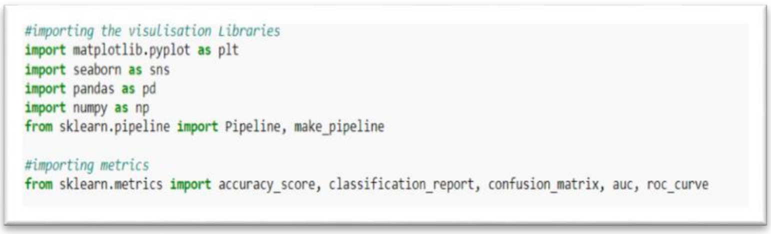
1.2 Proposed System

In the proposed system, we will use machine learning algorithms that have higher accuracy by comparing them. We need an accurate algorithm to preprocess given text and check all the possibilities for which category it belongs, whether spam or not. So, we first get to see all classification algorithms applied to the dataset. The accuracy and precision were calculated. Rather than do this first, we tokenize the text and remove stop words and a bag of words so we can focus on the main theme of the text and check which words match with spam and which words match with not-spam. We use the NLTK[6], which stands for natural language toolkit, to remove unwanted characters like punctuation, stop words, etc. The algorithm is applied to this, and the model is saved. After that, we implement this model on a website using HTML and CSS.

II. Materials And Methods

2.1 Python Libraries

There are several libraries[7] used in the data visualization of this operation. The number is used for array operations and matrices. The Pandas library is extensively used for data analysis and ML tasks. The Matplot library is used for creating static and amped visualizations. Seaborn is data visualization with the Matplot library. It supports over 40 unique map types for statistical analysis. Pickle is used for reissuing and deserializing objects.[12]

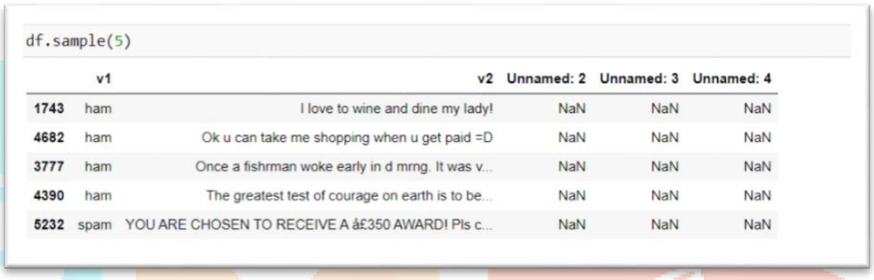


```
#importing the visulisation Libraries
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
from sklearn.pipeline import Pipeline, make_pipeline

#importing metrics
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, auc, roc_curve
```

Fig 1. Importing the Libraries

2.2 Data Set

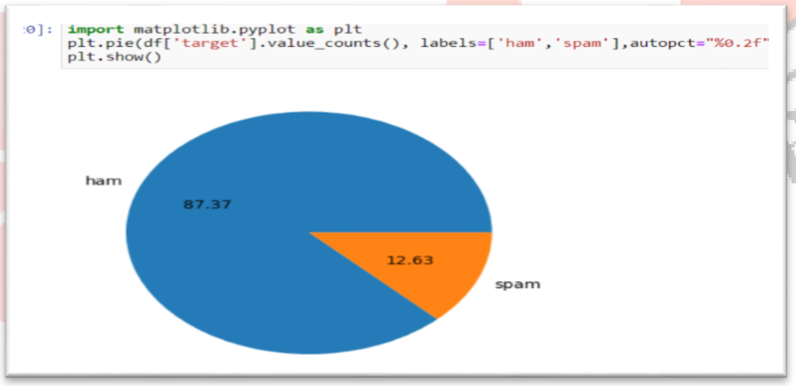
The data set[2] is needed and can be collected from Kaggle. The dataset consists of all spam and non-spam messages. Here we are taking over 5000 messages. The data is taken in the form of rows and columns in a table and saved in the form of a CSV file. It can contain text message examples and indicate whether it is spam or not. We categorize and pre-process them.



```
df.sample(5)
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
1743	ham	I love to wine and dine my lady!	NaN	NaN	NaN
4682	ham	Ok u can take me shopping when u get paid =D	NaN	NaN	NaN
3777	ham	Once a fishrman woke early in d mrng. It was v...	NaN	NaN	NaN
4390	ham	The greatest test of courage on earth is to be...	NaN	NaN	NaN
5232	spam	YOU ARE CHOSEN TO RECEIVE A â£350 AWARD! Pls c...	NaN	NaN	NaN

Fig 2. The sample five rows of the dataset



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

Fig 3. The pie chart deals with percentage of spam, not spam messages

2.3 NLTK(Natural Language Toolkit)

Language is a form of communication in which we speak, read, and write. Natural language processing (NLP) is a subfield of computer wisdom, specifically the artificial intelligence (AI), that deals with making computer systems understand and operates human language. We have many open-source NLP tools, but NLTK (the Natural Language Toolkit) scores more for ease of use and theoretical content. Python is very fast to learn, and since NLTK is written in Python, there is good literacy for NLTK as well. NLTK includes functions such as tokenization, body generation, lemmatization, tokenization, character counting, and word counting. It is very elegant and smooth to use.

2.4 HTML and CSS

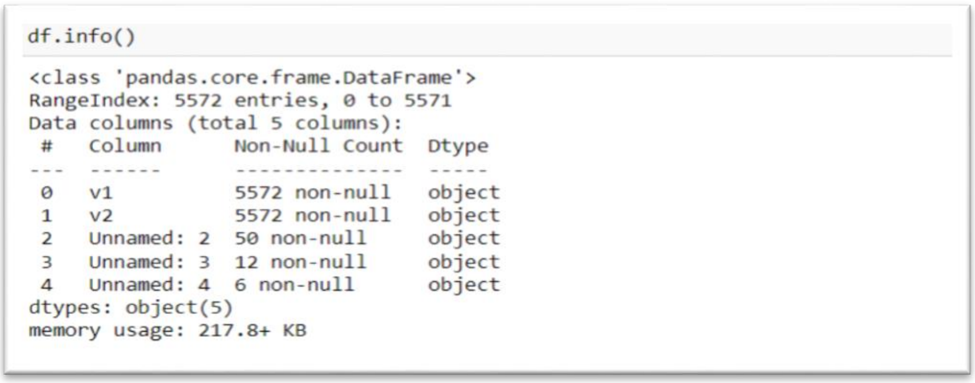
HTML (Hypertext Markup Language)[8] is used to produce web runners using markup. Hypertext describes links between web runners. A Markup language is used to describe information in markers that describe the structure of web runners. HTML is a luxury language that browsers use to render textbook, images, and other data to show them in the intended format. CSS (Cascading Style Sheets) is a language used to design web runners attractively. The reason for using CSS is to simplify the process of creating web pages. CSS allows you to style websites. CSS lets you do this independently of the HTML that makes up each website.

III. Results and Discussion

The whole process could be divided into some steps and can go on with classification. In each step, the text has undergone some process and techniques, and the main theme of the text is taken.

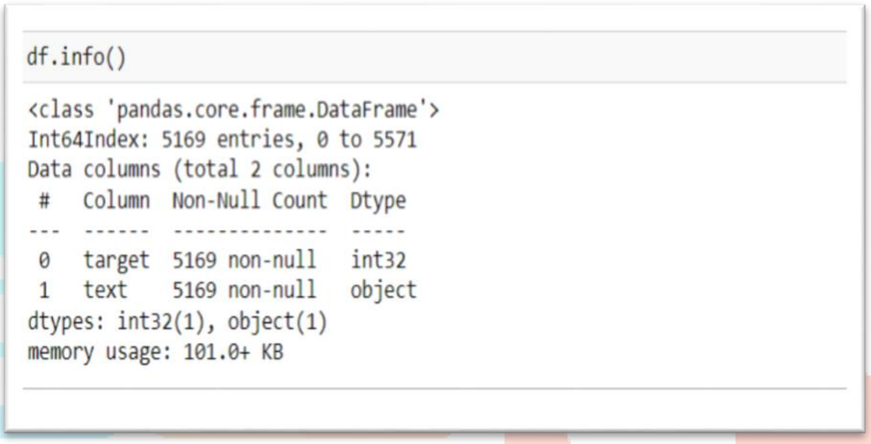
3.1 Data Cleaning

The info of the dataset is taken to know what are attributes present in the file. This shows the number of columns, rows, and null objects present in it. We need to clean all the unnecessary data present in it. The labels of the columns can be changed. If any duplicates are present can be checked and removed.



```
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
```

Fig 4. Total info on the dataset

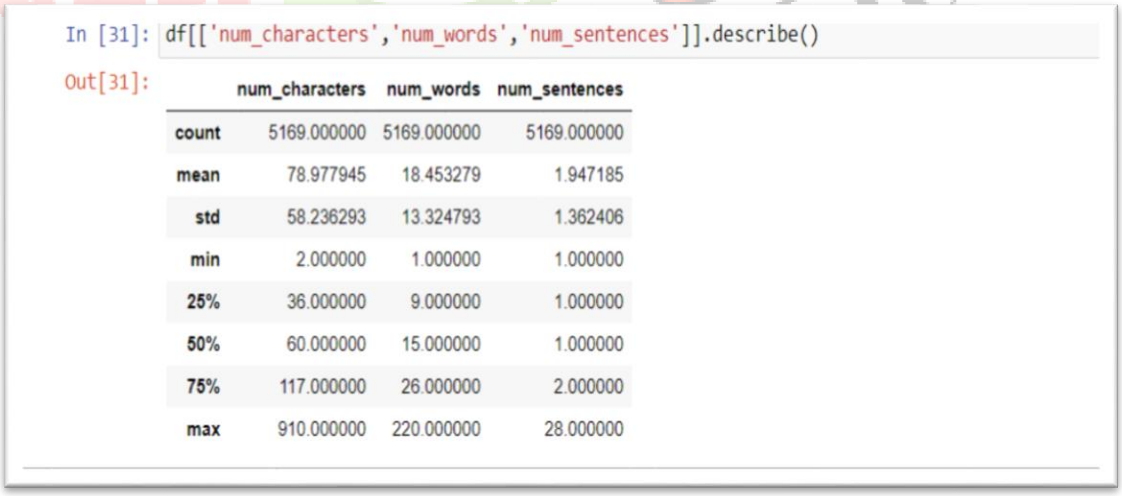


```
df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 5169 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   target  5169 non-null   int32
 1   text    5169 non-null   object
dtypes: int32(1), object(1)
memory usage: 101.0+ KB
```

Fig 5. The info after data cleaning

3.2 EDA (Exploring Data Analysis)

The EDA enables to give statistical data and analyze the total dataset with some of the visual methods. The main theme characteristics are to be concluded and the data is given. This help to have a good knowledge of the data and how each attribute is related to one other. We can further explain link each text can be divided into some sentences, words, and characters.



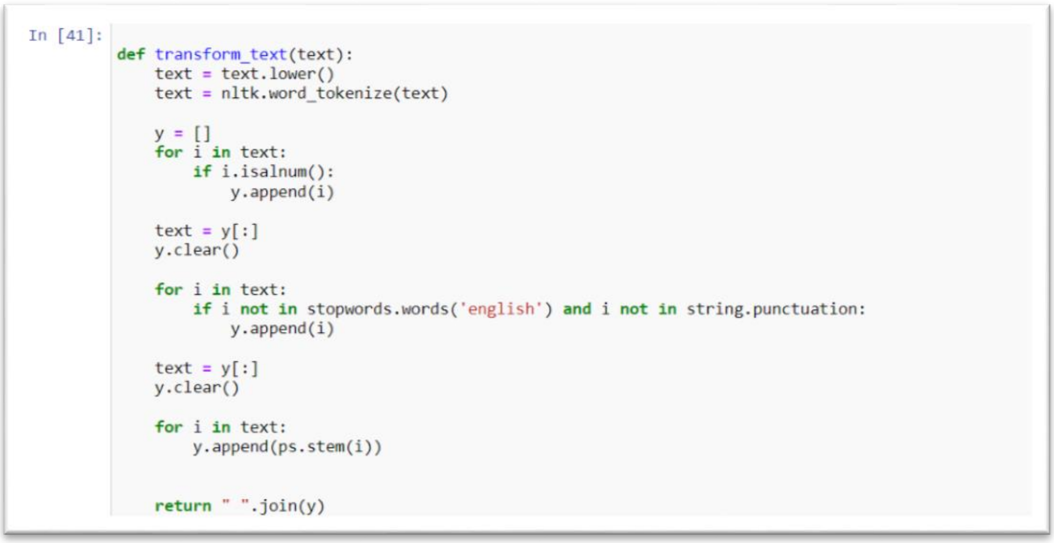
```
In [31]: df[['num_characters','num_words','num_sentences']].describe()
Out[31]:
```

	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.977945	18.453279	1.947185
std	58.236293	13.324793	1.362406
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	28.000000

Fig 6. Total texts description

3.3 Data Preprocessing

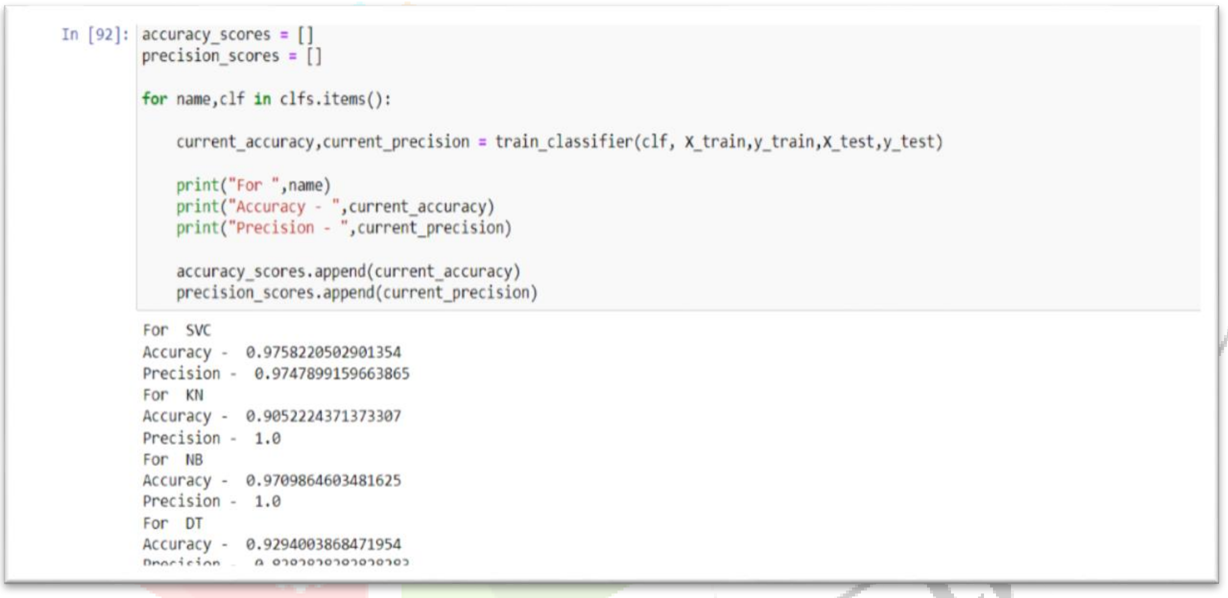
This would be the crucial step to be considered. The NLTK library is used as it goes with the sequencing process. The text is taken and turned into lowercase, and then the tokenization starts. Each text is divided into tokens so we could remove the special characters. NLTK is used to remove unnecessary stop words such as a, an, the, and punctuation. The stemming of the words is done, which means only the root verb is taken.



```
In [41]:
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 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)
```

Fig 7. Function for the data preprocessing

3.4 Model Building

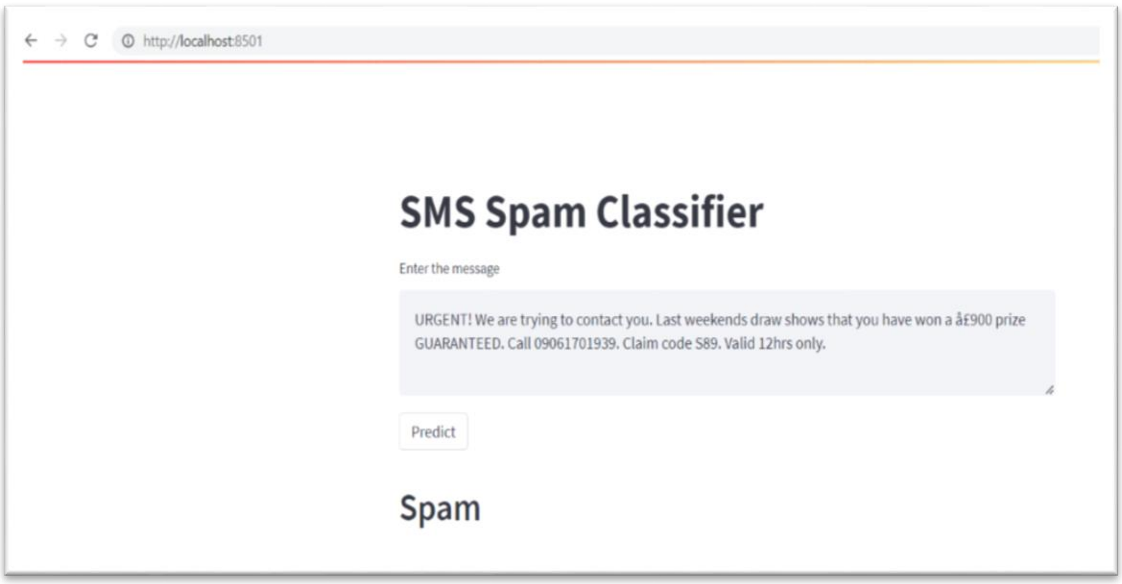


```
In [92]: accuracy_scores = []
         precision_scores = []

         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 -  0.9758220502901354
Precision -  0.9747899159663865
For  KN
Accuracy -  0.9052224371373307
Precision -  1.0
For  NB
Accuracy -  0.9709864603481625
Precision -  1.0
For  DT
Accuracy -  0.9294003868471954
Precision -  0.9292929292929293
```

Fig 8. Classification algorithms accuracy and precision score

Different algorithms[15] are applied to the dataset after splitting the test data set. The algorithms' precision and accuracy are calculated, and the comparison occurs. We examine the accuracy of the SVC (support vector machine), KN (K-nearest neighbor), NB (naive bayes), and DT (decision tree) algorithms. Here we see the accuracy is higher for NB[3] and SVC[4], but the precision is higher for NB, so we choose NB.
Using the Pickle library, we save the Naive Bayes multinomial model and implement it in the user interface to predict the output.

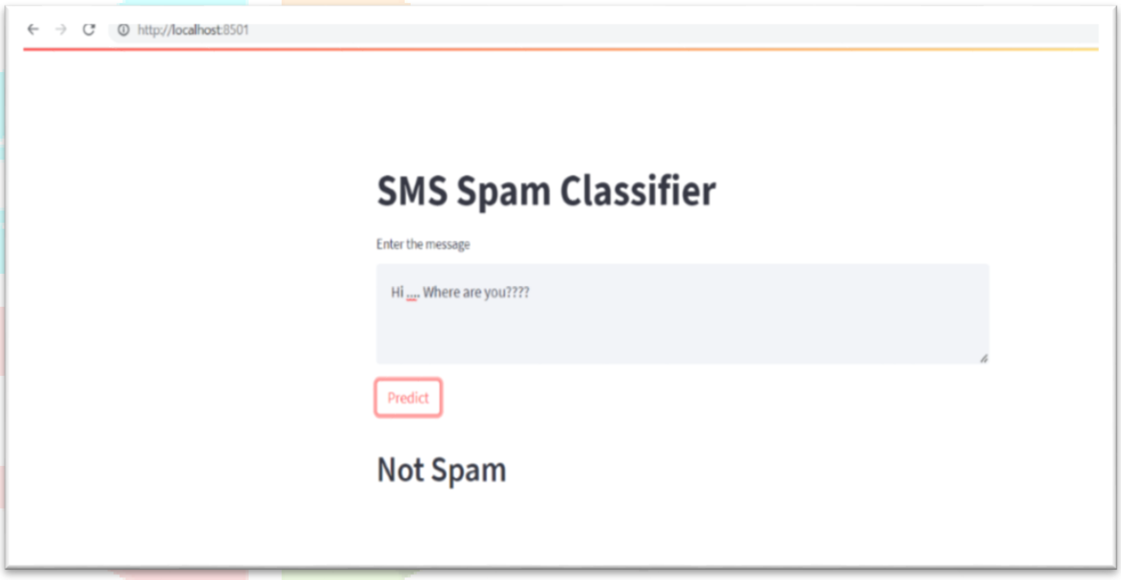
3.5 Final Output



URGENT! We are trying to contact you. Last weekends draw shows that you have won a â£900 prize GUARANTEED. Call 09061701939. Claim code S89. Valid 12hrs only.

Spam

Fig 9. The final website predicting SMS which is spam



Hi .... Where are you????

Not Spam

Fig 10. The website predicts SMS which is not spam

IV. CONCLUSION

Finally, we can see how the proposed model can predict spam and non-spam messages. The implementation of the Python libraries and the NLTK libraries gives us more beneficial techniques to tokenize and vectorize the data. This gives us an easy approach to analyse the text specifically, and the model predicts the result with the comparison of each token and observes whether it is familiar to spam or not. This whole process with the Nave Bayes model is implemented on a website, and we can predict the outcome by displaying the text in a text box and predicting the result.

REFERENCES

[1] https://towardsdatascience.com/spam-detection-in-sms-messages-3322e03300f
[2] Dataset - https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset
[3] https://www.machinelearningplus.com/predictive-modeling/how-naive-bayes-algorithm-works-with-example-and-full-code/
[4] https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm
[5] https://link.springer.com/chapter/10.1007/978-981-15-0947-6_41
[6] https://realpython.com/nltk-nlp-python/
[7] https://www.geeksforgeeks.org/libraries-in-python/
[8] https://www.geeksforgeeks.org/design-a-web-page-using-html-and-css/
[9] https://bdtechtalks.com/2020/11/30/machine-learning-spam-detection/
[10] https://blog.logrocket.com/email-spam-detector-python-machine-learning/
[11] SMS Spam Collection v.1, "http: //www.dt.fee.unicamp.br/∼tiago/smsspamcollection"
[12] Scikit-learn Ensemble Documentation, "http: //sci-kit-learn.org/stable/ modules/ensemble.html"
[13] Sedhai, S., Sun, A.: Semi-supervised spam detection in the twitter stream. IEEE Trans. Comput. Soc. Syst., 169–175 (2018)
[14] https://en.wikipedia.org/wiki/Supervised_learning
[15] https://www.kaggle.com/code/metetik/classification-algorithms-comparison