North South University

Department of Electrical & Computer Engineering

SPRING'23

Final Project Submission (Phase-3)

Course Code : CSE445

Course Title: Machine Learning

Course Instructor: Sarnali Basak (SLB)

Section: **07**

Project Name: **E-mail/SMS Based Spam Detection**

Submitted By,

Rahul Deb Roy (1931132042)
Phone: 01747128130

Moriom Islam Mou (1931333042)
Phone: 01955446803

Date of Submission: 10/06/2023

E-mail/SMS Based Spam Detection

***Abstract*—** Users can send and receive messages quickly and cheaply over the internet by using E-mail. Although email is a useful tool for exchanging information, occasionally spam is sent in mass to a large number of recipients. The idea of spam is broad because it might include chain letters, unwelcome adverts, and other things. Spam causes wastage of resources and it is very annoying problem which is being faced by almost everyone having an email account. So, filtering spam email before sending it to user's inboxes is a crucial and difficult undertaking. With this proposed model the specified message/mail can be stated as spam or not spam using Bayes theorem, Naïve Bayes Classifier and others.

***Keywords*— preprocessing, deploy, Naïve Bayes**

I. INTRODUCTION

We know, with the growing use of email communication, Email spam has become a significant challenge in recent years. Spam emails not only consume valuable time and resources but also pose a security threat as they may contain malicious content or links. The proposed system has practical applications in improving Email/SMS filtering for individuals and organizations by reducing the number of unwanted or malicious emails/SMS that reach their inboxes. This project has the potential to contribute to the field of email security and spam filtering by providing a more effective and efficient solution. In this project, we have taken e-mail Spam-Ham dataset from Kaggle– "E-mail Spam Collection Dataset". The project will involve several stages, including data collection and text preprocessing, model building, evaluation and improvement. For further improvement, we could build a website by creating a pipeline and deploy. We will explore various machine learning algorithms such as Naive Bayes, Decision Trees, Logistic Regression, Extra Trees Classifier etc.

The proposed system has practical applications in improving email filtering for individuals and organizations. By reducing the number of unwanted or malicious emails, the proposed system will help individuals and organizations to save time, increase productivity and minimize security risks.

II. DATA SET

https://www.kaggle.com/datasets/mfaisalqureshi/spam-email

III. REFERENCE

Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., & Alegre, E. (2022). A review of spam email detection: analysis of spammer strategies and the dataset shift problem. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-022-10195-4

E-mail spam and non-spam filtering using Machine Learning. (n.d.). Www.enjoyalgorithms.com. https://www.enjoyalgorithms.com/blog/email-spam-and-non-spam-filtering-using-machine-learning

IV. RESULT & DISCUSSION

As a specific output we want to predict whether an e-mail as a Spam or Ham (i.e. not Spam) one. From the dataset we had **5572** rows, but after the data cleaning process (i.e. Removing 415 duplicate values, by keeping the original values) we got **5157** rows. The dataset is saved in the 'kaggle' website and can be accessed via link mentioned above. The dataset was last updated 2yrs ago. Changes/biases was normalized in the data cleaning process.

The most relevant factors for predicting our specific output was Accuracy, Precision, Recall and Confusion Matrix. But, we are mostly concerned about the Precision than the accuracy, since our dataset is large and in this case, Precision is particularly useful when the cost of false positives is high, such as in medical diagnoses or spam email classification.

We are hoping to use 10 fold cross validation where the algorithm is then trained on nine of the folds and the remaining fold is used for testing. This process is repeated ten times, each time using a different fold for testing, and the results are averaged to give an overall performance metric. We did 90-10 train test split.

For the simple benchmark to compare the result against, is to predict/ asses the result by comparing the value of Accuracy, Precision and the Confusion Matrix. Again, another benchmark of comparison is number of characters, words and sentences and even the mean of these three, are usually greater in value for the Spam messages than the Ham messages.

The minimum level of accuracy we expect is 86%. To clarify our vision, we will create a pipeline from the codes and convert it into a website, where we expect to have a box (text field) in a window where we will provide the input texts for the testing sets and then by pressing the predict button it will give the result accordingly as spam or not spam. Each testing data will have to perform the following three steps in the pipeline before showing the result in the website though 'Streamlit' (an open-source app framework for Machine Learning):

a) Text preprocessing (Lower casing, Tokenization, Removal of special words, stops words and punctuations, Stemming)

b) Vectorization(Converting textual data into numerical vectors

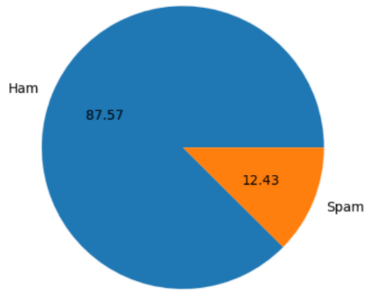
c) Application of different algorithms

The following Accuracy, Precision, Recall and Confusion Matrix for different models on the data set are mentioned below:

SL No.	Algorithms	Accuracy	Precision	Recall	Confusion Matrix
1.	K-Nearest Neighbors	0.913	1.0	0.291	$\begin{bmatrix} 905 & 0 \\ 90 & 37 \end{bmatrix}$
2.	Multinomial Naïve Bayes	0.971	1.0	0.764	$\begin{bmatrix} 905 & 0 \\ 30 & 97 \end{bmatrix}$
3.	Gaussian Naïve Bayes	0.866	0.476	0.866	$\begin{bmatrix} 784 & 121 \\ 17 & 110 \end{bmatrix}$
4.	Bernoulli Naïve Bayes	0.984	0.982	0.882	$\begin{bmatrix} 903 & 2 \\ 15 & 112 \end{bmatrix}$
5.	Random Forest	0.974	1.0	0.788	$\begin{bmatrix} 905 & 0 \\ 27 & 100 \end{bmatrix}$
6.	Extra Trees Classifier	0.978	0.991	0.827	$\begin{bmatrix} 904 & 1 \\ 22 & 105 \end{bmatrix}$
7.	SVM	0.976	0.981	0.819	$\begin{bmatrix} 903 & 2 \\ 23 & 104 \end{bmatrix}$
8.	Logistic Regression	0.958	0.938	0.709	$\begin{bmatrix} 899 & 6 \\ 37 & 90 \end{bmatrix}$
9.	Decision Trees	0.936	0.821	0.614	$\begin{bmatrix} 888 & 17 \\ 49 & 78 \end{bmatrix}$
10.	Voting Classifier	0.983	1.0	0.858	$\begin{bmatrix} 905 & 0 \\ 18 & 109 \end{bmatrix}$

Here, the **Voting Classier** is working the best, since it's giving the highest accuracy with 100% precision that is, no false positives are there. So, we will hopefully proceed with this algorithm.

V. TASK (FINAL WORK)



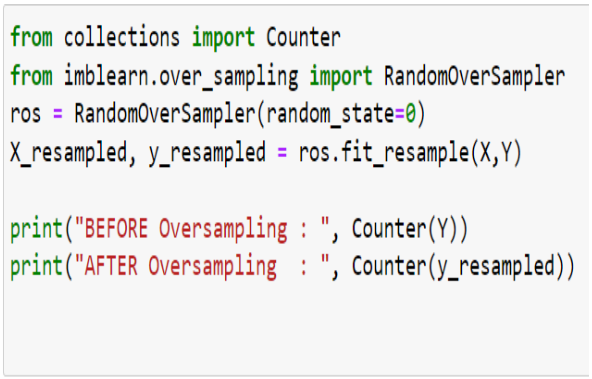
Here, **Ham = 4516** (87.57%) and **Spam = 641** (12.43%). Thus, we can see how our dataset is pretty imbalanced (Biased).

So, the correction given to us during the presentation was to balance the dataset that is <u>to make the biased dataset unbiased</u> and to show the result accordingly.

Changed Part:

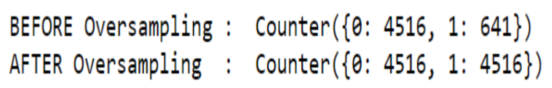
We did Oversampling that basically aims to balance the class distribution by increasing the number of instances in the minority class. By oversampling the minority target class (Spam), the resulting dataset became more balanced. Since, our dataset is large, **RandomOverSampler** can be a simpler and effective option.

Here, 0 = Ham & 1 = Spam

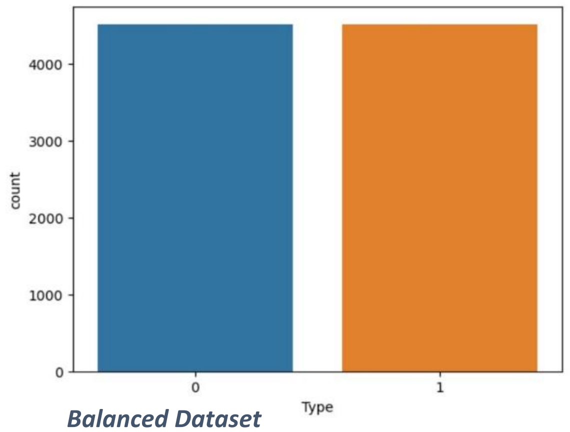


```
from collections import Counter
from imblearn.over_sampling import RandomOverSampler
ros = RandomOverSampler(random_state=0)
X_resampled, y_resampled = ros.fit_resample(X,Y)

print("BEFORE Oversampling : ", Counter(Y))
print("AFTER Oversampling  : ", Counter(y_resampled))
```



```
BEFORE Oversampling :  Counter({0: 4516, 1: 641})
AFTER Oversampling  :  Counter({0: 4516, 1: 4516})
```

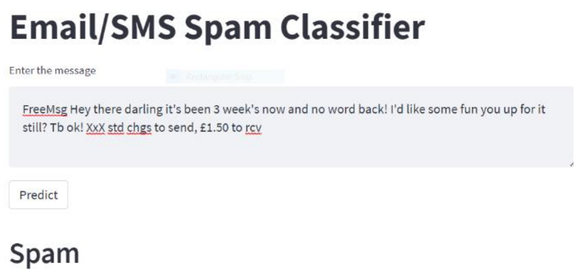


Balanced Dataset

SL No.	Algorithms	Accuracy	Precision	Recall	Confusion Matrix
1.	K-Nearest Neighbors	0.96	0.78	0.98	$\begin{bmatrix} 870 & 35 \\ 2 & 125 \end{bmatrix}$
2.	Multinomial Naïve Bayes	0.97	0.83	0.96	$\begin{bmatrix} 880 & 25 \\ 5 & 122 \end{bmatrix}$
3.	Gaussian Naïve Bayes	0.88	0.51	1.0	$\begin{bmatrix} 783 & 122 \\ 0 & 127 \end{bmatrix}$
4.	Bernoulli Naïve Bayes	0.99	0.99	0.93	$\begin{bmatrix} 904 & 1 \\ 9 & 118 \end{bmatrix}$
5.	Random Forest	0.99	1.0	0.98	$\begin{bmatrix} 905 & 0 \\ 2 & 125 \end{bmatrix}$
6.	Extra Trees Classifier	0.99	1.0	0.98	$\begin{bmatrix} 905 & 0 \\ 2 & 125 \end{bmatrix}$
7.	SVM	0.98	0.85	0.98	$\begin{bmatrix} 883 & 22 \\ 3 & 124 \end{bmatrix}$
8.	Logistic Regression	0.98	0.92	0.95	$\begin{bmatrix} 894 & 11 \\ 6 & 121 \end{bmatrix}$
9.	Decision Trees	0.92	0.62	0.83	$\begin{bmatrix} 841 & 64 \\ 22 & 105 \end{bmatrix}$
10.	Voting Classifier	0.98	1.0	0.86	$\begin{bmatrix} 905 & 0 \\ 18 & 109 \end{bmatrix}$

Now, after balancing the dataset we can see the changes in Accuracy, Precision, Recall and Confusion Matrix for each models. Here, for the balanced dataset **Random Forest** and **Extra Trees Classifier** are working the best. Since, both are giving the highest accuracy with 100% precision that is, no false positives are there. So, we can proceed with either one of the algorithms.

<u>Screenshots of the results</u>:

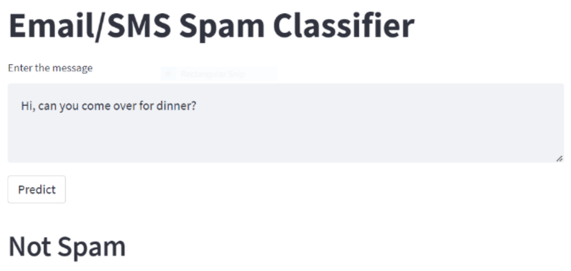


Email/SMS Spam Classifier

Enter the message

FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, £1.50 to rcv

Spam



Email/SMS Spam Classifier

Enter the message

Hi, can you come over for dinner?

Not Spam