

## CHAPTER 1

### INTRODUCTION

Tom Mitchell provided a modern definition of machine learning: "A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ ."

Example: playing checkers.

$E$  = the experience of playing many games of checkers

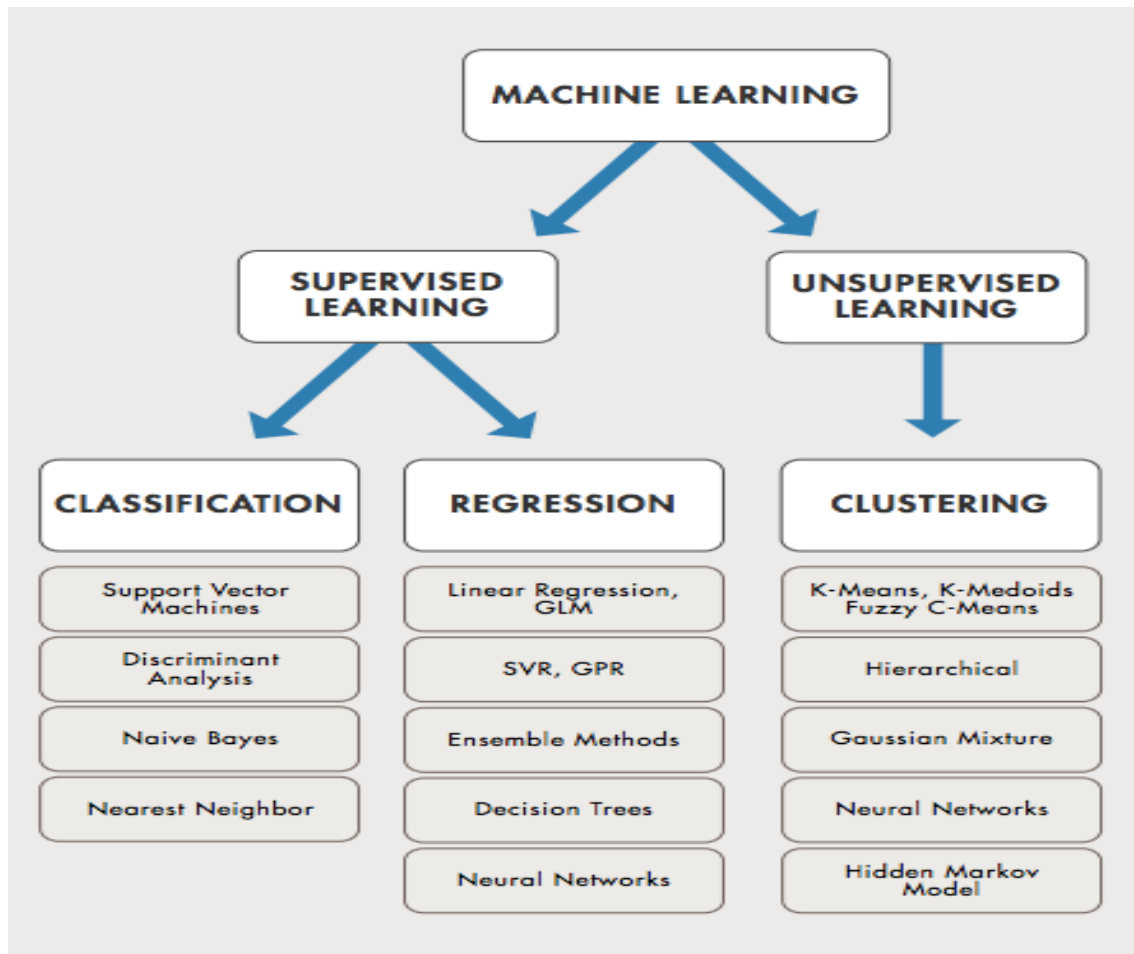
$T$  = the task of playing checkers.

$P$  = the probability that the program will win the next game.

Machine learning algorithms are mainly grouped into 2 types as mentioned in **FIG 1.1**.

**Supervised learning:** Supervised learning, in the context of artificial intelligence (AI) and machine learning, is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. The term supervised learning comes from the idea that an algorithm is learning from a training dataset, which can be thought of as the teacher. Naïve bayes algorithm comes under supervised learning.

**Unsupervised learning** is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.



**FIG 1.1 MACHINE LEARNING ALGORITHM CLASSIFICATION**

Spam classifier using Naïve Bayes algorithm is an efficient way of classifying emails whose accuracy is proven to be 98% efficient. As emails have become an effective, faster and cheaper way of communication, it is better to have an effective method to classify them as spam and ham.

Spam refers to an unwanted email while ham refers to a legitimate, wanted, solicited mail. Spam filtering is the process of classifying and organizing emails based on a pre-defined criteria. Spams are increasing day by day because of an effective, fast and cheaper way of transfer of information with each other. According to a survey it is proven that a person receives more spams than hams which can lead to loss of confidential information and can bring in insecurity among users. Spam is considered to be one of the major problems that attack the existence of electronic mails.

The existing system is completely manual. All emails are read and analyzed by a person for understanding information such as vendor of the business, description, price and tax of the product. When it comes to dealing with a huge number of emails the task becomes more complex.

Maintaining such a system manually is a tedious and time-consuming process. It is very difficult to create a consistent and integrated data of all invoices. An ultimate solution for handling a huge amount of emails that contains invoice details is needed.

The trivial method of manually going through mails is next to impossible in a world of spammers and marketers that looks to flood the inbox. Hence, it is necessary to automate this process, with the advancement and development of machine learning and natural language processing it's possible to classify them in a matter of no time.

The familiar Bayesian approach is being used. A dataset from Kaggle which contains test cases of spam and ham messages sent via email is used here along with various python libraries, such as pandas, nltk, re, sklearn. Naïve bayes algorithm is based on one of the simple but yet powerful concepts of probability called **Bayes theorem** which is given is **FIG 1.2.**

Probability of B occurring  
given evidence A has already  
occurred

Probability of A occurring

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Probability of A occurring  
given evidence B has already  
occurred

Probability of B occurring

**FIG 1.2: Bayes theorem**

Using this we compute the formula for Naïve bayes algorithm which is given below.

$$P(\text{spam} | \text{word}) = P(\text{spam}) * P(\text{word} | \text{spam}) / (P(\text{spam}) * P(\text{word} | \text{spam}) + P(\text{ham}) * P(\text{word} | \text{ham}))$$

Accuracy of this project is calculated using a confusion matrix shown in the **FIG 1.3**.

TP = TRUE POSITIVE

FN = FALSE NEGATIVE

FP= FALSE POSITIVE

TN= TRUE NEGATIVE

N = TOTAL INSTANCES

	<i>Class 1 Predicted</i>	<i>Class 2 Predicted</i>
<b>Class 1 Actual</b>	TP	FN
<b>Class 2 Actual</b>	FP	TN

**FIG 1.3 CONFUSION MATRIX**

The accuracy can be determined using the formula given below

$$\text{ACCURACY} = (\text{TP} + \text{TN}) / \text{N}$$

## 1.1 PURPOSE OF STUDY

**Purpose:** To train a machine learning model that has the ability to classify the emails as ham and spam. As people tend to receive a lot of emails every day and it is be a tedious process to segregate each of them as spam and ham, this could also be a risk as few might believe that a junk mail is genuine further it is highly important to automate the process

with efficient accuracy. This project helps in exploring the accuracy of the project. To train the dataset and test the dataset and to give the output of the analysis.

**Scope:** This project aims at fetching the raw data and training it using Naïve Bayes algorithm and finally testing the accuracy of the algorithm. The accuracy is what matters the most as emails are one of the most effective means of communication. It is highly important for them to be secured. The accuracy of this project is proven to be 98% accurate.

## 1.2 PROBLEM STATEMENT

Email spamming is increasing day by day because of effective, fast and cheap way of exchanging information with each other. According to the investigation, User receives

spam mails >ham mails

Spam is a major problem that attacks the existence of electronic mails. So, it is very important to distinguish ham emails from spam emails. Hence, developing a robust algorithm for the same is required.

## 1.3 MOTIVATION OF PROJECT

Email spamming has become one of the major issues in every possible sectors of the industry. Speaking from the user point of view, it is proven that people receive more spams compared to hams and possibilities of people losing their confidential information or credential is increasing day by day and the main reason behind it is not having a robust method in order to classify emails into ham and spam.

Intrigued about how the computers can process language and make calculation. Understanding the mathematical concepts that goes into building this model. Intrigued about how the data sets are imported, trained, and tested.

## 1.4 METHODOLOGY

To implement a spam classifier using Naïve Bayes, the first step is to import libraries such as pandas. Pandas is used for reading the dataset into the program and creating a data frame.

The next step is to clean and preprocess the data in the dataset which includes stemming of the data using porter.stemmer and removing all the stopwords such as him, her, in, the etc. from the dataset, all this is done using the library nltk. Once the cleansing and preprocessing of the data is done, the next step is to create a bag of words using countvectorizer which takes the most frequently occurring words. Convert the independent and dependent variables into 0's and 1's as the machine learning algorithm will only take numeric values for processing. Splitting of the dataset into train and test is done using sklearn.split. Using Naïve Byaes classifier, model is trained using the train dataset. Test dataset is then sent

through the trained model and the predicted output is recorded and the accuracy is determined.

Naïve Bayes classifier algorithm:

$$P(\text{spam}|\text{word}) = \frac{P(\text{spam}) * P(\text{word}|\text{spam})}{(P(\text{spam}) * P(\text{word}|\text{spam}) + P(\text{ham}) * P(\text{word}|\text{ham}))}$$

## CHAPTER 2

### SYSTEM REQUIREMENT SPECIFICATION

A system Requirement Specification is an essential part of software documentation that specifies what the requirements of the project. This includes hardware, software and any other additional requirements that are needed for the program to execute. Typically, system requirement specification document lists the functional, non-functional, expendable and non-expendable resources required for the project. In this chapter hardware, software and language specifications required for implementing spam classifier using Naïve Bayes is listed below.

#### 2.1 Hardware System Configuration:

Processor	- Intel Core i5
Speed	- 1.8 GHz
RAM	- 256 MB (min)
Hard Disk	- 10 GB

#### Software System Configuration:

Operating System	- Windows 10
Programming Language	- Python
Compiler	- Anaconda

## 2.2 ABOUT THE LANGUAGE

Spam classifier using Naïve Bayes algorithm is implemented using python on anaconda spyder terminal. Most of the machine learning algorithms are implemented using python.

Python has various features such as:

- 1) it is considered to be an expressive language which means it is more understandable and readable.
- 2) it is free and is an open source – python can be downloaded for free from the website and is available with the source code which makes it an open source.
- 3) multiple libraries – Python comes with many built in packages which makes it easier for the users to build applications rapidly.
- 4) Cross platform language – python can run on different platforms such as windows, linux, unix etc.

### MODULES USED:

**NLTK:** A Python library package, one of the leading platforms in processing human language data. It comes packed with powerful functions like word tokenizer from sentences, tokenize sentences or words together. It provides with the ability to identify and remove stop words ("is", "the" ..) from the dataset for accurate analysis.

**PANDAS:** Python Data Analysis Library pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. pandas is a NUMFocus sponsored project. It enables us to read or import the dataset into the project.



**RE:** A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in UNIX world.

The Python "re" module provides regular expression support. Python's built-in "re" module provides excellent support for regular expressions, with a modern and complete regex flavor. Regular expressions are extremely useful in extracting information from text such as code, log files, spreadsheets, or even documents.

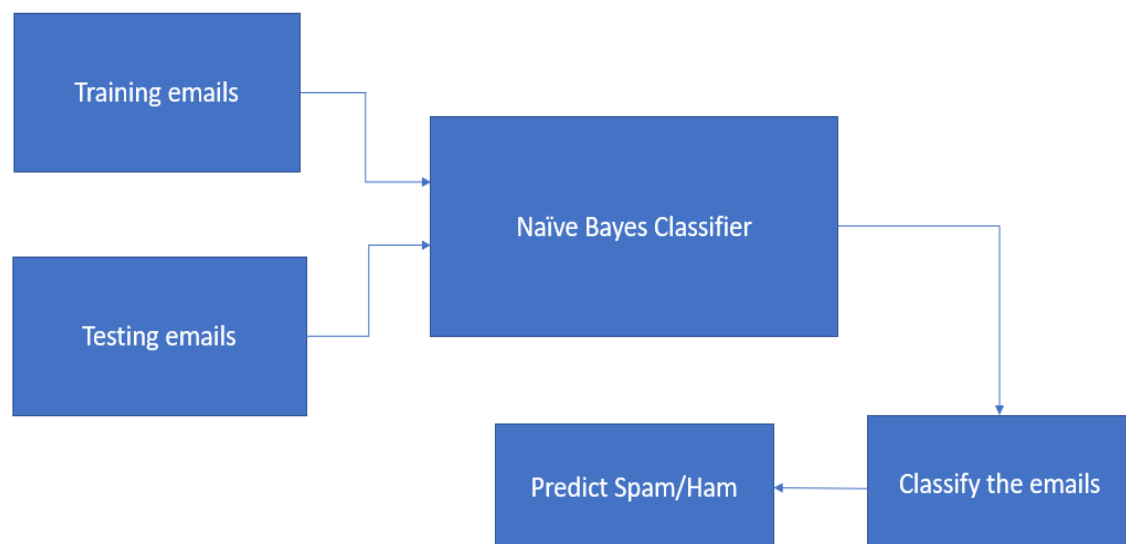
## CHAPTER 3

### SYSTEM DESIGN

system design specification describes the modules and functions of a project in detail so that the reader can understand the behavior and functions of the program. Typically, a system design document includes an architectural diagram, algorithm, flowchart, and the code for the project. In this chapter system design of spam classifier using Naïve Bayes algorithm is specified.

#### 3.1 ARCHITECTURE

A dataset consists of  $n$  emails which is split into 2 datasets (train dataset and test dataset). The train dataset consists of  $X$  which denotes messages and  $Y$  which denotes label (ham or spam).  $X$  and  $Y$  are converted into binary. The train data is fed into the Naïve bayes classifier which helps in training the model. And then the test dataset is fed into the naïve bayes classifier which classifies the emails and predicts whether they are ham or spam.



**Fig 3.1: Architecture of spam classifier using Naïve Bayes Algorithm**

### 3.2 ALGORITHM

Step 1: start

Step 2: Import the necessary packages such as pandas, sklearn.

Step 3: read the dataset using panda.

Step 4: split the dataset into test and train using sklearn.

Step 5: tokenize the data using NLTK.

Step 6: stemming the data using porter stemmer.

Step 7: Remove the stop words such as and, as, is etc.

Step 8: train the model using Naïve bayes algorithm.

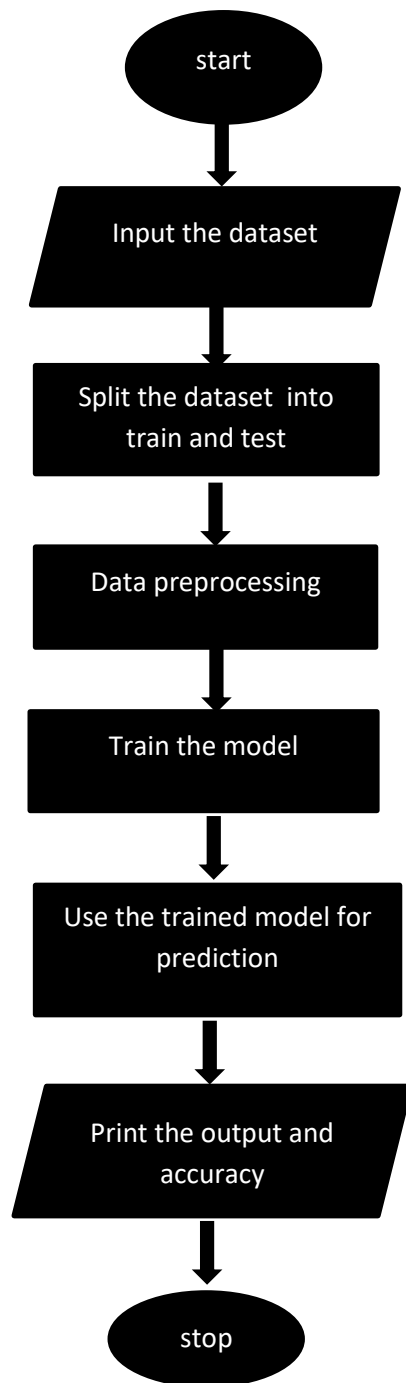
Step 9: use the trained model for prediction.

Step 10: print the accuracy and the result.

Step 11: stop.

### 3.3 FLOWCHART

The below figure (**FIG 3.1**) is a flowchart which is a graphical representation of the algorithm of a program that aids in understanding how a program works from when execution begins to the end. As illustrated in Figure 3.1, the program execution begins by reading the dataset into the program. The dataset is then split into train dataset and test dataset respectively. Both the datasets undergo a data preprocessing step where the data is stemmed, converted into lowercase and all the stop words are removed from it. The train dataset is now sent through the spam classifier to train the model. Further the test dataset is classified and prediction of ham and spam of the test dataset is performed.



**FIG 3.1 FLOWCHART FOR SPAM CLASSIFIER USING NAÏVE BAYES ALGORITHM**

### 3.4 CODE AND IMPLEMENTATION

#### **# importing the Dataset**

```
import pandas as pd
```

```
messages = pd.read_csv('SMSSpamCollection', sep='\t',  
                        names=["label", "message"])
```

#### **#Data cleaning and preprocessing**

```
import re
```

```
import nltk
```

```
nltk.download('stopwords')
```

```
from nltk.corpus import stopwords
```

```
from nltk.stem.porter import PorterStemmer
```

```
ps = PorterStemmer()
```

```
corpus = []
```

```
for i in range(0, len(messages)):
```

```
    review = re.sub('[^a-zA-Z]', ' ', messages['message'][i])
```

```
review = review.lower()
```

```
review = review.split()
```

```
review = [ps.stem(word) for word in review if not word in  
stopwords.words('english')]
```

```
review = ' '.join(review)
```

```
corpus.append(review)
```

### **# Creating the Bag of Words model**

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
cv = CountVectorizer(max_features=2500)
```

```
X = cv.fit_transform(corpus).toarray()
```

```
y=pd.get_dummies(messages['label'])
```

```
y=y.iloc[:,1].values
```

### **# Train Test Split**

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state =
0)

# Training model using Naive bayes classifier

from sklearn.naive_bayes import MultinomialNB

spam_detect_model = MultinomialNB().fit(X_train, y_train)

y_pred=spam_detect_model.predict(X_test)

from sklearn.metrics import confusion_matrix

confusion_n=confusion_matrix(y_test,y_pred)

from sklearn.metrics import accuracy_score

accuracy=accuracy_score(y_test,y_pred)
```

## CHAPTER 4

### RESULTS AND DISCUSSIONS

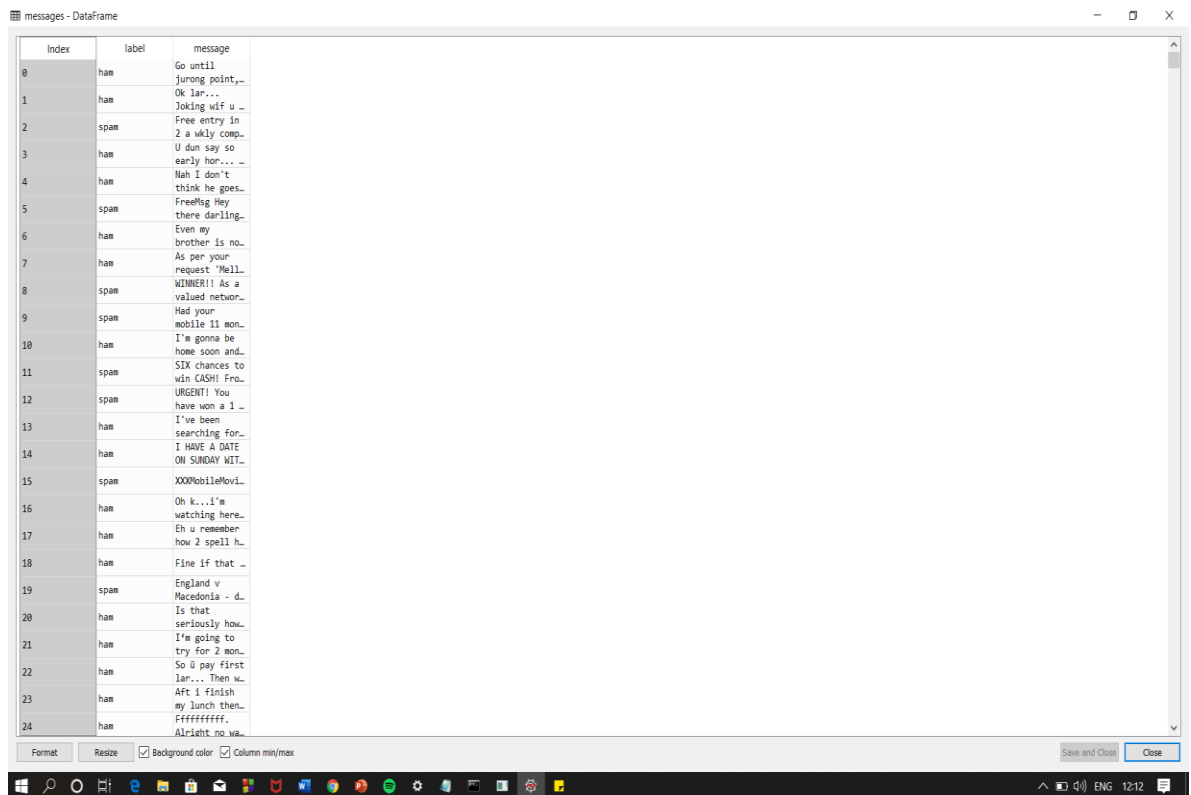
The results of a project to be displayed is an important step in documentation. It gives the reader an idea about how the program works when seen from a user's perspective and also the method of using the program. In the following screenshots, the working of the spam classifier using Naïve Bayes algorithm is demonstrated.

#### 4.1 SUMMARY

Naïve bayes spam classifier is mainly used for classifying the emails. Using the trained data, data to be tested is classified into spam and ham with the accuracy of 98% using naïve bayes algorithm. The dataset is read using pandas. All the stopwords are removed using `nlTK.download('stopwords')`. Stemming of the data is done using `ps = PorterStemmer()`. Bag of words are created using sklearn countvectorizer. convert the labels (ham =0 and spam=1 ) using `get_dummies`. Split the dataset into train and test. 20% for testing. Train the dataset using `sklearn.naive_bayes import MultinomialNB` Test the dataset using the trained model. Find the accuracy by importing `accuracy_Score` from sklearn.



## 4.2 OUTPUT (SNAPSHOTS)

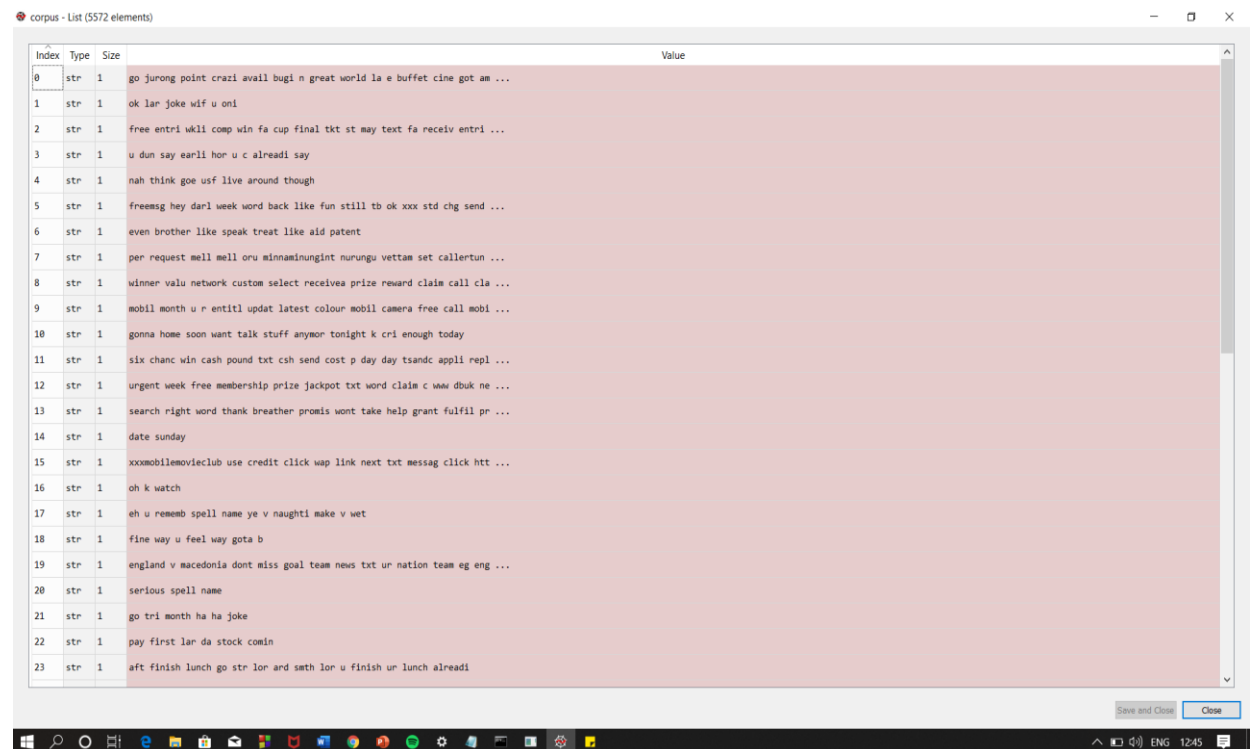


Index	label	message
0	ham	Go until junong point...
1	ham	Ok lar... Joking wif u...
2	spam	Free entry in 2 a wkly comp...
3	ham	U dun say so early hor... -
4	ham	Nah I don't think he goes...
5	spam	FreeMsg Hey there darling...
6	ham	Even my brother is no...
7	ham	As per your request 'Well...
8	spam	WINNER!! As a valued network...
9	spam	Had your mobile 11 mon...
10	ham	I'm gonna be home soon and...
11	spam	SIX chances to win CASH! Fro...
12	spam	URGENT! You have won a 1...
13	ham	I've been searching for...
14	ham	I HAVE A DATE ON SUNDAY MIT...
15	spam	XXXMobileMovie...
16	ham	Oh k...i'm watching here...
17	ham	Eh u remember how 2 spell h...
18	ham	Fine if that -
19	spam	England v Macedonia - d...
20	ham	Is that seriously how...
21	ham	I'm going to try for 2 mon...
22	ham	So u pay first lar... Then w...
23	ham	Aft i finish my lunch then...
24	ham	fffffffff. Alrighr no wa...

**FIG 4.1: converting the dataset into a dataframe (messages) consisting of 2 columns (label, message)**

The above figure (**FIG 4.1**) depicts Importing the data set using pandas and then separate the dataset into two columns one being label which consists of ham and spam and the other consisting of message. And is read into a dataframe called message.

## Spam Classifier

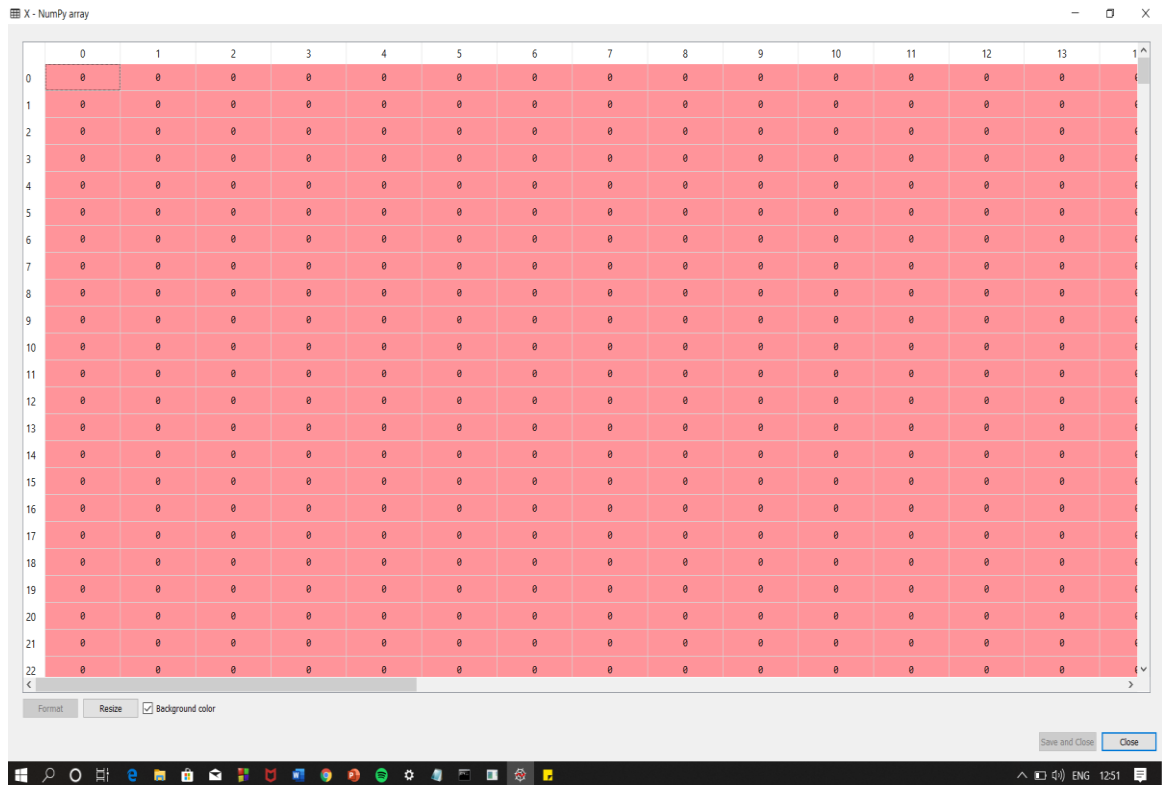


Index	Type	Size	Value
0	str	1	go jurong point crazi avail bugi n great world la e buffet cine got am ...
1	str	1	ok lar joke wif u oni
2	str	1	free entri wkli comp win fa cup final tkt st may text fa receiv entri ...
3	str	1	u dun say earli hor u c already say
4	str	1	nah think goe usf live around though
5	str	1	freemsg hey darl week word back like fun still tb ok xxx std chg send ...
6	str	1	even brother like speak treat like aid patent
7	str	1	per request mell mell oru minnamungint nurungu vettam set callertun ...
8	str	1	winner valu network custom select receivea prize reward claim call cla ...
9	str	1	mobli month u r entitl updat latest colour mobil camera free call mobi ...
10	str	1	gonna home soon want talk stuff anymor tonight k cri enough today
11	str	1	six chanc win cash pound txt csh send cost p day day tsandc appli repl ...
12	str	1	urgent week free membership prize jackpot txt word claim c waw dbuk ne ...
13	str	1	search right word thank breather promis wont take help grant fulfil pr ...
14	str	1	date sunday
15	str	1	xxmobilemovieclub use credit click wap link next txt messag click htt ...
16	str	1	oh k watch
17	str	1	eh u rememb spell name ye v naughti make v wet
18	str	1	fine way u feel way gota b
19	str	1	england v macedonia dont miss goal team news txt ur nation team eg eng ...
20	str	1	serious spell name
21	str	1	go tri month ha ha joke
22	str	1	pay first lar da stock comin
23	str	1	aft finish lunch go str lor and smth lor u finish ur lunch already

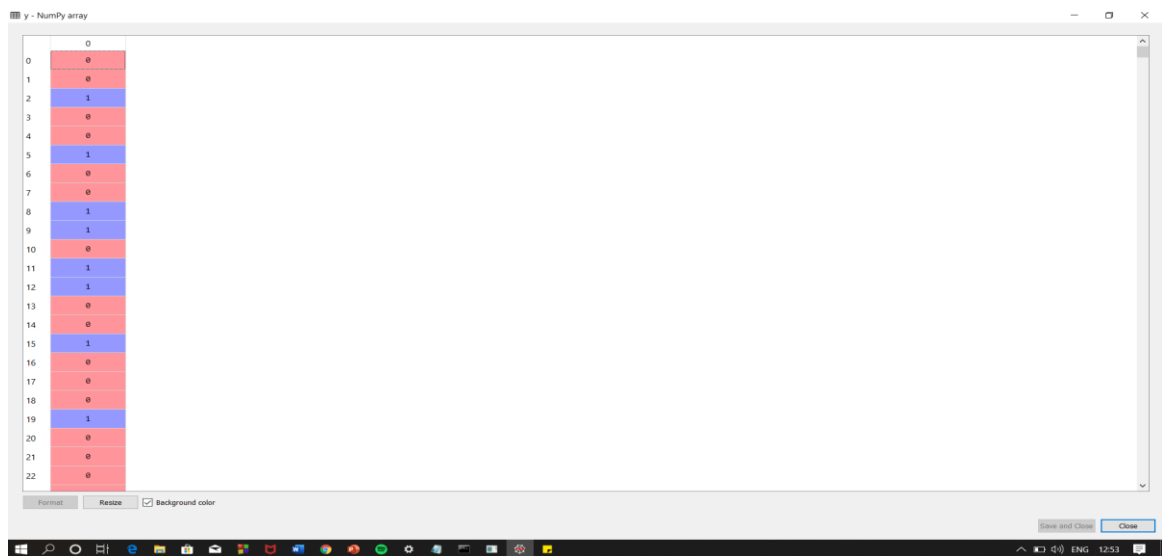
**FIG 4.2: cleaning and preprocessing the data present in dataframe and storing it in corpus.**

The above figure (**FIG 4.2**) depicts Data cleaning and preprocessing where removing all the stop words such as in, is, the, or, but, him, etc. takes place as these words do not contribute towards identifying whether the given message is ham or spam. Then stemming of the words takes place where words such as gone, go, going is converted into go and then the data is converted into lowercase. And all of this is appended into corpus.

## Spam Classifier

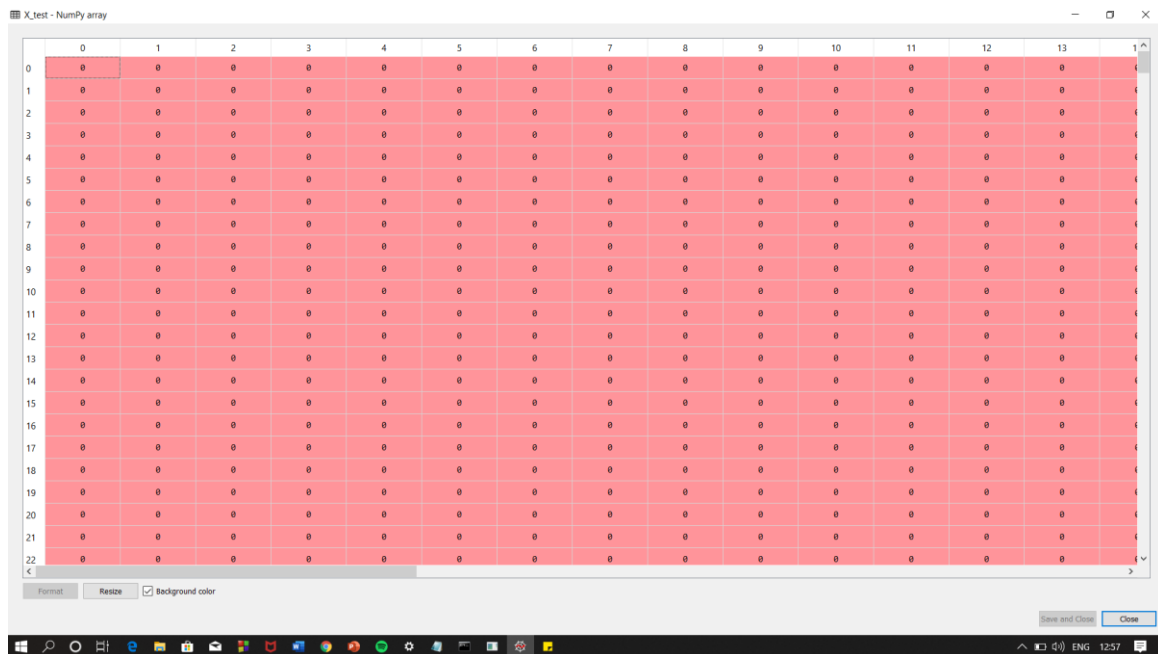


**FIG 4.3: Transferring the corpus into an array and storing it in X values  
(independent variables)**



**FIG 4.4: converting the label (ham=0, spam=1) stored in Y values  
(dependent variable)**

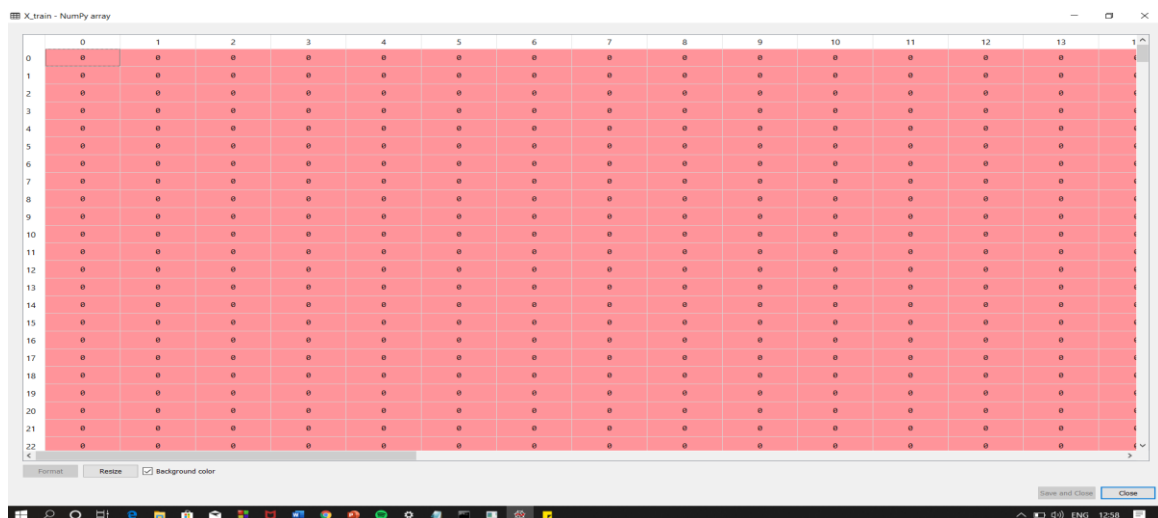
The above figures depict Creating a bag of words using countvector and transfer the corpus ( **FIG 4.2**) into an array and this will be independent values or variables which will be appended to X values (**FIG 4.3**). further, convert the labels that is ham and spam into numbers as the machine learning model takes only numerical data as input. Hence, convert all the hams to 0's and all the spam to 1's. and is stored in Y values (**FIG 4.4**).



X\_test - NumPy array

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0

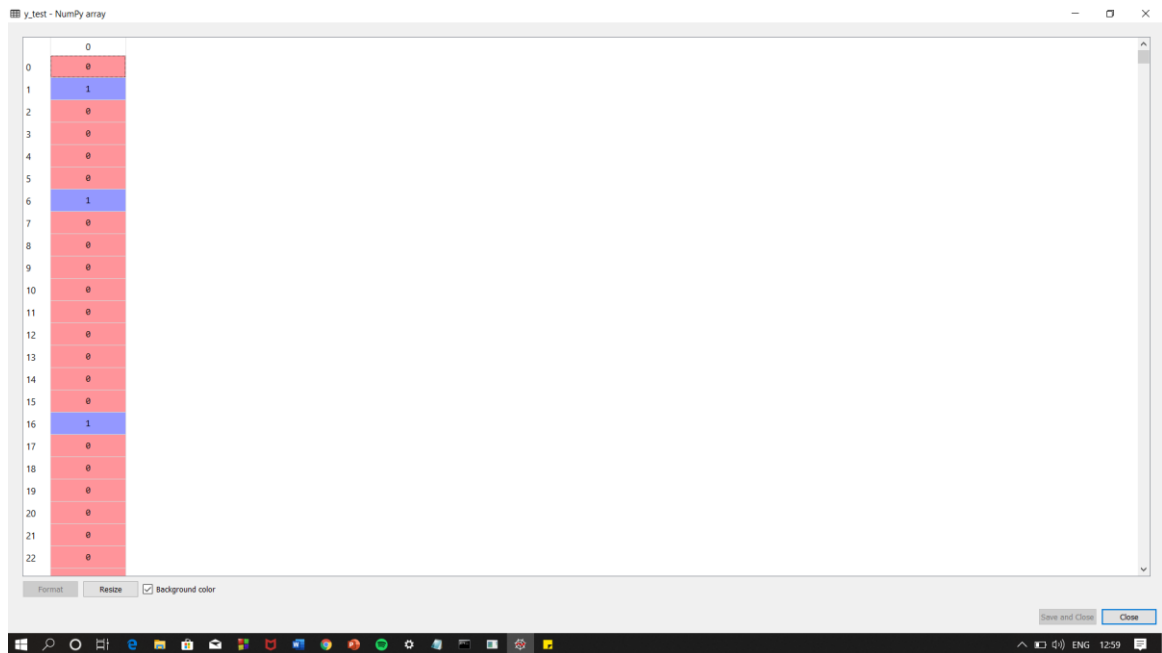
**FIG 4.5: The dataset is split into X\_test dataset**



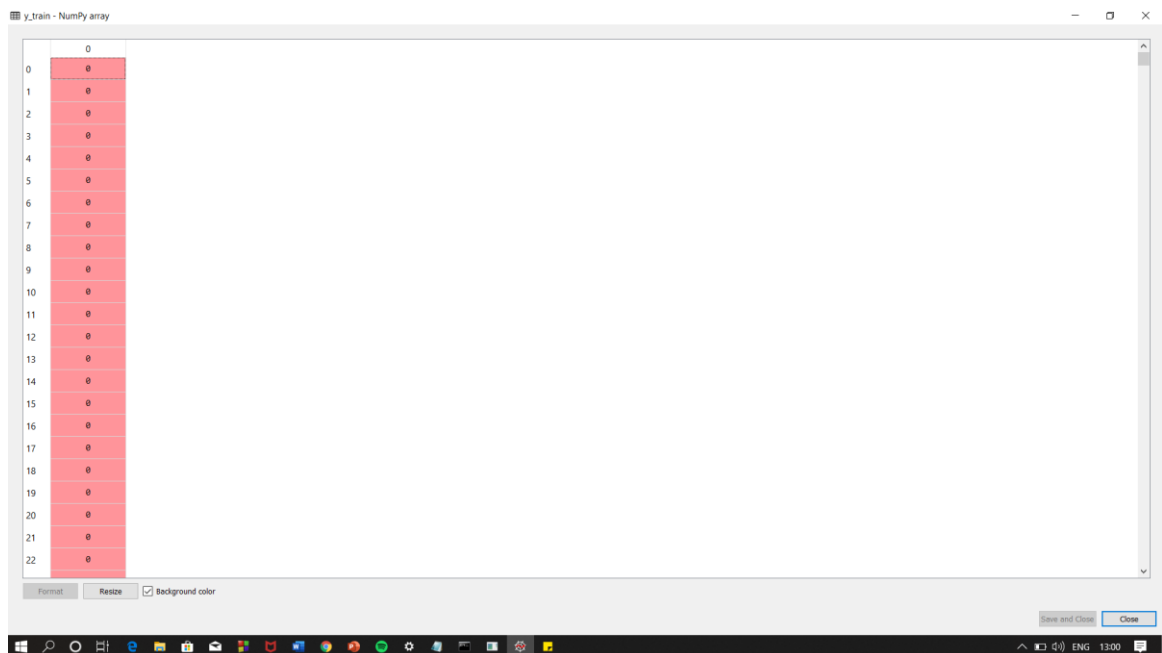
X\_train - NumPy array

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FIG 4.6: The dataset is split into X\_train dataset**

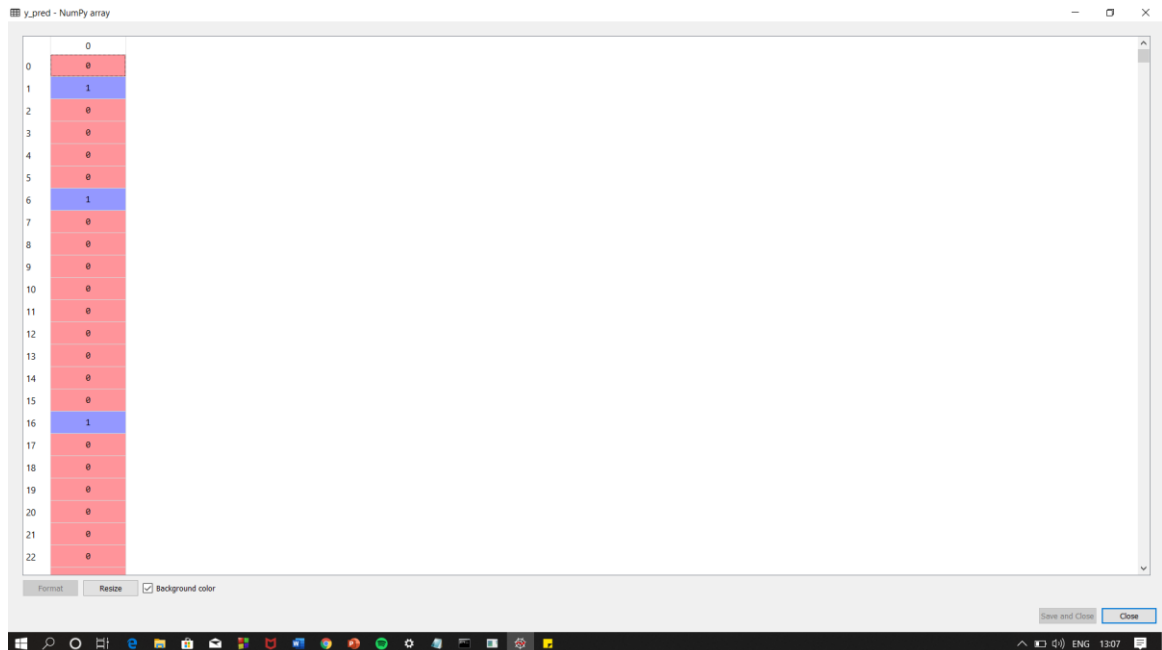


**FIG 4.7: The dataset is split into Y\_test dataset**



**FIG 4.8: The dataset is split into Y\_train dataset**

The above figures (**FIG 4.5 TO FIG 4.8**) depicts the splitting of the dataset into 20% test dataset and 80% train dataset and also shows the independent variables of the train and test dataset (**FIG 4.5, FIG 4.6**). Dependent variables of the train dataset and test dataset (**FIG 4.7, FIG 4.8**).



**FIG 4.9: Y\_pred (predicted output values after testing)**

Above figure (**FIG 4.9**) depicts about training the model using naïve bayes classifier algorithm by passing X\_train (**FIG 4.6**) and Y\_train (**FIG 4.8**). pass the X\_test values (**FIG 4.5**) and predict output values and store it in Y\_pred.

confusion\_n - NumPy array

	0	1
0	946	9
1	7	153

FIG 4.10: Confusion matrix of the model

Variable explorer

Name	Type	Size	Value
X_test	int64	(1115, 2500)	[[0 0 0 ... 0 0 0] [0 0 0 ... 0 0 0]
X_train	int64	(4457, 2500)	[[0 0 0 ... 0 0 0] [0 0 0 ... 0 0 0]
accuracy	float64	1	0.9856502242152466

FIG 4.11: Accuracy of the naïve bayes classifier

The above figures (**FIG 4.10, FIG 4.11**) depict the calculation of accuracy which can be calculated by performing the following steps:

Compare **FIG 1.3** WITH **FIG 4.10** from this we can compute

**TP (TRUE POSITIVE) =946 refer (FIG 1.3)**

**TN( TRUE NEGATIVE)=153 refer (FIG 1.3)**

**N= TOTAL INSTANCES = 946+9+7+153 = 1115**

Hence the accuracy can be determined using the formula given below

**ACCURACY = (TP+TN)/N**

by substituting the values we get

**ACCURACY = (946+153)/1115 = 0.9856 shown in FIG 4.11**



## CHAPTER 5

### CONCLUSION

The proposed problem statement has been solved and realized through python programming language. The program successfully trains the model on the given dataset. The trained model is further tested and then commences to compute the analysis on the input message using the trained model and outputs the result. The above system forms the basis of message classification using Naive Bayes theorem in Statistics. The project was successfully completed giving an almost accurate prediction and analysis

As emails are one of the most effective ways of communication, it is mandatory to come up with an algorithm that will help in segregating the emails as ham and spam. Ensuring that the algorithm works accurately will be our utmost concern.

Naïve bayes has proven to be the best suitable algorithm to train the data and to test with the accuracy level of about 98%.

## REFERENCES

- [1] <https://www.javatpoint.com/python-features>
- [2] <https://towardsdatascience.com/spam-classifier-in-python-from-scratch-27a98ddd8e73>
- [3] <https://medium.com/@contactsunny/how-to-split-your-dataset-to-train-and-test-datasets-using-scikit-learn-e7cf6eb5e0d>
- [4] <https://towardsdatascience.com/spam-filtering-using-naive-bayes-98a341224038>