



# PROJECT NAME

# EMAIL SPAM FILTER

SUBMITTED BY  
JAMIL ISAH

07/12/2022

FACULTY OF COMPUTERS AND INFORMATION KAFRELSHIEK UNIVERSITY EGYPT  
LEVEL 300

[JAMILUISAH26@GMAIL.COM](mailto:JAMILUISAH26@GMAIL.COM)  
KANO STATE NIGERIA

+2348061133526, +201220552809

SUPERVISED BY  
DR AMR ABOHANY, DR AMENA MAHMOUD, ENGINEER ATRAB AHMED

## ABSTRACT

The number of complicated documents and texts has grown exponentially in recent years, necessitating a deeper comprehension of machine learning techniques in order to effectively classify texts in numerous applications. Numerous machine learning techniques have exceeded

results in the processing of natural language. These learning algorithms' effectiveness depends on their ability to comprehend intricate models and non-linear correlations in data. However, it can be difficult for researchers to locate appropriate structures, architectures, and methods for text classification. This paper offers a quick review of text categorization techniques. This review discusses several ways for extracting text features, dimensionality reduction strategies, currently used algorithms and procedures, and evaluation techniques. Finally, each technique's limits and their application to actual problems are examined.

The number of complicated documents and texts has grown exponentially in recent years, necessitating a deeper comprehension of machine learning techniques in order to effectively classify texts in numerous applications. many machines learning the number of complicated documents and texts has grown exponentially in recent years, necessitating a deeper comprehension of machine learning techniques in order to effectively classify texts in numerous applications. In natural language processing, numerous machine learning techniques have shown astounding outcomes. These learning algorithms' effectiveness depends on their ability to comprehend intricate models and non-linear correlations in data. However, it can be difficult for researchers to locate appropriate structures, architectures, and methods for text classification. In this article, a There is a brief discussion on text classification algorithms. This review discusses several ways for extracting text features, dimensionality reduction strategies, currently used algorithms and procedures, and evaluation techniques. The limitations of each technique and how they apply to actual problems are then examined

## Table of content

Chapter 1 .....	4
1.1 Project overview.....	5
1.2 Document scope .....	6
1.3 Intended Audience and reading Suggestion .....	6
1.4 Product Scope .....	7
1.5 References .....	7
Chapter 2 .....	8
2.1 Product perspective .....	9
2.2 Product Function .....	9
2.3 User Classes .....	9
2.3.1 User Case .....	10
2.3.1.1 Register .....	10
2.3.1.2 Login.....	10
2.3.1.3 Receive emails .....	10
2.3.1.4 View Emails .....	11
2.3.1.5 View Account information .....	11
2.3.1.6 View Spam Report .....	11
2.3.1.7 View Register Report .....	11
2.3.1.8 Logout .....	11
2.4 Data Flow Diagram .....	12
2.5 Functional Decomposition Diagram .....	13
2.6 State Diagram .....	14
A – Operating Environment .....	14
B – Design and Implementation Constraints .....	15
C – Binary Classification .....	16
D – Naïve Byes .....	16
2.7 User Documentation .....	16
2.8 Assumptions And Dependencies .....	17
Chapter 3 .....	18
3.1 User Interfaces .....	19
3.2 Software Interfaces .....	20
Chapter 4 .....	21
4.1 System feature .....	22
4.1.1 Description and Priority .....	22
4.1.2 Programming Practice .....	22
4.1.3 Collaboration Across Project and tools .....	22
4.1.4 Data organizing and cleaning .....	22
4.1.5 Data visualization and Sharing .....	22
4.1.6 Teaching Data Science Skills .....	22
4.2 Stimulus/Responses .....	23
4.3 Functional Requirements .....	24
Chapter 5 .....	25
5.1 Performance Requirements .....	26
5.2 Safety Requirements .....	26
5.3 Business Rules .....	27
5.4 Future Work .....	27
5.5 Conclusions/Recommendations .....	28

# CHAPTER 1

## INTRODUCTION

## 1.1 PROJECT OVERVIEW

About 50% of all email messages sent globally are spam. Email providers – including both Microsoft and Google – spend a lot of time, effort, and money making sure that most of that spam doesn't end up in your inbox (and that legitimate emails don't end up in your spam folder)

In recent times, unwanted commercial bulk emails called spam has become a huge problem on the internet. The person sending the spam messages is referred to as the spammer. Such a person gathers email addresses from different websites, chatrooms, and viruses. 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 77% of the whole global email traffic. 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. According to report from Kaspersky lab, in 2015, the volume of spam emails being sent reduced to a 12-year low. Spam email volume fell below 50% for the first time since 2003. In June 2015, the volume of spam emails went down to 49.7% and in July 2015 the figures were further reduced to 46.4% according to anti-virus software developer Symantec. This decline was attributed to reduction in the number of major botnets responsible for sending spam emails in billions. Malicious spam email volume was reported to be constant in 2015. The figure of spam mails detected by Kaspersky Lab in 2015 was between 3 million and 6 million. Conversely, as the year was about to end, spam email volume escalated. Further report from Kaspersky Lab indicated that spam email messages having pernicious attachments such as malware, ransomware, malicious macros, and JavaScript started to increase in December 2015. That drift was sustained in 2016 and by March of that year spam email volume had quadrupled with respect to that witnessed in 2015. In March 2016, the volume of spam emails discovered by Kasp

## 1.2 Document Scope

This Functional Specification document is a document which provides detailed information about how the email spam filter classification tool will function. This document clearly outlines the actors that will be interacting with the system and in what way they are expected to interact with the system, what they should expect from the system and what they can and cannot do while interacting with the system.

## 1.3 Intended Audience and Reading Suggestion

While the software requirement specification (SRS) document is written for a more general audience, this document is intended for any individuals or companies directly affect or not affect in Email spam .. This document need not be read sequentially; users are encouraged to jump to any section they find relevant. Below is a brief overview of each part of the document.

- Part 1 (Introduction) o This section offers a summary of the Email Spam filter project, including goals and objectives, project overview project scope, general system details, References and some major constraints associated with the intended platform
- Part 2 (Overall description) o Readers interested in how Email Spam filter organizes and handles data should consult this section, which covers data structures and flow patterns utilized by the system, , product perspective, product functions, the system class by class, including interface details, class hierarchies, operating environment, design and implementation Constraints, User Documentations, assumptions and dependencies
- Part 3 (External interface Requirements) o This section describes User interfaces, software interfaces, Communication interfaces.
- Part 4 (System features) o This section covers all of the details related to the Description and priority, stimulus responses, functional Requirements. Readers can view this section for a tentative glimpse of what the final product will look like.
- Part 5 (Other Non-functional Requirements) This section describes performance/design constraints, process details, safety Requirements, Security Requirements, Software Quality Attributes Business rules and algorithmic models

## 1.4 Product Scope

- ❖ Collecting the Dataset
- ❖ Libraries
- ❖ Load Data
- ❖ EDA (1)
- ❖ Data Cleaning
- ❖ EDA (2)
- ❖ Preprocessing
- ❖ Model Building
- ❖ Performance measurements
- ❖ Cross Validation
- ❖ GUI interface

## 1.5 References

- [Support.wharton.upenn.edu/help/spam-filtering-overview](https://support.wharton.upenn.edu/help/spam-filtering-overview)
- [Securrence.com](https://www.securrence.com)
- [nobledesktop.com classes-top-uses-for-jupyter-notebook](https://nobledesktop.com/classes-top-uses-for-jupyter-notebook)
- [Simplelearn](https://simplelearn.com)
- E-mail address: [gbengadada@unimaid.edu.ng](mailto:gbengadada@unimaid.edu.ng) (E.G. Dada)

## **CHAPTER 2**

### **OVERALL DESCRIPTION**

## 2.1 PRODUCT PERSPECTIVE

- ✓ External perspective:
- ✓ Interaction perspective:
- ✓ Structural perspective:
- ✓ Behavioral perspective

## 2.2 PRODUCT FUNCTIONS

Spam is most well-known for spreading viruses and scams to unwitting people across the internet, but it can actually cause plenty of problems for the modern business. This is why effective spam filtering, like Securence spam filtering, is an important part of running a successful business in the 21st century. Here are just a few reasons why spam filtering is important for not only keeping you safe from viruses, but also for helping your company be more effective and successful.

### 1. It Streamlines Inboxes

The average office worker receives roughly 121 emails per day, half of which are estimated to be spam. But even at 60 emails a day, it is easy to lose important communications to the sheer number that are coming in. This is one of the secret benefits of spam filtering that people do not know about: it simply streamlines your inbox. With less garbage coming into your inbox, you can actually go through your emails more effectively and stay in touch with those who matter.

### 2. Protect Against Malware

Malware, viruses, and other forms of malicious attacks are heading to people's email inboxes every day. Some of these can be easily weeded out by your internet provider's own spam filters, but spam gets smarter every day. Smarter spam gets into more inboxes, which makes it more likely to be opened and more likely to cause harm. With spam filtering, you can stay on top of the many spam tactics that are being used today so you can ensure that your email inboxes stay free of harmful messages.

### 3. Keeps You Compliant

Many small and medium sized businesses are losing out on important clients today because their cybersecurity is not up to par. Spam filtering is a major part of any cybersecurity plan, and it helps you stay compliant with the wishes and demands of companies and agencies that are concerned about their information. Without proper spam filtering, you could unwittingly put spyware in your emails and break security protocols. The result could be a loss of business, reputation, and ultimately income.

### 4. It Saves You Money

Every day, someone falls prey to a phishing scam, a particular kind of spam-based scheme where someone thinks they are getting a legitimate email and ends up divulging credit card information. Sometimes it is a personal credit card, sometimes it is a company credit card. In both instances, the end result is losing valuable time and money to a scam.

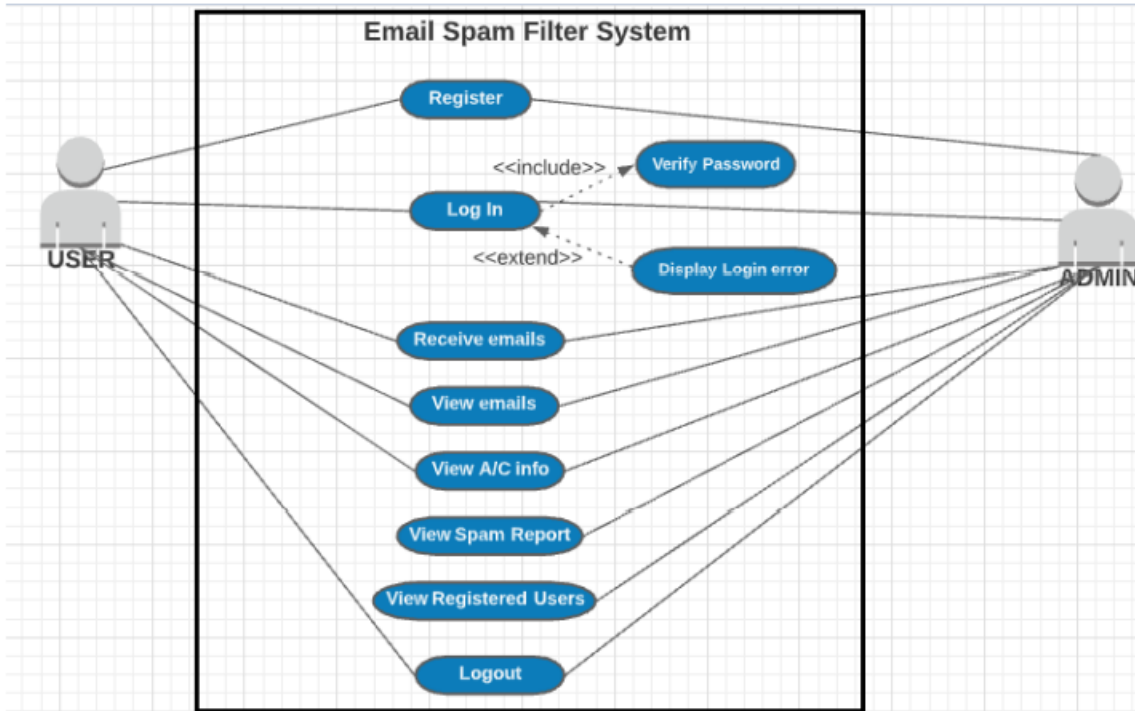
Spam filtering is also incredibly affordable, making it a cheap but extremely effective way to keep yourself safe.

Your inboxes need to be effective tools for communication, not a place where anyone can get into and start hitting you with useless or dangerous emails. That is why spam filtering is such an important aspect of modern businesses. Rather than relying on outdated, free spam filtering services, choose Securence spam filtering. With up-to-the-minute security protocols, it can help your business communicate more effectively while keeping malware out of your inboxes, all for less than you may think.

## 2.3 User classes



### 2.3.1 User case



#### 2.3.1.1 - Register

Primary Actors: User, Admin

Precondition: The user has successfully registered to the web application.

Steps:

- User clicks on register button on the desktop application
- User enters their username, email address and password twice and clicks register

Expected Result: The user has registered to the system successfully

#### 2.3.1.2 - Login

Primary Actors: User, Admin

Precondition: The user has successfully registered with the system.

Steps:

- User visits the website
- User enters their email address and password on the login screen
- User clicks Login
- If credentials match the database for this user a successfully login will occur
- The user will have access to their account

Expected Result: The user successfully logs into the application and is presented with the welcome screen.

#### 2.3.1.3- Receive Emails

Primary Actors: User, Admin

Precondition: The recipient is successfully logged into the system. An email is successfully sent to the recipient.

Steps:

1. The user will login to the system
2. The user will navigate to their inbox folder to view their email
3. The users most recent email received will appear at the top of the list on the inbox screen

Expected Result: The email is successfully received and is visible to the user in the correct folder.

#### **2.3.1.4- View Emails**

Primary Actors: User, Admin

Precondition: The user is successfully logged into the system.

Steps:

- 1.The user will login to the system
2. The user will navigate to the specific folder (sent, inbox or spam folder).

Expected Result: An email which seems suspicious is displayed in the spam folder and has a spam score of 1 in the database. An email received with spam score of 0 will be displayed in the inbox folder. All the users sent mails will be displayed in the sent folder.

#### **2.3.1.5 - View Account Information**

Primary Actors: User, Admin

Precondition: The user is successfully logged into the system.

Steps:

1. The user is successfully logged into the system
2. The user visits the account page.
  3. The users' username and email address they used to register for the application will display.

Expected Result: Their account information displays.

#### **2.3.1.6 - View Spam report**

Primary Actors: Admin

Precondition: The administrator has successfully logged in to the system.

Steps:

- 1.The administrator is successfully logged into the system
- 2.The administrator visits the report page

Expected Result: The spam report is displayed to the Administrator

#### **2.3.1.7. View Register report**

Primary Actors: Admin Precondition: The administrator has successfully logged in to the system.

Steps:

1. The administrator is successfully logged into the system
2. The administrator visits the register page

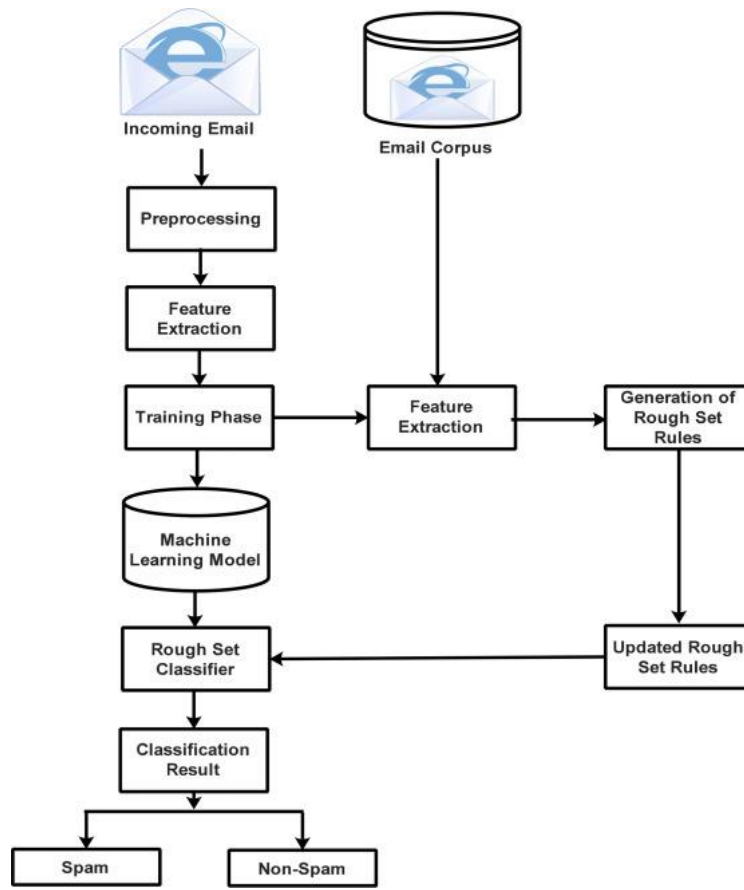
Expected Result: All registered users are displayed to the Administrator

#### **2.3.1.8. Logout**

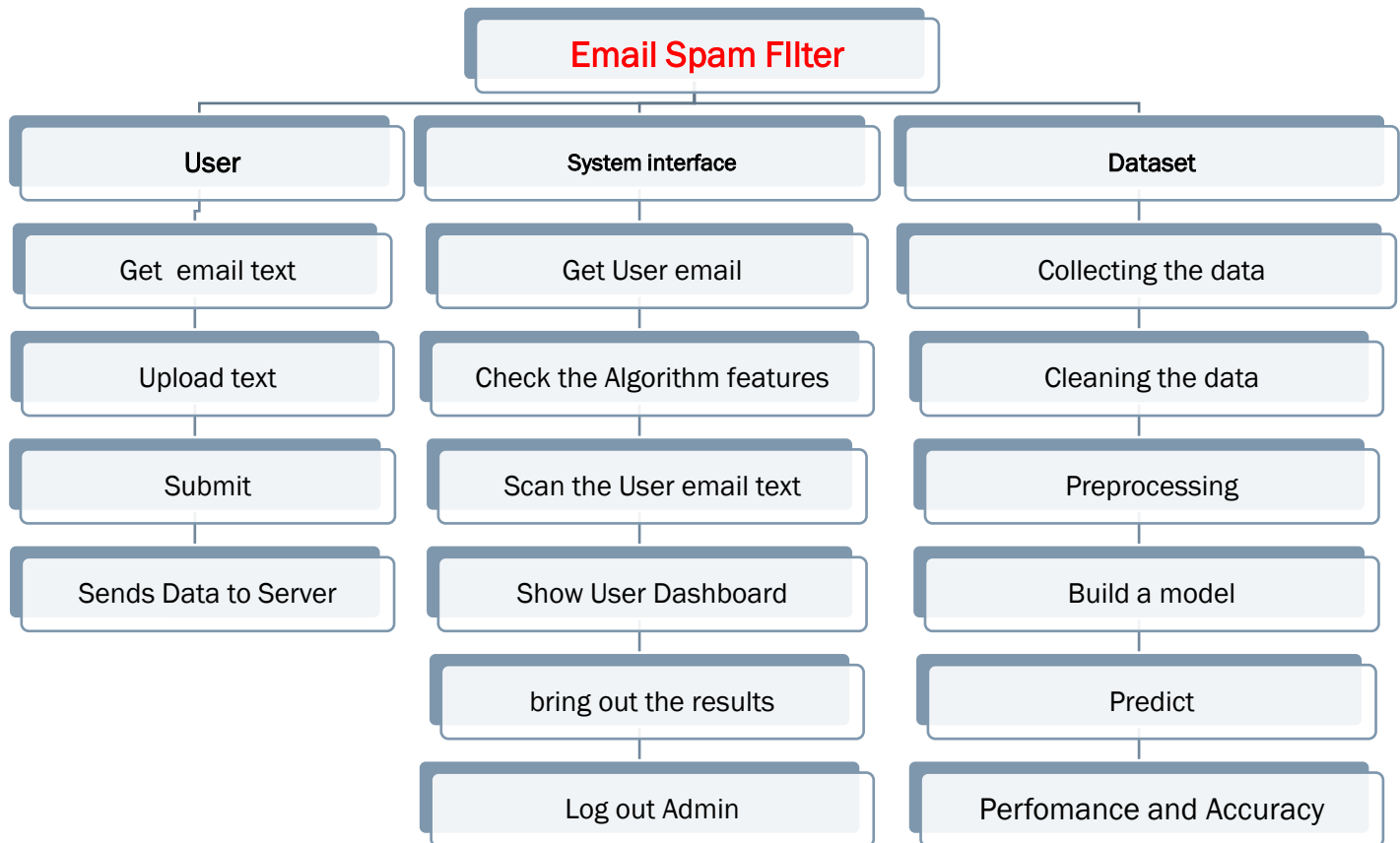
Primary Actors: User, Admin

Precondition: The user wants to logout of the system

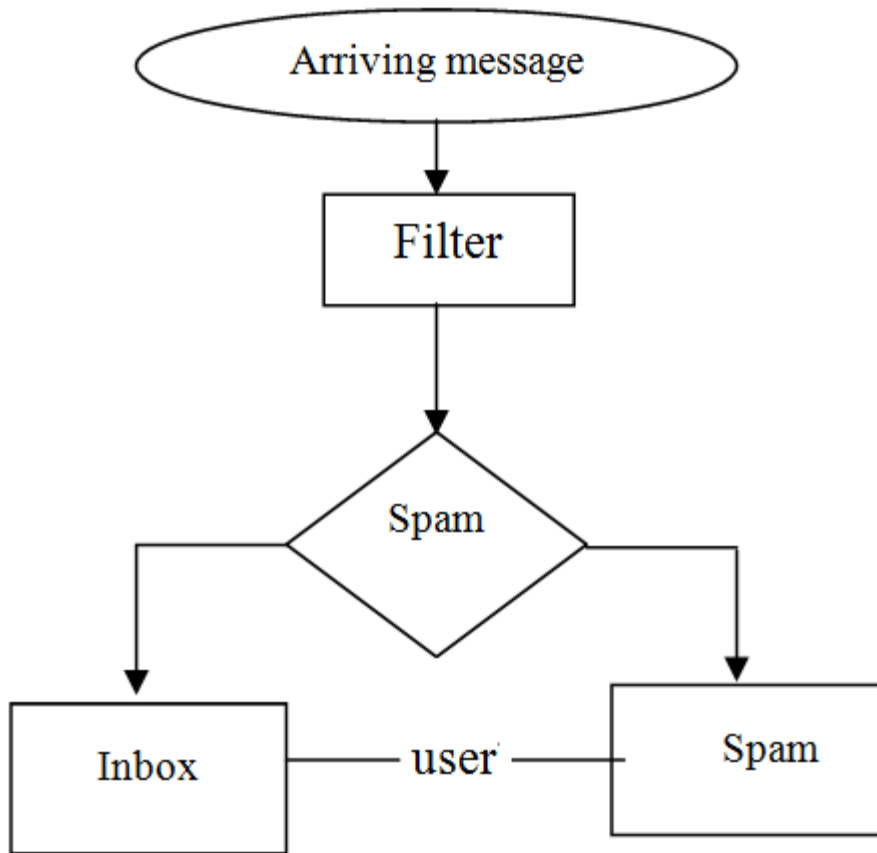
## 2.4 DATA FLOW DIAGRAM



## 2.5 FUNCTIONAL DECOMPOSITION DIAGRAM



## 2.6 STATE DIAGRAM



### A. OPERATING ENVIRONMENT

Most recently, other technology companies have also created their own notebooks that allow even more users to collaborate using Jupiter Notebooks. Google created their own platform for Jupiter Notebooks called Google Collaboratory that allows users to collaborate on projects using Google Drive. Jupiter Notebook has also inspired the creation of cloud-based coding notebooks such as Microsoft's Azure Notebook and Amazon's Sage maker Notebook, which are both compatible with, and allow collaboration on, projects within the Jupiter Notebook web-based application.

Whether you use programming simulators or popular exercises, there are many data science tools that make the process of learning how to code in a new programming language significantly easier. Jupiter Notebook is another one of those data science tools. Developed in 2014, Jupiter Notebook is an open-source collaboration technology that can be used to edit and run code using many of the most popular programming languages. Within the realm of programming, notebooks act as a one-stop-shop that allows you to work on all stages of the data analysis process on one interactive page. Commonly used amongst educators or teams, Jupiter Notebook is especially useful for working in any type of group-setting or cloud-based environment.

How do you use Jupiter Notebook?

While there are many ways to use Jupiter Notebook depending on your goals and intentions, the first step in using the notebook is getting to know the technology. One of the best parts of using Jupiter Notebook is its flexibility and versatility, as you can work with the

technology by downloading Jupiter lab, or simply opening your browser to use the notebook on the go through the original application. A tool that was made for collaboration and interactivity, Jupiter Notebook is also compatible with the most commonly used programming languages such as R, C++, Ruby, and Python.

Jupyter Notebook also gives the user access to a community of fellow users and open-source programming libraries. Once you begin using it, it is easy to find additional information and instructions on how to use the technology and integrate it into other components that may interest you. Divided into front end and back end interfaces, Jupiter Notebook not only gives users access to the outcome of their code but also assists in the process of tweaking and editing the code before it is executed

## B. DESIGN AND IMPLEMENTATION CONSTRAINTS

Classification algorithms used in machine learning utilize input training data for the purpose of predicting the likelihood or probability that the data that follows will fall into one of the predetermined categories. One of the most common applications of classification is for filtering emails into “spam” or “non-spam”, as used by today’s top email service provide

In short, classification is a form of “pattern recognition,”. Here, classification algorithms applied to the training data find the same pattern (similar number sequences, words or sentiments, and the like) in future data sets.

We will explore classification algorithms in detail, and discover how a text analysis software can perform actions like sentiment analysis - used for categorizing unstructured text by opinion polarity (positive, negative, neutral, and the like).

A classification problem in machine learning is one in which a class label is anticipated for a specific example of input data.

Problems with categorization include the following:

- Give an example and indicate whether it is spam or not.
- Identify a handwritten character as one of the recognized characters.
- Determine whether to label the current user behavior as churn.

A training dataset with numerous examples of inputs and outputs is necessary for classification from a modeling standpoint.

A model will determine the optimal way to map samples of input data to certain class labels using the training dataset. The training dataset must therefore contain a large number of samples of each class label and be suitably representative of the problem.

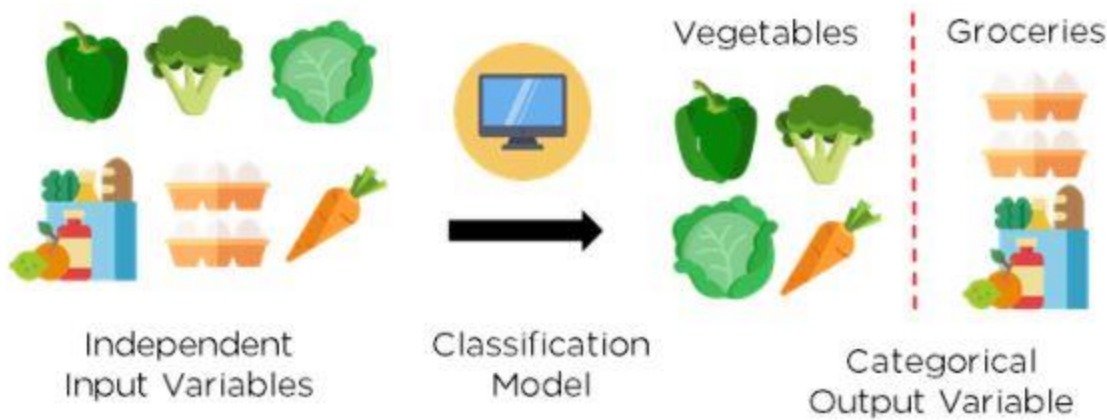
When providing class labels to a modeling algorithm, string values like "spam" or "not spam" must first be converted to numeric values. Label encoding, which is frequently used, assigns a distinct integer to every class label, such as "spam" = 0, "no spam," = 1.

There are numerous varieties of algorithms for classification in modeling problems, including predictive modeling and classification.

It is typically advised that a practitioner undertake controlled tests to determine what algorithm and algorithm configuration produces the greatest performance for a certain classification task because there is no strong theory on how to map algorithms onto issue types.

Based on their output, classification predictive modelling algorithms are assessed. A common statistic for assessing a model's performance based on projected class labels is classification accuracy. Although not perfect, classification accuracy is a reasonable place to start for many classification jobs.

Some tasks may call for a class membership probability prediction for each example rather than class labels. This adds more uncertainty to the prediction, which a user or application can subsequently interpret. The ROC Curve is a well-liked diagnostic for assessing anticipated probabilities



## C Binary Classification

Those classification jobs with only two class labels are referred to as binary classification

Examples comprise -

- Prediction of conversion (buy or not).
- Churn forecast (churn or not).
- Detection of spam email (spam or not).

Binary classification problems often require two classes, one representing the normal state and the other representing the aberrant state.

For instance, the normal condition is "not spam," while the abnormal state is "spam." Another illustration is when a task involving a medical test has a normal condition of "cancer not identified" and an abnormal state of "cancer detected."

Class label 0 is given to the class in the normal state, whereas class label 1 is given to the class in the abnormal condition.

A model that forecasts a Bernoulli probability distribution for each case is frequently used to represent a binary classification task.

The discrete probability distribution known as the Bernoulli distribution deals with the situation where an event has a binary result of either 0 or 1. In terms of classification, this indicates that the model forecasts the likelihood that an example would fall within class 1, or the abnormal state.

## D Naive Byes

Naive Bayes determines whether a data point falls into a particular category. It can be used to classify phrases or words in text analysis as either falling within a predetermined classification or

## 2.7 USER DOCUMENTATION

The primary user group of this system would ideally be for individual businesses and everyday email users. It would be implemented on a case-by-case basis. This system aims to provide a tool for an organisation to catch any incoming spam emails. This will increase the organization's security and less attacks will occur

The user is the person who will be interacting with the system to access their emails. They will interact with the system by registering. Once they registered to the application, they will enter their email address and password to login. Once the login is successful the user will be able to view their emails (inbox, sent, spam), receive emails, view their account information and logout of the application

## 2.8 ASSUMPTIONS AND DEPENDENCIES

- this software will help to decrease the cost of lost of money and materials to the extent that a user or company might have good understanding and receiving the notification for any spam email coming through their phone or Pc's
- most of the email spam messages may not have much opportunities to attack the User or clients
- 2.1.3 The product may influence to have accurate and enough information regarding to Spamming messages
- as time goes on the software will stand in depending the devices for any incompatible or disrupt of the resources




## **CHAPTER 3**

### **EXTERNAL INTERFACE REQUIREMENTS**

### 3.1 USER INTERFACES

Email Spam Filter



**PROJECT NAME**  
**EMAIL SPAM FILTER**

**SUBMITTED BY**  
**JAMIL ISAH**  
07/12/2022

Email Spam Detection Using AI

Enter email

**Predict**

## 3.2 SOFTWARE INTERFACES

transform classes into binary

```
In [16]: data['class'] = data['class'].map({'spam' : 1, 'ham' : 0}) #simple
```

🌟 Done transformation

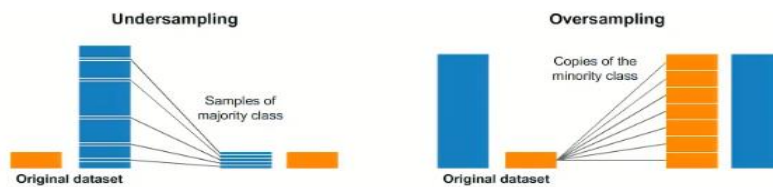
Delete Duplicated data

```
In [17]: data.drop_duplicates(keep = 'first', inplace = True)
data.shape
```

```
Out[17]: (5169, 2)
```

🌟 duplicated data solved

Apply UnderSampling



```
In [18]: counts_class_0, counts_class_1 = data['class'].value_counts()

data_c0 = data[data['class'] == 0]
data_c1 = data[data['class'] == 1]

data_under_c0 = data_c0.sample(counts_class_1, random_state = 42)

data_under = pd.concat([data_under_c0, data_c1], axis = 0)

data_under.shape
```

```
Out[18]: (1306, 2)
```

💡 Confusion matrix shows high quality classification

```
In [84]: from sklearn.model_selection import cross_val_score

Bow = count_vec.fit_transform(data_under["pre_pro_doc"])

cross_val_score(naive_bayes, Bow, data_under["class"], scoring="accuracy", cv = 20).max()
```

```
Out[84]: 0.9846153846153847
```

### Accuracy 98.46%

#Save Model

### Single Prediction

```
In [1]: from sklearn.feature_extraction.text import CountVectorizer
import pandas as pd
import numpy as np
import joblib as jb
```

```
In [2]: #Load Classifier
classifier=jb.load('spammodel.sav')
count_vec=jb.load('vec.sav')
```

## ***CHAPTER 4***

### **SYSTEM FEATURES**

## 4.1 SYSTEM FEATURE 1

### 4.1.1 Description and priority

This list outlines the top five uses for the Jupiter Notebook both within and outside of the classroom.

### 4.1.2. Programming Practice

One of the primary ways that both students and professionals can use Jupiter Notebook is as a form of practice when writing, or learning how to write, code. When writing code, there is a certain trial and error method to understanding how to fix a program that just isn't running correctly. In the past, programmers would have to write an entire program and then execute it before knowing whether or not the program actually worked. If the program did work, the computer would simply run the code as written. If it didn't, the outcome would either not be what was expected from the code or the computer would return an error message with no indicator of what portion of the program was written incorrectly.

With Jupiter Notebook, you no longer have to guess about any mistakes or missteps in your code, because Jupiter Notebook gives coders the option to practice their program in an environment that makes it significantly easier to edit and modify specific portions of code. Specifically, Jupiter Notebook gives users the option to run a section of code without executing the entire program, in order to determine if it works before moving on to writing the next line of code. By letting users run code step by step, or line by line, Jupiter Notebook provides an excellent practice space to work through a program before sending it out or sharing it with others.

### 4.1.3. Collaborating Across Projects and Tools

Another aspect of the Jupiter Notebook that stands out is the ability to collaborate on projects across a large community of users and tools. The Jupiter Notebook community was built on collaboration, as Jupiter Notebook was created to allow people to work together on data science, engineering, or other programming projects. Some features of Jupiter Notebook which make methods of collaboration easier are the fact that it supports multiple languages, is compatible with various tools and applications, and offers open-source and cloud-based programming.

### 4.1.4. Data Organization and Cleaning

One of the most tedious tasks for a data scientist is the work of cleaning and organizing data. A time-consuming process, data cleaning requires that your data be organized in such a way that the portions of the dataset that you do not want can be easily identified and removed before you can begin using the data to make suggestions or predictions. Since Jupyter Notebook is compatible with multiple tools, it can also be used to streamline this process of data organization and cleaning.

Tools such as Jupyter Innotator can be installed as part of Jupyter Notebook to streamline the process of cleaning and organizing your data. Jupyter Innotator gives users the option to create boxes and boundaries around images, as well as to use drop-down boxes to classify and categorize different parts of the dataset. The drag and drop tools within JupyterLab also make it easier to clean and organize data because users can reorder the data in a notebook and create links that make the process of searching and filtering easier and more efficient.

### 4.1.5 Data Visualization and Sharing

Once you have cleaned, organized, and analyzed your data, Jupyter Notebook can be used for data visualization and sharing. Jupyter Notebook allows users to compile all aspects of a data project in one place making it easier to show the entire process of a project to your intended audience. Through the web-based application, users can create data visualizations and other components of a project to share with others via the platform. There are multiple ways that you can share a data project both within and outside of Jupyter Notebook.

One of the primary methods of data visualization and sharing via Jupyter Notebook is through creating a compressed file or folder with all of your information and data. This file can then be shared as a link with your fellow collaborators or students. This is one of the most common ways that notebook data is shared through repositories such as GitHub. You can also work with a program like Anvil, which turns Jupyter Notebook into a shareable application. As mentioned above, you can also use integrated products and tools like Google Collaboratory to share documentation within a shared drive or platform.

### 4.1.6. Teaching Data Science Skills

Based on all of the previous uses for Jupyter Notebook, it is apparent Jupyter Notebook is primarily used to teach data science skills. It is quite common to see Jupyter Notebooks used within the classroom because it allows students to collaborate on data science and computer programming projects. In addition, it can be used to practice important data science and programming skills, such as data analysis and organization. Through the sharing and visualization component, Jupyter Notebook is an excellent tool for having students show their work on a project from start to finish, as well as learning how to identify and understand where they went right or wrong when writing code.

## 4.2 STIMULUS /RESPONSES

Machine learning algorithms have been extensively applied in the field of spam filtering. Substantial work have been done to improve the effectiveness of spam filters for classifying emails as either ham (valid messages) or spam (unwanted messages) by means of ML classifiers.

They have the ability to recognise distinctive characteristics of the contents of emails. Many significant works have been done in the field of spam filtering using techniques that does not possess the ability to adapt to different conditions; and on problems that are exclusive to some fields e.g. identifying messages that are hidden inside a stego image. Most of the machine learning algorithms used for classification of tasks were designed to learn about inactive objective groups.

The authors in [119] posited that when these algorithms are trained on data that has some data that have been poisoned by an enemy, it makes the algorithms susceptible to a number of different attacks on the reliability and accessibility of the data. As a matter of fact, manipulating as minute as 1% of the training data is enough in certain instances [120].

Though it might be strange to hear that the data supplied by an enemy is used to train a system, it does E.G. Dada et al. Heliyon 5 (2019) e01802 19 happen in some real-world systems. Examples include spam detection systems, spam connection, financial fraud, credit card fraud, and other unwelcome deeds where the earlier deeds of the enemy are a major origin of training data.

The unfortunate thing is that a good number of systems are re-trained regularly using the new instances of undesirable activities. This serves as a launching pad for attacker to launch more attacks on such system. One of the open problem that needs to be addressed is handling of threat to the security of the spam filters. Though some attempt have been made to address this problem. For example, the threat model for adaptive spam filters proposed by [121] categorises attacks based to whether they are causative or exploratory, targeted or indiscriminate, and if they are meant to interrupt reliability or accessibility.

The purpose of causative attack is to trigger error in categorisation of messages, whereas an exploratory attack aims to determine the classification of a message or set of messages. An attacks on integrity is meant to have a negative influence on the classification of spam, on the other hand, attacks on accessibility is meant to have a negative influence on the c classification of ham. The fundamental purpose of a spammer is to send spam which cannot be seized by the filter (or user) and labeled as spam. There are other potential capabilities of attack which all depend entirely on the ability to send random messages grouped as spam. A larger percentage of spam filters are nevertheless susceptible to different kinds of attack. For example, Bayes filter is susceptible to mimicry attack [120]. Naïve Bayes and AdaBoost also demonstrated endless deterioration to adversary control attack

## 4.3 FUNCTIONAL REQUIREMENTS

#	Function	Description	Importance	Tech Issues	Dependencies
1	Registration	Allows new users to set up an account	High	N/A	
2	Login	Allows user to login with Name and password	High		Function 1
3	Send Mails	Allow user to send mails	High	N/A	Function 2
4	Classify Emails from the library	stores the emails in the correct format in the SQL database	High	N/A	Email server
5	Display emails	show the result of the scanning email	High	N/A	SQL Database
6	Logout	allow the user to logout	High	N/A	The user should logout after finish scanning

# **CHAPTER 5**

## ***OTHER NONFUNCTIONAL REQUIREMENTS***



## 5.1 PERFORMANCE REQUIRMENTS

Spam filters are usually evaluated on large databases containing ham and spam messages that are publicly available to users. An example of the performance measures that are used is classification accuracy (Acc). It is the comparative number of messages rightly classified, the percentage of messages rightly classified is used as an added measure for evaluating performance of the filter. It has however been highlighted that using Accuracy as the only performance indices is not sufficient. Other performance metrics such as recall, precision and derived measures used in the field of information retrieval must be considered, so also is false positives and false negatives used in decision theory. This is very important because of the costs attached to misclassification. When a spam message is wrongly classified as ham, it gives rise to a somewhat insignificant problem, because the only thing the user need to do is to delete such message. In contrast, when a non-spam message is wrongly labeled as Spam, this is obnoxious, because it indicates the possibility of losing valuable information as a result of the filter's classification error. This is very imperative especially in settings where Spam messages are automatically deleted. Therefore, it is inadequate to evaluate the performance of any Machine Learning algorithm used in spam filter using classification accuracy exclusively. Furthermore, in a setting that is lopsided or biased where the number of spam messages utilized for testing the performance of the filter is very much higher than that of ham messages, the classifier can record a very high accuracy by concentrating

on the detection of spam messages solely. In a real world environment where there is nothing like zero probability of wrongly categorizing a ham message, it is required that a compromise be reached between the two kinds of errors, depending on the predisposition of user and the performance indicators used. The formulae for calculating the classification accuracy and classification error are depicted in Eqn. (1) and (2) below:

Assuming

NH ¼ Number of non-spam messages to be classified

NS ¼ Number of spam messages to be classified

$$\text{Classification Accuracy (Acc)} = \frac{|H \rightarrow H| + |S \rightarrow S|}{N_H + N_S} \quad (1)$$

$$\text{Classification Error (Err)} = 1 - \text{Acc} = \frac{|H \rightarrow S| + |S \rightarrow H|}{N_H + N_S} \quad (2)$$

According to fig [1,2], classification accuracy and error mutually take into account False Positive  $|H \rightarrow S|$  and False Negative  $|S \rightarrow H|$  occurrences to bear equal cost. It is necessary to point it out that disproportionate error costs is involved in spam filtering. Wrongly classifying a ham message as spam (also known as false positive event) is an expensive mistake compared to the spam message just evading the filter. Such incident is referred to as false negative event. When a legitimate e-mail is rightly classified as ham, it is called a true positive event  $|H \rightarrow H|$ . However, when a spam e-mail is rightly classified as spam, then a true negative event  $|S \rightarrow S|$  has occurred. Based on the above explanations, the false positive rate (FPR) can be defined as the ratio of ham or valid e-mails that are classified as spam.

## 5.2 SECURITY REQUIRMENTS

- The content of the emails will be stored in a SQL database.
- The user's passwords credentials which are stored in a database must be encrypted. The passwords must be hashed and salted using a suitable and secure hashing algorithm.

## 5.3 SOFTWARE QUALITY ATTRIBUTES

Many researchers and academicians have proposed different email spam classification techniques which have been successfully used to classify data into groups. These methods include probabilistic, decision tree, artificial immune system, support vector machine (SVM) artificial neural networks (ANN), and case-based technique. It has been shown in literature that it is possible to use these classification methods for spam mail filtering by using content-based filtering technique that will identify certain features (normally keywords frequently utilized in spam emails). The rate at which these features appear in emails ascertain the probabilities for each characteristic in the email, after which it is measured against the threshold value. Email messages that exceed the threshold value are classified as

spam. ANN is a non-linear model that seeks to imitate the functions of biological neural networks. It is made up of simple processing components named neurons and carries out its computational operations by processing information. Several research works have employed neural network to classify unwanted emails as spam by applying content-based filtering. These techniques decide the properties by either computing the rate of occurrence of keywords or patterns in the email messages. Literatures show that Neural Network algorithms that are utilised in email filtering attain moderate classification performance. Some of the most popular spam email classification algorithms are Multilayer Perceptron Neural Networks (MLPNNs) and Radial Base Function Neural Networks (RBFNN). Researchers used MLPNN as a classifier for spam filtering but not many of them used RBFNN for classification.

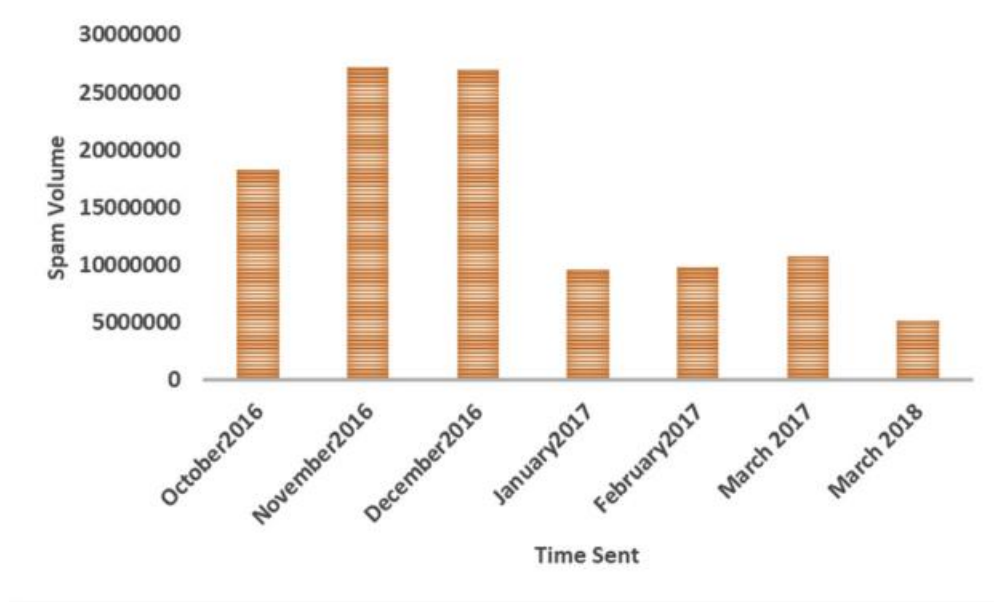


Fig. 1. The volume of spam emails 4th quarter 2016 to 1st quarter 2018.

## 5.4 BUSINESS RULES

When you create an anti-spam policy, you're actually creating a spam filter rule and the associated spam filter policy at the same time using the same name for both.

When you modify an anti-spam policy, settings related to the name, priority, enabled or disabled, and recipient filters modify the spam filter rule. All other settings modify the associated spam filter policy.

When you remove an anti-spam policy, the spam filter rule and the associated spam filter policy are removed

## 5.5 FUTURE WORK

The system will work better in detecting the Email Spam filter as required by the competition

By going throughout the project I believe that still there's a room to improve the efficiency of the systems, by God willing in the future I should add many tools and features like modernizing, updating, Upgrading and Enhancement

## 5.6 CONCLUSIONS/RECOMMENDATIONS

In this study, we reviewed machine learning approaches and their application to the field of spam filtering. A review of the state-of-the-art algorithms been applied for classification of messages as either spam or ham is provided. The attempts made by different researchers to solving the problem of spam through the use of machine learning classifiers was discussed.

The evolution of spam messages over the years to evade filters was examined. The basic architecture of email spam filter and the processes involved in filtering spam emails were looked into. The paper surveyed some of the publicly available datasets and performance metrics that can be used to measure the effectiveness of any spam filter.

The challenges of the machine learning algorithms in efficiently handling the menace of spam was pointed out and comparative studies of the machine learning techniques available in literature was done. We also revealed some open research problems associated with spam filters.

In general, the figure and volume of literature we reviewed shows that significant progress have been made and will still be made in this field. Having discussed the open problems in spam filtering, further research to enhance the effectiveness of spam filters need to be done. This will make the development of spam filters to continue to be an active research field for academicians and industry practitioners researching machine learning techniques for

effective spam filtering. My hope is that research students (my colleagues) will use this paper as a spring board for doing qualitative research in spam filtering using machine learning, deep learning and data science

**JAMIL ISAH**  
**KANO STATE NIGERIA**  
**FACULTY OF COMPUTERS AND INFORMATION**  
**LEVEL 300**  
**KAFRELSHIEK UNIVERSITY EGYPT**  
**[JAMILUISAH26@GMAIL.COM](mailto:JAMILUISAH26@GMAIL.COM)**  
**+2348061133526, +201220552809**