Seminar on special topic

Email Spam Detection

Submitted in the partial fulfillment of the requirements

for the degree of

Masters in Technology

by

Ankit Kumar

Roll No: 16030720004

Guide

Dr. Deepak Sharma

Department of Computer Engineering

K. J. Somaiya College of Engineering

(Constituent College of Somaiya Vidyavihar University)

Batch 2020 -2022

Somaiya Vidyavihar University

K. J. Somaiya College of Engineering

(Constituent College of Somaiya Vidyavihar University)

Certificate

This is to certify that the Special Topic Seminar Report on Email Spam Detection is bonafide record of the seminar work done by Ankit Kumar in the year 2021-22 under the guidance of Dr. Deepak Sharma Department of Computer Engineering in partial fulfillment of requirement for the Masters in Technology degree in Computer Engineering of Somaiya Vidyavihar University.

Dr. Deepak Sharma

Guide / Co-Guide

_____

Chairperson BOS/ACP/HOD

_____

Principal/Director/Dean

Date:

Place: Mumbai-77

Somaiya Vidyavihar University

K. J. Somaiya College of Engineering

(Constituent College of Somaiya Vidyavihar University)

Certificate of Approval of Examiners

We certify that this Special Topic Report and Seminar on Email Spam Detection is bonafide record of seminar work done by Ankit Kumar for partial fulfillment of requirement for the Masters in Technology degree in Computer Engineering of Somaiya Vidyavihar University.

_____ **Dr. Deepak Sharma**

External Examiner / Expert Guide / Internal Examiner

Date:

Place: Mumbai-77

Somaiya Vidyavihar University

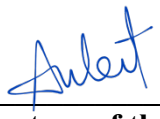
K. J. Somaiya College of Engineering

(Constituent College of Somaiya Vidyavihar University)

DECLARATION

I declare that this written report submission represents the work done based on my and / or others' ideas with adequately cited and referenced the original source. I also declare that I have adhered to all principles of academic honesty and integrity as I have not misinterpreted or fabricated or falsified any idea/data/fact/source/original work/ matter in my submission.

I understand that any violation of the above will be cause for disciplinary action by the college and may evoke the penal action from the sources which have not been properly cited or from whom proper permission is not sought.



Signature of the Student

Ankit Kumar
Name of the Student

16030720004
Roll No.

Date: 13-11-2021

Place: Mumbai-77

Abstract

Email is the most used source of official communication method for business purposes. The usage of the email continuously increases despite of other methods of communications. Automated management of emails is important in the today's context as the volume of emails grows day by day. Out of the total emails, more than 55 percent is identified as spam. This shows that these spams consume email user time and resources generating no useful output. The spammers use developed and creative methods in order to fulfil their criminal activities using spam emails, Therefore, it is vital to understand different spam email classification techniques and their mechanism. A spam-detection algorithm must find a way to filter out spam while and at the same time avoid flagging authentic messages that users want to see in their inbox. An increasing volume of unsolicited emails is bringing down the productivity dramatically. There is a need for reliable anti-spam filters to separate such messages from legitimate ones. Therefore, for this Special Topic Seminar, a spam email detection system has been implemented that employs the most commonly used Machine Learning Classification algorithms. The classifiers have been trained on a dataset from the UCI Machine Learning Repository, a well-known spam/legitimate email dataset. This system has been developed as a web portal, which would consume the email a user uploads and provide the user with a decision on whether the email is a spam email of a relevant email.

Keywords – Spam, Email, Filters, Detection, Naïve Bayes, Decision Tree, SVM.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter presents the current scenario, the need and the motivation underlying the decision of choosing this problem statement. Furthermore, it describes the scope of the project, and the hardware as well as software requirements needed.

1.1 Problem Definition:

Email is a very crucial and necessary tool for enabling rapid and cheap communication. It is a widespread medium and an essential part of the life. However, spam or unsolicited email is an inconvenience in this form of communication. They can be in the form of advertisements or similar explicit content which may contain malicious code embedded in them. A study has estimated that 70 percent of the total business emails are spam. It has also been found that rapid growth of spam email has consequences like over-flowing of users' mailboxes, consumption of bandwidth and storage space, compromising of important emails and problems to the user in terms of consuming their time to sort through all emails. Currently, there is growing interest in the field of spam classification because of the complexity that has been introduced by the spammers making it difficult to distinguish between spam mails (unsolicited emails) and ham mails (legitimate emails). For spam classification, various spam filtering methods are used. The function of a spam filter is to identify spam email and prevents it from going to the mailbox. With the help of filters, the adverse impact of spam email is mitigated and operates like a predictable and reliable tool to eliminate unwanted emails. However, there exists a small risk of misclassification or removal of legitimate emails. In this special topic seminar report, a web-based tool to classify textual emails has been presented. The methodology used for classification are most commonly used machine learning algorithms like Naïve Bayes Classifier, Decision Tree, SVM and others. Moreover, a comparative study of 11 different spam detection algorithms has also been provided.

1.2 Scope:

The scope of this special topic seminar is to provide the user a web interface. The user has the facility to provide the email in a textual format. The webpage will then provide the user with a decision on whether the provided email text is classified as spam (irrelevant) email or ham (relevant) email. The future scope of this system is multifold. An immediate development can be to acquire the user's feedback and then record the feedback for future training purpose of the machine learning algorithm. A long-term development goal in this system can be to develop a browser extension, which will automatically read emails in the user's inbox and transfer spam emails to a spam/junk or any such folder.

1.3 Hardware and Software Requirements:

Hardware requirements:

- Intel i3/i5 7 7th Gen+
- 2 GB RAM
- 500 MB free disk space

Software requirements:

- Python 3.8
- PyCharm
- Google Colab

1.4 Project Motivation:

In the today's world, email plays a very critical role in communication, but unsolicited emails hamper such communications. Being one of the major communication ways on the Internet, the emailing systems need to be protected from spam which represents unsolicited messages with serious threats to both individual users and organizations. Realizing this issue, it is an important necessity to develop a more accurate and effective spam detection system for the emailing platforms. Hence the primary motivating idea underlying this special topic seminar is to provide the user with a user interface which will assist the user in classifying emails. Therefore, the user's task of skimming through every email in their inbox is cut down. This in turn minimizes the time

and efforts spent by the user. From a developer perspective, the motivation is to gain insights and in-depth knowledge about how spam filters function.

1.5 Objectives:

- Design a web-based application for email spam detection.
- Save the users resources like time and memory.
- Gain insights in regard to the different techniques used for spam detection.

1.6 Proposed Model

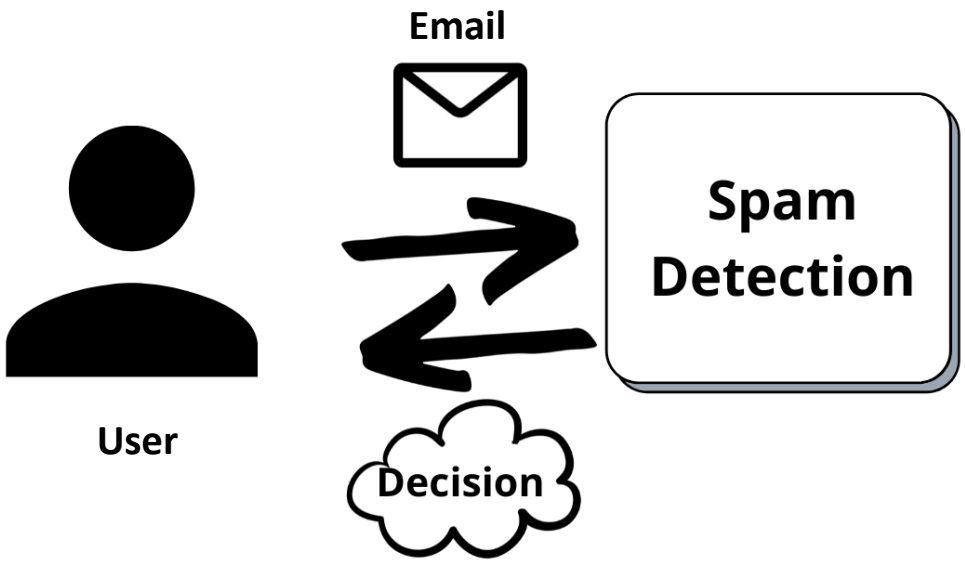


Figure 1: Proposed model

The proposed model for the planned Spam Email Detection webpage is quite straightforward in nature. The user is provided with a simple user interface for classification of emails. In this webpage the user has the facility to enter their email in a textual format. The webpage then replies with a decision as to whether the provided email text is of spam nature or ham nature. The pictorial representation of the workflow is shown in the above figure.

Chapter 2

Literature survey

This chapter presents the findings that have been gathered and the learnings that have been inferred by referring numerous technical papers based on different methods for detecting spam in textual emails. The technical papers reviewed have been recently

In order to finalize the underlying machine learning algorithm which would be employed to perform email classification, a literature survey has been conducted. In this literature survey numerous machine learning algorithms which can be used for said problem statement have been reviewed. The datasets, techniques, accuracies and other scores have taken into consideration. Also, each technical paper presented its own set of preprocessing techniques. These techniques were similar in nature across all the papers. However, due to the datasets considered by the authors being different, some unique techniques pertaining to the dataset were also discovered in this phase. The different machine learning algorithms which were considered are described below.

2.1 Naïve Bayes Classification

Upon Enron's dataset containing 6000 emails (1500 ham and 4500 spam), Naïve Bayes Classification was performed. The preprocessing techniques considered were as follows:

- URL handling.
- Lemmatization (test, tested, testing is replaced by test).
- Discarding stop-words.
- Handling special symbols.
- Word count.

In order to apply the Naïve Bayes theorem for classification, the spamicity of any token should be calculated. This is given by

$$P(S/W) = P(W/S)*P(S) / (P(W/S)*P(S)+P(W/H)*P(H)$$

where P(S/W) denotes the probability that a message is spam with the word W in it; P(S) is the overall probability that any given message is spam; P(W/S) is the probability that a particular word appears in spam messages; P(H) is the overall probability that any given message is ham; and P(W/H) is the probability that a particular word appears in ham messages. Most

Bayesian spam filtering algorithms are based on formulae that hold strictly only if the words present in the message are independent of each other, which is not always satisfied (for example, in natural languages like English the probability of finding an adjective is affected by the probability of having a noun), but it is a useful idealization, especially since the statistical correlations between the individual words are usually not known. On this basis, one can apply the Bayes' theorem to calculate the probability that a message is a spam with the bag of words in it.

P = P1P2 … PN / P1P2 … PN + (1-P1)(1-P2)…(1-PN)

where, P1, P2 … PN are the probabilities that a message is spam-knowing words P1, P2 … PN in it respectively.

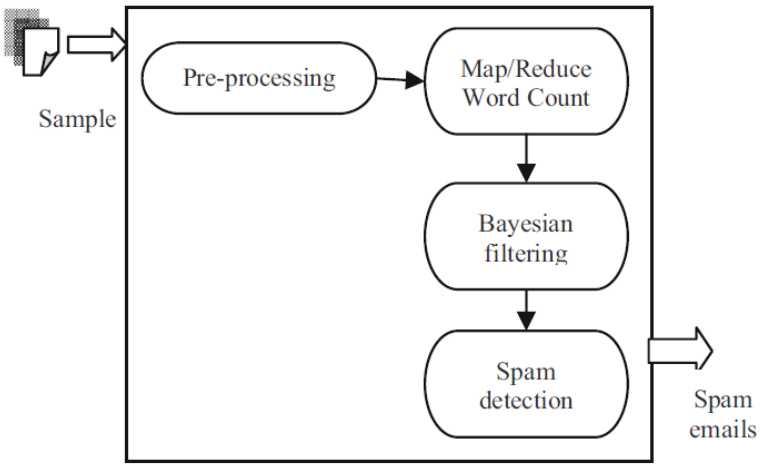


Figure 2: Naive Bayes model

The algorithm is depicted in the following way. Once the training phase is over, the knowledge is learned and a feature list (with the tokens and the probabilities that they contribute to make a message spam) is ready for the classification of the incoming emails. According to formulation, the incoming emails are examined word by word - if the word is already in the feature list, the algorithm gets its value and move forward. If the word is not in the feature list, it assigns a probability value to it; in this publication, the authors have set it as 0.5. By applying the Naïve Bayes theory, the probability that a message is spam given the bag of words in it is obtained. If the result is wrong after a checking by the user, the email would be sent back to training phase to update the feature list. The accuracy using SVMs was 89%.

2.2 Support Vector Machines:

Support Vector Machines were used for classification on a SpamAssassin dataset which contains 4000 training samples. The preprocessing techniques considered are as follows:

- Conversion to lower case.
- Removing HTML tags.
- Standardizing URLs.
- Standardizing emails.
- Standardizing numbers and special symbols.
- Word formatting.
- Removal of stop-words.

The workflow of the system using SVM is as follows:

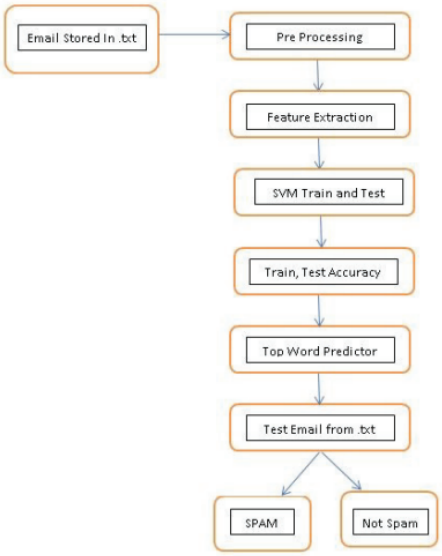


Figure 3: SVM

The fundamental point of SVM is to create a model which predicts class labels of information occurrences in the testing set which are given only the features. At the show, the support vector machines have been broadly utilized as a part of content-based hostile to spam system. A support vector machine (SVM) can be utilized when our information has totally two classes. An SVM classifies information by finding the ideal hyperplane that isolates all information purposes of one class from those of alternate class. The hyperplane for an SVM implies the one with the biggest margin between the two classes. Margin implies the maximal width of the segment parallel to the hyperplane that has no interior information points. Unlike several learning algorithms, SVM

results in sensible performances while not the necessity to include previous data. Moreover, the employment of positive definite kernel within the SVM may be taken as associate degree embedding of the input area into a high dimensional feature area wherever the classification is meted out while not exploitation expressly this feature area. Hence, the matter of selecting design for a neural network application is replaced by the matter of selecting an acceptable kernel for a Support Vector Machine. Support Vector Machine has shown power in Binary Classification. Linear separable problem like spam detection is that in which the classes can be separated using a Linear Decision Boundary. The main task of a machine learning problem is to find the best decision boundary. There can be several decision boundaries for a particular problem however the decision boundary that perfectly fits the data given as well as also is able to classify any new data is considered as a good decision boundary. As far as SVM is concerned a decision boundary having maximum margin from both classes is the best. Therefore, the values of margin m should be maximum possible for the given training data. The accuracy recorded by using SVMs was 92%.

2.3 Decision Trees:

Decision Tree algorithm was employed to perform classification on a dataset containing 4000 emails (3465 ham). The preprocessing techniques considered were as follows:

- Replace email addresses with ‘emailaddr’
- Replace URLs with ‘httpaddr’
- Replace phone numbers with constant string

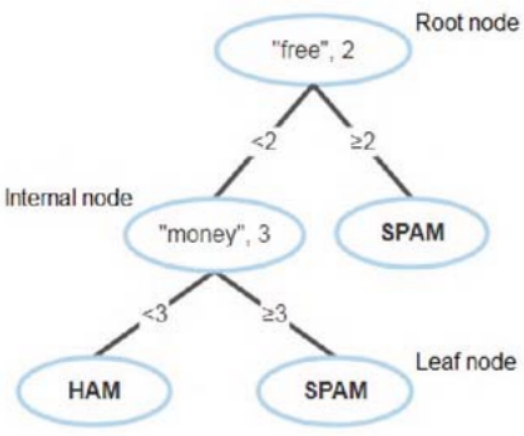


Figure 4: Decision

A decision tree consists of root node, internal nodes and leaf nodes. Internal nodes represent the conditions based on which the tree splits into branches and the leaf nodes represent possible outcomes for each path. Each node typically has two or more nodes extending from it. “When classifying an unknown instance, the unknown instance is routed down the tree according to the values of the attributes in the successive nodes and when a leaf is reached the instance is classified according to class assigned to the leaf’. The main advantage for using decision tree is that it is easy to follow and understand. The below tree presents example of a typical decision tree. Words “free” and “money” are typical spam words, and they are used as features. If the word “free” appears more than two times in an email than the email is classified as spam. Otherwise, we are asking does the email contain the word “money”. If the word “money” appears more than three times than the email is certainly spam, otherwise it is ham. The accuracy using decision trees was found by the authors to be 83%.

2.4 Comparative Study:

After reviewing 3 machine learning techniques mentioned previously, a comparative study was performed with regard to the merits, demerits, of each algorithm. Since the datasets considered for these algorithms were distinct, much emphasis should not be put onto the comparison of the actual accuracy and precision scores. The table for the comparative study has been shown below.

Table 1: Comparative study of spam detection techniques in literature survey.

Parameter	**Naive Bayes**	**SVM**	**Decision Tree**
Token combinations	Not considered	Considered but restricted to existing list of tokens only	considered
Encounter of new tokens	Considered by assigning probability as 0.5	Not used	Not used

Cleaning technique	All adequate cleaning methods followed	All adequate cleaning methods followed	Some information loss is possible

As a result of the comparative study performed, Naïve Bayes Classification has been chosen as the machine learning algorithm to be used for implementation.

Chapter 3

Logical analysis

This chapter discusses the use of Naïve Bayes Classification, in combination with another technique named TFIDF for email spam detection in the implemented webpage.

3.1 General Idea:

Given a message m = (w1, w2, ... , wn), where (w1, w2, ... , wn) is a set of unique words contained in the message. The objective is to first consider each token individually, and calculate the probability of that that token being a part of a spam email. Once this process is repeated for all tokens, the acquired probabilities of all tokens can be combined to calculate the probability of the overall email text. This combined probability can be calculated using the below formula.

$$P(spam|w1 \cap w2 \cap \ldots \cap wn) \; versus \; P(\sim spam|w1 \cap w2 \cap \ldots \cap wn)$$

In order to calculate the individual spamicity of any given token, the concept of TFIDF has been used. It is discussed as follows.

3.2 TFIDF:

TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents. It works by increasing proportionally to the number of times a word appears in a document but is offset by the number of documents that contain the word. So, words that are common in every document, such as this, what, and if, rank low even though they may appear many times, since they don't mean much to that document in particular.

$$IDF(w) = \log \frac{Total\ number\ of\ messages}{Total\ number\ of\ messages\ containing\ w}$$

For example, there are two messages in the dataset. 'please click' and 'please send contact'. TF('please') is 2. IDF('please') is log(2/2). If a word occurs a lot, it means that the word gives less information.

In this model each word has a score, which is TF(w)*IDF(w). Probability of each word is counted as:

$$P(w) = \frac{TF(w) * IDF(w)}{\sum_{\forall\ words\ x \in train\ dataset} TF(x) * IDF(x)}$$

For classifying a given message, first preprocessing takes place. For each word w in the processed message, a product of P(w|spam) is calculated. If w does not exist in the train dataset, TF(w) is considered as 0 and P(w|spam) is computed using above formula. This is then multiplied with P(spam) The resultant product is the P(spam|message). Similarly, P(ham|message) is calculated. Whichever probability among these two is greater, the corresponding tag (spam or ham) is assigned to the input message.

Chapter 4

Implementation

This chapter presents the details about implementation of the suggested prototype of the Spam Email Detection system. Starting with the methodology, the dataset, the front-end, back-end, have been discussed.

To begin with the implementation, it is important to look into the dataset that has been used, in order to ensure a thorough understanding of the implementation process.

4.1 Dataset:

The dataset used has been imported from the UCI Machine Learning Repository. It contains 4516 spam emails, along with 653 ham emails. Below is a screenshot of the .csv file.

	A	B	C	D	E	F	G	H
1	v1	v2						
2	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got						
3	ham	Ok lar... Joking wif u oni...						
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to rece						
5	ham	U dun say so early hor... U c already then say...						
6	ham	Nah I don't think he goes to usf, he lives around here though						
7	spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you u						
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.						
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as y						
10	spam	WINNER!! As a valued network customer you have been selected to receivea Â£900 prize rev						
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles wit						
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cri						
13	spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 15						
14	spam	URGENT! You have won a 1 week FREE membership in our Â£100,000 Prize Jackpot! Txt the						
15	ham	I've been searching for the right words to thank you for this breather. I promise i wont take y						
16	ham	I HAVE A DATE ON SUNDAY WITH WILL!!						
17	spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click h						
18	ham	Oh k...i'm watching here:)						
19	ham	Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.						
20	ham	Fine if thatÃ’s the way u feel. ThatÃ’s the way its gota b						

Figure 5: Dataset

4.2 Methodology:

The methodology can be split into 2 phases, namely the training phase and the testing phase. A pictorial representation of the 2 phases is shown below.

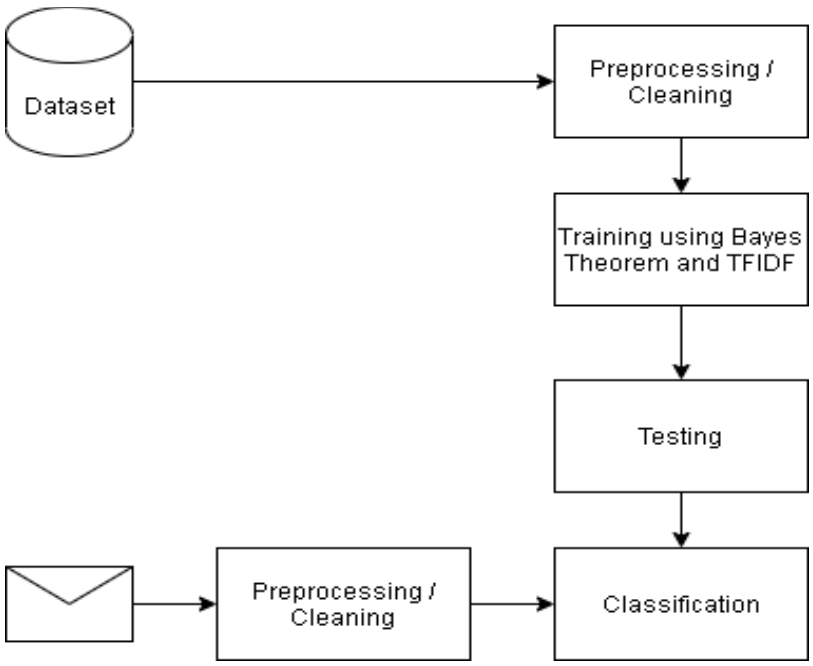


Figure 6: Methodology

4.2.1 Steps performed in training phase:

1. Importing the dataset.
2. Perform cleaning operations on emails and extract tokens.
3. Compute and store the probabilities of individual tokens present in the dataset.

4.2.2 Steps performed in testing phase:

1. Clean and tokenize the email provided by the user / incoming from the testing split.
2. Use the probabilities of the stored token to compute the overall probability of the email.
3. Using the overall probability, classify the email into one of the 2 categories: ham/spam.

4.3 Preprocessing, training and testing:

Once the methodology for implementing the machine learning algorithm has been formulated, the implementation of the functions to be carried out in the individual phases can begin. The implementation of all the functions has been achieved using Python (3.8). In certain nuanced functions, the use of external Python libraries such as Natural Language Toolkit (nltk). Moreover, the Pandas library has also been employed to achieve the initial task of importing the dataset into

dataframes. The primary priority in both the phases has been to minimize the use of inbuilt library functions.

4.3.1 Preprocessing:

As preprocessing is a common feature in both phases, the implemented function can be reused. As per the preprocessing / cleaning techniques discussed in the literature survey, cleaning of emails has been performed.

3. Data Preprocessing

- Lower case
- Tokenization
- Removing special characters
- Removing stop words and punctuation
- Stemming

```
[ ]  import string
     import nltk
     nltk.download("stopwords")
     from nltk.corpus import stopwords
     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)
```

Figure 5: Preprocessing block

4.3.2 Training:

As mentioned in the logical analysis, the objective of the training phase is to assign each token with a probability of it being spam. To achieve the same, the concept of Term Frequency Inverse Document Frequency (TFIDF) has been used. Below is a snapshot of one of the code blocks of the training phase.

4. Model Building

```
[ ] from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
    cv = CountVectorizer()
    tfidf = TfidfVectorizer(max_features=3000)

[ ] X = tfidf.fit_transform(df['transformed_text']).toarray()

[ ] #from sklearn.preprocessing import MinMaxScaler
    #scaler = MinMaxScaler()
    #X = scaler.fit_transform(X)

[ ] # appending the num_character col to X
    #X = np.hstack((X,df['num_characters'].values.reshape(-1,1)))

[ ] X.shape

    (5169, 3000)

[ ] y = df['target'].values

[ ] from sklearn.model_selection import train_test_split

[ ] X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)

[ ] from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
    from sklearn.metrics import accuracy_score,confusion_matrix,precision_score

[ ] gnb = GaussianNB()
    mnb = MultinomialNB()
    bnb = BernoulliNB()
```

Figure 8: Training the dataset.

4.3.3 Testing:

The testing phase is classifying a given email into one of the 2 categories. After classification, the category is confirmed with the given category in the dataset.

The classification module is reused in the web application wherein the user is presented with a decision. The results of the training and testing phase will be discussed in the latter part. Below is a screenshot of one of the classification modules.

```
[ ] mnb.fit(X_train,y_train)
    y_pred2 = mnb.predict(X_test)
    print(accuracy_score(y_test,y_pred2))
    print(confusion_matrix(y_test,y_pred2))
    print(precision_score(y_test,y_pred2))
```

```
0.9729206963249516
[[896   0]
 [ 28 110]]
1.0
```

```
[ ] 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 -  0.9748549323017408
Precision -  0.9666666666666667
For  KN
Accuracy -  0.9052224371373307
Precision -  1.0
For  NB
Accuracy -  0.9729206963249516
Precision -  1.0
For  DT
Accuracy -  0.9323017408123792
Precision -  0.8333333333333334
For  LR
Accuracy -  0.9574468085106383
Precision -  0.9519230769230769
For  RF
Accuracy -  0.971953578336557
Precision -  0.9739130434782609
For  AdaBoost
Accuracy -  0.9642166344294004
Precision -  0.9316239316239316
For  BgC
Accuracy -  0.9545454545454546
Precision -  0.8582677165354331
For  ETC
Accuracy -  0.9777562862669246
Precision -  0.9831932773109243
For  GBDT
Accuracy -  0.9487427466150871
Precision -  0.9292929292929293
For  xgb
Accuracy -  0.9468085106382979
Precision -  0.946236559139785
```

```
[ ]  import pickle
     pickle.dump(tfidf,open('vectorizer.pkl','wb'))
     pickle.dump(mnb,open('model.pkl','wb'))
```

Figure 6: Testing block

4.4 Back-end:

The backend has been developed using the NLTK framework provided by Python. The Python application can be initialized with the straightforward '*streamlit run app.py*' command.

```
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)

tfidf = pickle.load(open('vectorizer.pkl','rb'))
model = pickle.load(open('model.pkl','rb'))
```

Figure 10: Backend Framework

4.5 Front-end:

The front-end has been designed using the streamlit framework.

```python
tfidf = pickle.load(open('vectorizer.pkl','rb'))
model = pickle.load(open('model.pkl','rb'))

st.title("Email/SMS Spam Classifier")
st.title("by Ankit Kumar | 16030720004 | KJSCE | S.Y. M.Tech")
input_sms = st.text_area("Enter the message")

if st.button('Predict'):

    # 1. preprocess
    transformed_sms = transform_text(input_sms)
    # 2. vectorize
    vector_input = tfidf.transform([transformed_sms])
    # 3. predict
    result = model.predict(vector_input)[0]
    # 4. Display
    if result == 1:
        st.header("Spam")
    else:
        st.header("Not Spam")
```

Figure 11: Front-end using streamlit.

Below are the screenshots of the components present in the web application.

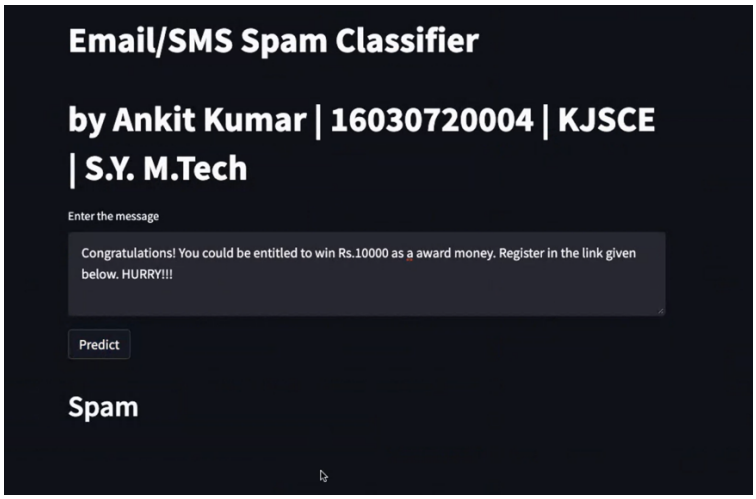


Figure 12: Spam Detected

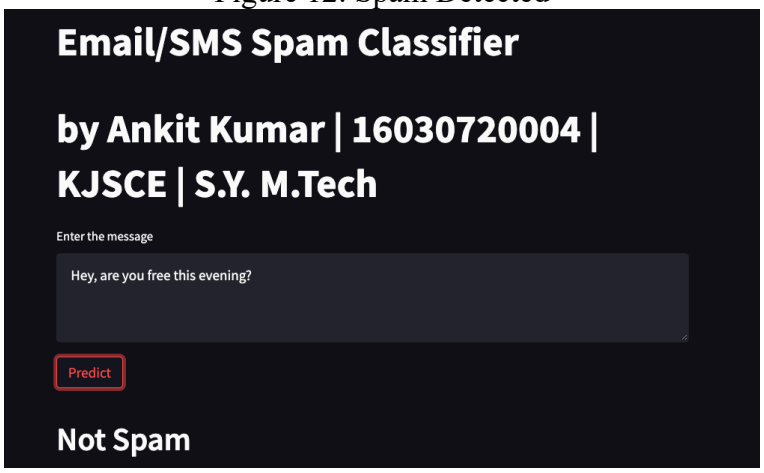


Figure 13: Not Spam Detected

Chapter 5

Results

This chapter discusses the results of the Naïve Bayes Classifier algorithm used for spam email detection. Moreover, the interpretation of results has been discussed.

Naïve Bayes Classification has been used to perform spam email detection. As this algorithm has been developed with minimal use of external inbuilt library functions, it is important to perform a comparative study with another method to implement the same algorithm. For this, the famous Python library 'scikit-learn' has been considered. This library has an inbuilt model for Naïve Bayes Classification which can be initialized by creating an object of '*MultinomialNB()*'. However, this model does not make use of the TFIDF concept, which is present in the proposed system. The same dataset has been used to train both the models. The comparison table for all the methods is shown below.

	Algorithm	Accuracy	Precision	Accuracy_scaling_x	Precision_scaling_x	Accuracy_scaling_y	Precision_scaling_y	Accuracy_num_chars	Precision_num_chars
0	KN	0.905222	1.000000	0.905222	1.000000	0.905222	1.000000	0.905222	1.000000
1	NB	0.972921	1.000000	0.972921	1.000000	0.972921	1.000000	0.972921	1.000000
2	ETC	0.977756	0.983193	0.977756	0.983193	0.977756	0.983193	0.977756	0.983193
3	RF	0.971954	0.973913	0.971954	0.973913	0.971954	0.973913	0.971954	0.973913
4	SVC	0.974855	0.966667	0.974855	0.966667	0.974855	0.966667	0.974855	0.966667
5	LR	0.957447	0.951923	0.957447	0.951923	0.957447	0.951923	0.957447	0.951923
6	xgb	0.946809	0.946237	0.946809	0.946237	0.946809	0.946237	0.946809	0.946237
7	AdaBoost	0.964217	0.931624	0.964217	0.931624	0.964217	0.931624	0.964217	0.931624
8	GBDT	0.948743	0.929293	0.948743	0.929293	0.948743	0.929293	0.948743	0.929293
9	BgC	0.954545	0.858268	0.954545	0.858268	0.954545	0.858268	0.954545	0.858268
10	DT	0.932302	0.833333	0.932302	0.833333	0.932302	0.833333	0.932302	0.833333

Table 2: Comparison between different ML algorithms

As it can be seen, the accuracy and precision of the Naïve Bayes algorithm is the highest compared to the others. An important detail to be noted here is that the precision parameter is equal for both techniques. Precision is defined as the ratio of true positives to the sum of true positives as well as true negatives. In this particular use case of spam detection, false positives have a huge impact on the accuracy of the algorithm. This is because, if a relevant/ham email is incorrectly classified as spam by the classifier, it may be deleted by the user. This would result in loss of relevant information. Hence, the precision parameter has also been computed which is equal using both techniques.

Chapter 6

Future scope

This chapter discusses the future scope of the proposed system. Updates in regard to front-end, back-end, machine learning algorithm, dataset have been taken into consideration.

The proposed system is a web application which assists the user in decluttering their inbox. However, improvements with respect to the UI, functionality, dataset are possible. Some of these major updates are as follows:

- A browser extension for PCs can be considered. This browser extension will access the user's emails with their prior permission when the user opens their inbox. Hence, the process of classification and moving the emails to a spam folder can also be automated.
- Currently the machine learning algorithm has been trained with a fixed dataset. However, this dataset may or may not resonate with the spam emails present in the user's inbox. Hence, a functionality update which would allow the user to upload a dataset of their choice for training, can also be considered.
- A minor update can be to request the user's feedback whenever a spam email is detected by the system. If the decision provided by the system has been incorrect, then with the user's permission, the email text can be automatically added to the training dataset with its correct label.

Chapter 7

Conclusion

This chapter discusses the conclusions drawn after developing the prototype of the proposed system.

In today's world, communication through email has become one of the cheapest and easy ways for the official and business users due to easy availability of internet access. Most of the people prefer to use email to share important information and to maintain their official records. However, like the two sides of coin, many people misuse this easy way of communication by sending unwanted & useless bulk emails to others. These unwanted emails are spam emails that affect the normal user to face problems. Hence, there is the need of an automated system which assists the user in deciding about spam emails.

In this report, a prototype of a Spam Email Detection system has been discussed and developed. After conducting the literature survey, Naïve Bayes Classification has been selected as the machine learning algorithm to perform spam detection. By researching various algorithms and their preprocessing techniques, cleaning techniques have been employed in the proposed system. The back-end of the system has been designed using Python. The front-end has been designed using the Streamlit framework (Python). This system has been developed as a webpage. Hence, it is platform independent with minimal resources required on the client side.

References

[1] M. RAZA, N. D. Jayasinghe and M. M. A. Muslam, "A Comprehensive Review on Email Spam Classification using Machine Learning Algorithms," 2021 International Conference on Information Networking (ICOIN), 2021, pp. 327-332, doi: 10.1109/ICOIN50884.2021.9334020.

[2] S. Nandhini and J. Marseline K.S., "Performance Evaluation of Machine Learning Algorithms for Email Spam Detection," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-4, doi: 10.1109/ic-ETITE47903.2020.312.

[3] S. Suryawanshi, A. Goswami and P. Patil, "Email Spam Detection : An Empirical Comparative Study of Different ML and Ensemble Classifiers," 2019 IEEE 9th International Conference on Advanced Computing (IACC), 2019, pp. 69-74, doi: 10.1109/IACC48062.2019.8971582.

[4] M. Singh, R. Pamula and S. k. shekhar, "Email Spam Classification by Support Vector Machine," 2018 International Conference on Computing, Power and Communication Technologies (GUCON), 2018

[5] I. Čavor, "Decision Tree Model for Email Classification," 2021 25th International Conference on Information Technology (IT), 2021, pp. 1-4, doi: 10.1109/IT51528.2021.9390143.

[6] W. You, K. Qian, D. Lo, P. Bhattacharya, M. Guo and Y. Qian, "Web Service-Enabled Spam Filtering with Naïve Bayes Classification," 2015 IEEE First International Conference on Big Data Computing Service and Applications