# AN INTERNSHIP REPORT ON

# ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted in accordance with the requirement for the degree of

Bachelor of Degree

Under the Faculty Guidenship of

**Radha Krishna**

Department of

**Computer Science**

**Pragati Degree college**

Submitted by:

Name :

Reg.No :

Department of:

PRAGATI DEGREE COLLEGE

# Instructions to Students

Please read the detailed Guidelines on Internship hosted on the website of AP State Council of Higher Education **https://apsche.ap.gov.in**

1. It is mandatory for all the students to complete Semester internship either in V Semester or in VI Semester.
2. Every student should identify the organization for internship in consultation with the College Principal/the authorized person nominated by the Principal.
3. Report to the intern organization as per the schedule given by the College. You must make your Own arrangement transportation to reach the organization.
4. You should maintain punctuality in attending the internship. Daily attendance is compulsory.
5. You are expected to learn about the organization, policies, procedures, and processes by interacting with the people working in the organization and by consulting the supervisor attached to the interns.
6. While you are attending the internship, follow the rules and regulations of the intern organization.
7. While in the intern organization, always wear your College Identity Card.
8. If your College has a prescribed dress as uniform, wear the uniform daily, as you attend to your assigned duties.
9. You will be assigned a Faculty Guide from your College. He/She will be creating a WhatsApp group with your fellow interns. Post your daily activity done and/or any difficulty you encounter during the internship.
10. Identify five or more learning objectives in consultation with your Faculty Guide. These learning objectives can address:
    a. Data and Information you are expected to collect about the organization and/or industry.
    b. Job Skills you are expected to acquire.
    c. Development of professional competencies that lead to future career success.

11. Practice professional communication skills with team members, co-interns, and your supervisor. This includes expressing thoughts and ideas effectively through oral, written, and non-verbal communication, and utilizing listening skills.

12. Be aware of the communication culture in your work environment. Follow up and communicate regularly with your supervisor to provide updates on your progress with work assignments.

13. Never be hesitant to ask questions to make sure you fully understand what you need to do your work and to contribute to the organization.

14. Be regular in filling up your Program Book. It shall be filled up in your own handwriting. Add additional sheets wherever necessary.

15. At the end of internship, you shall be evaluated by your Supervisor of the intern organization.

16. There shall also be evaluation at the end of the internship by the Faculty Guide and the Principal.

17. Do not meddle with the instruments/equipment you work with.

18. Ensure that you do not cause any disturbance to the regular activities of the intern organization.

19. Be cordial but not too intimate with the employees of the intern organization and your fellow interns.

20. You should understand that during the internship programme, you are the ambassador of your College, and your behaviour during the internship programme is of utmost importance.

21. If you are involved in any discipline related issues, you will be withdrawn from the internship programme immediately and disciplinary action shall be initiated.

22. Do not forget to keep up your family pride and prestige of your College.

# Student’s Declaration

I,______________________a student of ___________________Program,

Reg. No.__of the Department of_________________________

College do hereby declare that I have completed the mandatory internship

From to_______________in ____________under the Faculty

Guideship of______, Department of _________________________

*(Signature and Date)*

# Official Certification

This is to certify that_______________ Reg. No._________________has completed his/her Internship in _________________________under my supervision as a paet of partial fulfillment of the requirement for the Degree of ____________________________________________in the Department of____________________________________________

This is accepted for evaluation.

*(Signature with Date and Seal)*

## Endorsements

Faculty Guide

Head of the Department

Principal

# Certificate From Intern Organization

This is to certify that ______________________________

Reg.no ____________________of____________________________

underwent internship in _________________________

To_________________________

The overall performance of the intern during his/her

internship is found to be ____________________

Authorized Signatory with Date and Seal

**Title: ”Spam Detection using Machine Learning and Natural Language Processing”**

**TABLE OF CONTENTS**

## 1. Abstract

Now-a-days communication plays a major role in everything be it professional or personal. Email communication service is being used extensively because of its free use services, low-cost operations, accessibility, and popularity. Emails have one major security flaw that is anyone can send an email to anyone just by getting their unique user id. This security flaw is being exploited by some businesses and ill-motivated persons for advertising, phishing, malicious purposes, and finally fraud. This produces a kind of email category called SPAM. Spam refers to any email that contains an advertisement, unrelated and frequent emails. These emails are increasing day by day in numbers. Studies show that around 55 percent of all emails are some kind of spam. A lot of effort is being put into this by service providers. Spam is evolving by changing the obvious markers of detection. Moreover, the spam detection of service providers can never be aggressive with classification because it may cause potential information loss to incase of a misclassification. To tackle this problem we present a new and efficient method to detect spam using machine learning and natural language processing. A tool that can detect and classify spam. In addition to that, it also provides information regarding the text provided in a quick view format for user convenience.

# ACTIVITY LOG FOR THE WEEK

| Day<br>&<br>Date | Brief description<br>of the<br>Daily activity | Learning<br>Outcome | Person In-<br>Charge<br>Signature |
|---|---|---|---|
| Day – 1 | | | |
| Day - 2 | | | |
| Day – 3 | | | |
| Day – 4 | | | |
| Day – 5 | | | |
| Day –6 | | | |

# WEEKLY REPORT

**Objective of the Activity Done:**

**Detailed Report:**

# 2. Introduction

## 2.1 Introduction

Today, Spam has become a major problem in communication over internet. It has been accounted that around 55% of all emails are reported as spam and the number has been growing steadily. Spam which is also known as unsolicited bulk email has led to the increasing use of email as email provides the perfect ways to send the unwanted advertisement or junk newsgroup posting at no cost for the sender. This chances has been extensively exploited by irresponsible organizations and resulting to clutter the mail boxes of millions of people all around the world.

Spam has been a major concern given the offensive content of messages, spam is a waste of time. End user is at risk of deleting legitimate mail by mistake. Moreover, spam also impacted the economical which led some countries to adopt legislation.

Text classification is used to determine the path of incoming mail/message either into inbox or straight to spam folder. It is the process of assigning categories to text according to its content. It is used to organized, structures and categorize text. It can be done either manually or automatically. Machine learning automatically classifies the text in a much faster way than manual technique. Machine learning uses pre-labelled text to learn the different associations between pieces of text and it output. It used feature extraction to transform each text to numerical representation in form of vector which represents the frequency of word in predefined dictionary.

Text classification is important to structure the unstructured and messy nature of text such as documents and spam messages in a cost-effective way. Machine learning can make more accurate precisions in real-time and help to improve the manual slow process to much better and faster analysing big data. It is important especially to a company to analyse text data, help inform business decisions and even automate business processes.

In this project, machine learning techniques are used to detect the spam message of a mail. Machine learning is where computers can learn to do something without the need to explicitly program them for the task.

It uses data and produce a program to perform a task such as classification. Compared to knowledge engineering, machine learning techniques require messages that have been successfully pre-classified. The pre-classified messages make the training dataset which will be used to fit the learning algorithm to the model in machine learning studio. A combination of algorithms are used to learn the classification rules from messages. These algorithms are used for classification of objects of different classes. These algorithms are provided with pre labelled data and an unknown text. After learning from the prelabelled data each of these algorithms predict which class the unknown text may belong to and the category predicted by majority is considered as final.

### 2.2 Summary

From various studies, we can take that for various types of data various models performs better. Naïve Bayes, random forest, SVM, logistic regression are some of the most used algorithms in spam detection and classification.

## 3. Objectives and Scope

### 3.1 Problem Statement

Spammers are in continuous war with Email service providers. Email service providers implement various spam filtering methods to retain their users, and spammers are continuously changing patterns, using various embedding tricks to get through filtering. These filters can never be too aggressive because a slight misclassification may lead to important information loss for consumer. A rigid filtering method with additional reinforcements is needed to tackle this problem.

### 3.2 Objectives

The objectives of this project are

i. To create a ensemble algorithm for classification of spam with highest possible accuracy.
ii. To study on how to use machine learning for spam detection.
iii. To study how natural language processing techniques can be implemented in spam detection.
iv. To provide user with insights of the given text leveraging the created algorithm and NLP.

### 3.3 Project Scope

This project needs a coordinated scope of work.

i. Combine existing machine learning algorithms to form a better ensemble algorithm.
ii. Clean, processing and make use of the dataset for training and testing the model created.
iii. Analyse the texts and extract entities for presentation.

### 3.4 Limitations

This Project has certain limitations.

i. This can only predict and classify spam but not block it.
ii. Analysis can be tricky for some alphanumeric messages and it may struggle with entity detection.
iii. iii. Since the data is reasonably large it may take a few seconds to classify and anlayse the message.

## 4. Experimentation and Methods

### 4.1 Introduction

This chapter will explain the specific details on the methodology being used to develop this project. Methodology is an important role as a guide for this project to make sure it is in the right path and working as well as plan. There is different type of methodology used in order to do spam detection and filtering. So, it is important to choose the right and suitable methodology thus it is necessary to understand the application functionality itself.

### 4.2 System Architecture

The application overview has been presented below and it gives a basic structure of the application.

The UI, Text processing and ML Models are the three important modules of this project. Each Module’s explanation has been given in the later sections of this chapter. A more complicated and detailed view of architecture is presented in the workflow section.

## ACTIVITY LOG FOR THE WEEK

| Day<br>&<br>Date | Brief description<br>of the<br>Daily activity | Learning<br>Outcome | Person In-<br>Charge<br>Signature |
|---|---|---|---|
| Day – 1 | | | |
| Day - 2 | | | |
| Day – 3 | | | |
| Day – 4 | | | |
| Day – 5 | | | |
| Day –6 | | | |

# WEEKLY REPORT

**Objective of the Activity Done:**

**Detailed Report:**

### 4.3 Modules and Explanation

The Application consists of three modules.

i. UI
ii. Machine Learning
iii. Data Processing

**UI Module**

a. This Module contains all the functions related to UI(user interface).
b. The user interface of this application is designed using Streamlit library from python based packages.
c. The user inputs are acquired using the functions of this library and forwarded to data processing module for processing and conversion.
d. Finally the output from ML module is sent to this module and from this module to user in visual form.

**Machine Learning Module**

a. This module is the main module of all three modules.
b. This modules performs everything related to machine learning and results analysis.
c. Some main functions of this module are
    i. Training machine learning models.
    ii. ii. Testing the model
    iii. Determining the respective parameter values for each model.
    iv. Key-word extraction.
    v. Final output calculation
d. The output from this module is forwarded to UI for providing visual response to user.

**Data Processing Module**

a. The raw data undergoes several modifications in this module for further process.
b. Some of the main functions of this module includes
   i. Data cleaning
   ii. Data merging of datasets
   iii. Text Processing using NLP
   iv. Conversion of text data into numerical data(feature vectors).
   v. Splitting of data.
   vi. All the data processing is done using Pandas and NumPy libraries.
   vii. Text processing and text conversion is done using NLTK and scikit-learn libraries.

### 4.4 Requirements

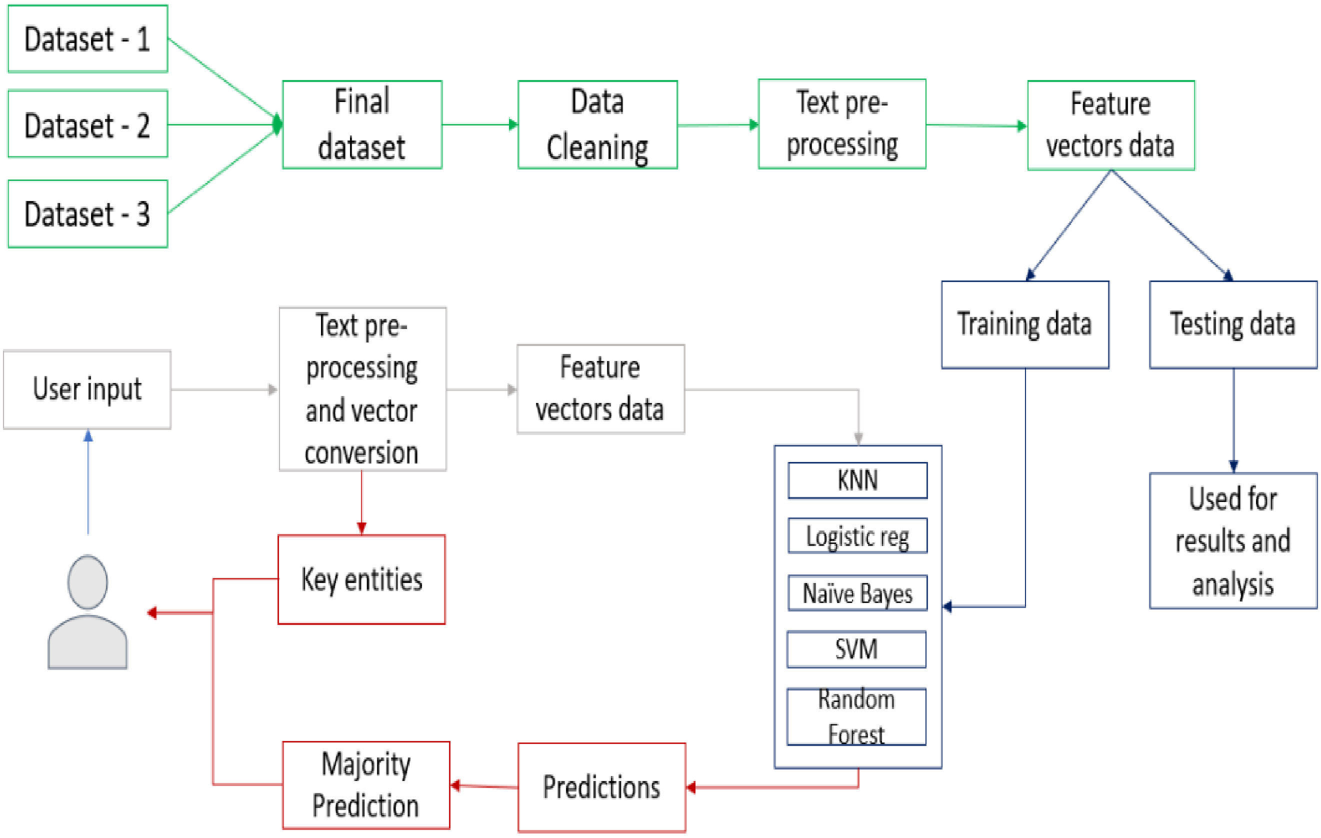
**Hardware Requirements**

- PC/Laptop
- Ram – 8 Gig
- Storage – 100-200 Mb

**Software Requirements**

- OS – Windows 7 and above
- Code Editor – Pycharm, VS Code, Built in IDE
- Anaconda environment with packages nltk, numpy, pandas, sklearn, tkinter, nltk data.
- Supported browser such as chrome, firefox, opera etc..

### 4.5 WorkFlow



In the above architecture, the objects depicted in Green belong to a module called Data Processing. It includes several functions related to data processing, natural Language Processing. The objects depicted in Blue belong to the Machine Learning module. It is where everything related to ML is embedded. The red objects represent final results and outputs.

### 4.5.1 Data Collection and Description

- Data plays an important role when it comes to prediction and classification, the more the data the more the accuracy will be.
- The data used in this project is completely open-source and has been taken from various resources like Kaggle and UCI
- For the purpose of accuracy and diversity in data multiple datasets are taken. 2 datasets containing approximately over 12000 mails and their labels are used for training and testing the application.
- 6000 spam mails are taken for generalisation of data and to increase the accuracy.

### 4.5.2 Data Processing

#### 4.5.2.1 Overall data processing It consists of two main tasks

- Dataset cleaning It includes tasks such as removal of outliers, null value removal, removal of unwanted features from data.
- Dataset Merging After data cleaning, the datasets are merged to form a single dataset containing only two features(text, label). Data cleaning, Data Merging these procedures are completely done using Pandas library.

#### 4.5.2.2 Textual data processing

- Tag removal Removing all kinds of tags and unknown characters from text using regular expressions through Regex library.
- Sentencing, tokenization Breaking down the text(email/SMS) into sentences and then into tokens(words). This process is done using NLTK pre-processing library of python.
- Stop word removal Stop words such as of , a ,be , … are removed using stopwords NLTK library of python.
- Lemmatization Words are converted into their base forms using lemmatization and pos-tagging This process gives key-words through entity extraction. This process is done using chunking in regex and NLTK lemmatization.
- Sentence formation The lemmatized tokens are combined to form a sentence. This sentence is essentially a sentence converted into its base form and removing stop words. Then all the sentences are combined to form a text.
- While the overall data processing is done only to datasets, the textual processing is done to both training data, testing data and also user input data.

#### 4.5.2.3 Feature Vector Formation

- The texts are converted into feature vectors(numerical data) using the words present in all the texts combined
- This process is done using countvectorization of NLTK library.
- The feature vectors can be formed using two language models Bag of Words and Term Frequency-inverse Document Frequency.

#### 4.5.2.3.1 Bag of Words

Bag of words is a language model used mainly in text classification. A bag of words represents the text in a numerical form. The two things required for Bag of Words are

- A vocabulary of words known to us.
- A way to measure the presence of words.

Ex: a few lines from the book “A Tale of Two Cities” by Charles Dickens.

“ It was the best of times,

it was the worst of times,

it was the age of wisdom,

it was the age of foolishness, ”

The unique words here (ignoring case and punctuation) are:

[ “it”, “was”, “the”, “best”, “of”, “times”, “worst”,“age”, “wisdom”, “foolishness” ]

The next step is scoring words present in every document.

After scoring the four lines from the above stanza can be represented in vector form as

“It was the best of times“ = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0] "

it was the worst of times" = [1, 1, 1, 0, 1, 1, 1, 0, 0, 0] "

it was the age of wisdom" = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0] "

it was the age of foolishness"= [1, 1, 1, 0, 1, 0, 0, 1, 0, 1]

This is the main process behind the bag of words but in reality the vocabulary even from a couple of documents is very large and words repeating frequently and important in nature are taken and remaining are removed during the text processing stage.

#### 4.5.2.3.2 Term Frequency-inverse document frequency

Term frequency-inverse document frequency of a word is a measurement of the importance of a word. It compares the repentance of words to the collection of documentsand calculates the score.

Terminology for the below formulae:

t – term(word)

d – document(set of words)

N – count of documents

The TF-IDF process consists of various activities listed below.

i) Term Frequency The count of appearance of a particular word in a document is called term frequency $\boldsymbol{tf(t,\ d) = count\ of\ t\ in\ d/ number\ of\ words\ in\ d}$

ii) Document Frequency Document frequency is the count of documents the word was detected in. We consider one instance of a word and it doesn't matter if the word is present multiple times. $\boldsymbol{df(t) = occurrence\ of\ t\ in\ documents}$

iii) Inverse Document Frequency

- IDF is the inverse of document frequency.
- It measures the importance of a term t considering the information it contributes. Every term is considered equally important but certain terms such as (are, if, a, be, that, ..) provide little information about the document. The inverse document frequency factor reduces the importance of words/terms that has highe recurrence and increases the importance of words/terms that are rare.

$$\boldsymbol{idf(t) = N/df}$$

Finally, the TF-IDF can be calculated by combining the term frequency and inverse document frequency.

$$\boldsymbol{tf\_idf(t,\ d) = tf(t,\ d) * \log(N/(df + 1))}$$

the process can be explained using the following example:

"Document 1 It is going to rain today.

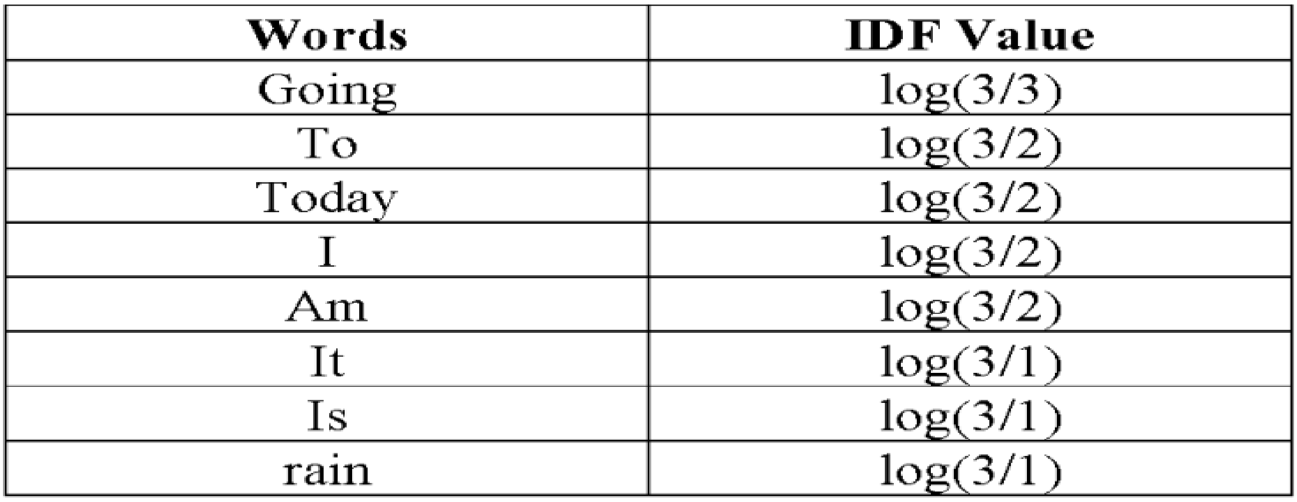
Document 2 Today I am not going outside.

Document 3 I am going to watch the season premiere."

The Bag of words of the above sentences is

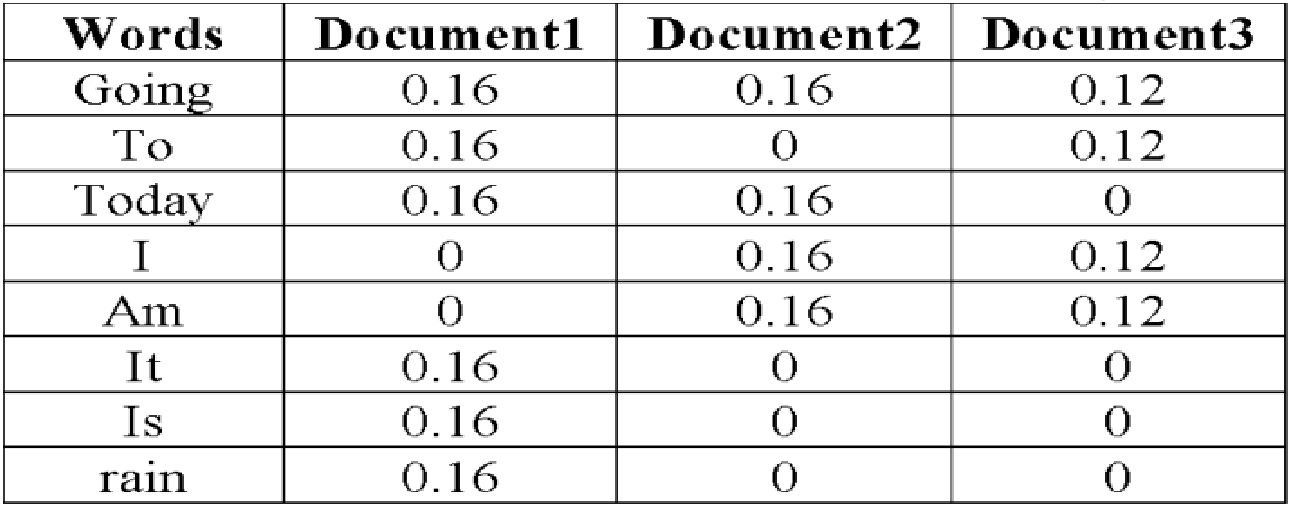
[going:3, to:2, today:2, i:2, am:2, it:1, is:1, rain:1]

Finding the term frequency



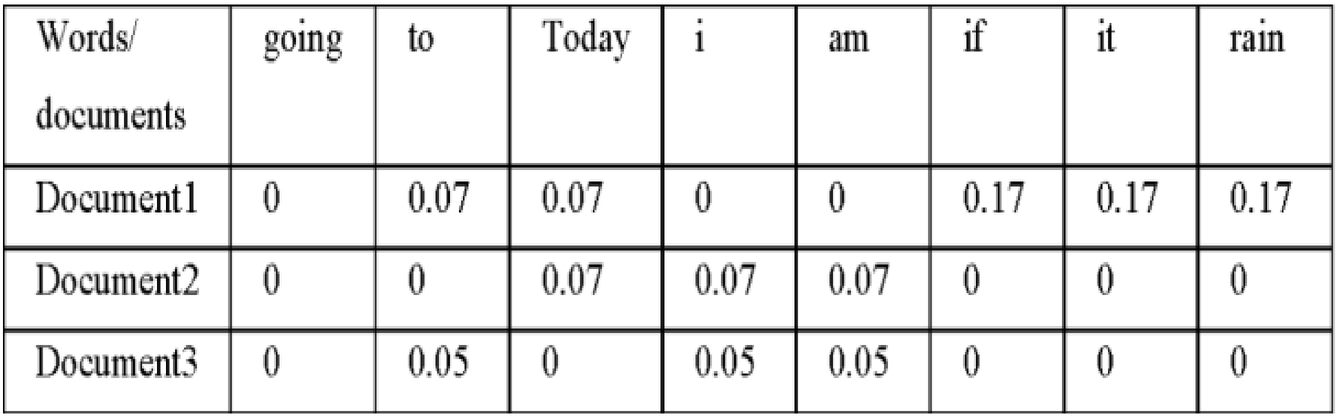
| Words | IDF Value |
|---|---|
| Going | log(3/3) |
| To | log(3/2) |
| Today | log(3/2) |
| I | log(3/2) |
| Am | log(3/2) |
| It | log(3/1) |
| Is | log(3/1) |
| rain | log(3/1) |

Inverse document frequency



| Words | Document1 | Document2 | Document3 |
|---|---|---|---|
| Going | 0.16 | 0.16 | 0.12 |
| To | 0.16 | 0 | 0.12 |
| Today | 0.16 | 0.16 | 0 |
| I | 0 | 0.16 | 0.12 |
| Am | 0 | 0.16 | 0.12 |
| It | 0.16 | 0 | 0 |
| Is | 0.16 | 0 | 0 |
| rain | 0.16 | 0 | 0 |

Applying the final equation the values of tf-idf becomes



| Words/ documents | going | to | Today | i | am | if | it | rain |
|---|---|---|---|---|---|---|---|---|
| Document1 | 0 | 0.07 | 0.07 | 0 | 0 | 0.17 | 0.17 | 0.17 |
| Document2 | 0 | 0 | 0.07 | 0.07 | 0.07 | 0 | 0 | 0 |
| Document3 | 0 | 0.05 | 0 | 0.05 | 0.05 | 0 | 0 | 0 |

Using the above two language models the complete data has been converted into two kinds of vectors and stored into a csv type file for easy access and minimal processing.

### 4.5.3 Data Splitting

The data splitting is done to create two kinds of data Training data and testing data. Training data is used to train the machine learning models and testing data is used to test the models and analyse results. 80% of total data is selected as training data and remaining data is testing data.

### 4.5.4 Machine Learning

#### 4.5.4.1 Introduction

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people. Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs. Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learning, suggest what movies or television shows to watch next based on user preferences. Self-driving cars that rely on machine learning to navigate may soon be available to consumers. Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes. In this tutorial, we'll look into the common machine learning methods of supervised and unsupervised learning, and common algorithmic approaches in machine learning, including the k-nearest neighbor algorithm, decision tree learning, and deep learning. We'll explore which programming languages are most used in machine learning, providing you with some of the positive and negative attributes of each. Additionally, we'll discuss biases that are perpetuated by machine learning algorithms, and consider what can be kept in mind to prevent these biases when building algorithms.

Machine Learning is process in which the computer performs certain tasks without giving instructions. In this case the models takes the training data and train on them. Then depending on the trained data any new unknown data will be processed based on the ruled derived from the trained data. After completing the countvectorization and TF-IDF stages in the workflow the data is converted into vector form(numerical form) which is used for training and testing models. For our study various machine learning models are compared to determine which method is more suitable for this task. The models used for the study include Logistic Regression, Naïve Bayes, Random Forest Classifier, K Nearest Neighbors, and Support Vector Machine Classifier and a proposed model which was created using an ensemble approach.

# ACTIVITY LOG FOR THE WEEK

| Day<br>&<br>Date | Brief description<br>of the<br>Daily activity | Learning<br>Outcome | Person In-<br>Charge<br>Signature |
|---|---|---|---|
| Day – 1 | | | |
| Day - 2 | | | |
| Day – 3 | | | |
| Day – 4 | | | |
| Day – 5 | | | |
| Day –6 | | | |

# WEEKLY REPORT

**Objective of the Activity Done:**

**Detailed Report:**

### 4.5.4.2 Algorithms

a combination of 5 algorithms are used for the classifications.

#### 4.5.4.2.1 Naïve Bayes Classifier

A naïve Bayes classifier is a supervised probabilistic machine learning model that is used for classification tasks. The main principle behind this model is the Bayes theorem. Bayes Theorem: Naive Bayes is a classification technique that is based on Bayes' Theorem with an assumption that all the features that predict the target value are independent of each other. It calculates the probability of each class and then picks the one with the highest probability.

Naive Bayes classifier assumes that the features we use to predict the target are independent and do not affect each other. Though the independence assumption is never correct in real-world data, but often works well in practice. so that it is called "Naive".

$$P(A \mid B)=(P(B \mid A)P(A))/P(B)$$

P(A|B) is the probability of hypothesis A given the data B. This is called the posterior probability. P(B|A) is the probability of data B given that hypothesis A was true.

P(A) is the probability of hypothesis A being true (regardless of the data). This is called the prior probability of A.

P(B) is the probability of the data (regardless of the hypothesis).

Naïve Bayes classifiers are mostly used for text classification. The limitation of the Naïve Bayes model is that it treats every word in a text as independent and is equal in importance but every word cannot be treated equally important because articles and nouns are not the same when it comes to language. But due to its classification efficiency, this model is used in combination with other language processing techniques.

#### 4.5.4.2.2 Random Forest Classifier

Random Forest classifier is a supervised ensemble algorithm. A random forest consists of multiple random decision trees. Two types of randomnesses are built into the trees. First, each tree is built on a random sample from the original data. Second, at each tree node, a subset of features is randomly selected to generate the best split [16]. Decision Tree: The decision tree is a classification algorithm based completely on features. The tree repeatedly splits the data on a feature with the best information gain. This process continues until the information gained remains constant. Then the unknown data is evaluated feature by feature until categorized. Tree pruning techniques are used for improving accuracy and reducing the overfitting of data. Several decision trees are created on subsets of data the result that was given by the majority of trees is considered as the final result. The number of trees to be created is determined based on accuracy and other metrics through iterative methods. Random forest classifiers are mainly used on condition-based data but it works for text if the text is converted into numerical form.

### 4.5.4.2.3 Logistic Regression

Logistic Regression is a "Supervised machine learning" algorithm that can be used to model the probability of a certain class or event. It is used when the data is linearly separable and the outcome is binary or dichotomous. The probabilities are calculated using a sigmoid function. For example, let us take a problem where data has n features. We need to fit a line for the given data and this line can be represented by the equation

z=b_0+b_1 x_1+b_2 x_2+b_3 x_3….+b_n x_n

here z = odds

generally, odds are calculated as

odds=p(event occurring)/p(event not occurring)

**Sigmoid Function**: A sigmoid function is a special form of logistic function hence the name logistic regression. The logarithm of odds is calculated and fed into the sigmoid function to get continuous probability ranging from 0 to 1.

The logarithm of odds can be calculated by

log(odds) = dot(features, coefficients) + intercept

and these log_odds are used in the sigmoid function to get probability.

h(z)=1/(1+e^(-z) )

The output of the sigmoid function is an integer in the range 0 to 1 which is used to determine which class the sample belongs to. Generally, 0.5 is considered as the limit below which it is considered a NO, and 0.5 or higher will be considered a YES. But the border can be adjusted based on the requirement.

### 4.5.4.2.4 K-Nearest Neighbors

KNN is a classification algorithm. It comes under supervised algorithms. All the data points are assumed to be in an n-dimensional space. And then based on neighbors the category of current data is determined based on the majority. Euclidian distance is used to determine the distance between points.

The distance between 2 points is calculated as

d=√( 〖(x2-x1)〗 ^2+ 〖(y2-y1)〗 ^2 )

The distances between the unknown point and all the others are calculated. Depending on the K provided k closest neighbors are determined. The category to which the majority of the neighbors belong is selected as the unknown data category. If the data contains up to 3 features then the plot can be visualized. It is fairly slow compared to other distance-based algorithms such as SVM as it needs to determine the distance to all points to get the closest neighbors to the given point.

### 4.5.4.2.5 Support Vector Machines(SVM)

It is a machine learning algorithm for classification. Decision boundaries are drawn between various categories and based on which side the point falls to the boundary the category is determined.

**Support Vectors**: The vectors closer to boundaries are called support vectors/planes. If there are n categories then there will be n+1 support vectors. Instead of points, these are called vectors because they are assumed to be starting from the origin.The distance between the support vectors is called margin. We want our margin to be as wide as possible because it yields better results. There are three types of boundaries used by SVM to create boundaries.

Linear: used if the data is linearly separable.

Poly: used if data is not separable. It creates any data into 3-dimensional data.

Radial: this is the default kernel used in SVM. It converts any data into infinite-dimensional data.

If the data is 2-dimensional then the boundaries are lines. If the data is 3-dimensional then the boundaries are planes. If the data categories are more than

3 then boundaries are called hyperplanes. An SVM mainly depends on the decision boundaries for predictions. It doesn't compare the data to all other data to get the prediction due to this SVM's tend to be quick with predictions.

### 4.5.5 Experimentation

The process goes like data collection and processing then natural language processing and then vectorization then machine learning.The data is collected, cleaned, and then subjected to natural language processing techniques specified. Then the cleaned data is converted into vectors using Bag of Words and TF-IDF methods which goes like...

The Data is split into Training data and Testing Data in an 80-20 split ratio. The training and testing data is converted into Bag-of-Words vectors and TF-IDF vectors.

There are several metrics to evaluate the models but accuracy is considered for comparing BoW and TF-IDF models.

Accuracy is generally used to determine the efficiency of a model. Accuracy: "Accuracy is the number of correctly predicted data points out of all the data points".

**Naïve Bayes Classification algorithm**: Two models, one for Bow and one for TF-IDF are created and trained using respective training vectors and training labels. Then the respective testing vectors and labels are used to get the score for the model.

The scores for Bag-of-Words and TF-IDF are visualized. The scores for the Bow model and TF-IDF models are 98.04 and 96.05 respectively for using the naïve bayes model. Logistic Regression: Two models are created following the same procedure used for naïve Bayes models and then tested the results obtained are visualized below.

**K-Nearest Neighbors**: Similar to the above models the models are created and trained using respective vectors and labels. But in addition to the data, the number of neighbors to be considered should also be provided. Using Iterative Method K =3 (no of Neighbors) provided the best results for the BoW model and K = 9 provided the best results for the TF-IDF model.

### 4.5.7 Working Procedure

The working procedure includes the internal working and the data flow of application.

i. After running the application some procedures are automated.
   1. Reading data from file
   2. Cleaning the texts
   3. Processing
   4. Splitting the data
   5. Intialising and training the models

ii. The user just needs to provide some data to classify in the area provided.

iii. The provided data undergoes several procedures after submission.
   1. Textual Processing
   2. Feature Vector conversion
   3. Entity extraction

iv. The created vectors are provided to trained models to get predictions.

v. After getting predictions the category predicted by majority will be selected.

vi. The accuracies of that prediction will be calculated

vii. The accuracies and entities extracted from the step 3 will be provided to user. Every time the user gives something new the procedure from step 2 will be repeated.

# ACTIVITY LOG FOR THE WEEK

| Day<br>&<br>Date | Brief description of the Daily activity | Learning Outcome | Person In-Charge Signature |
|---|---|---|---|
| Day – 1 | | | |
| Day - 2 | | | |
| Day – 3 | | | |
| Day – 4 | | | |
| Day – 5 | | | |
| Day –6 | | | |

# WEEKLY REPORT

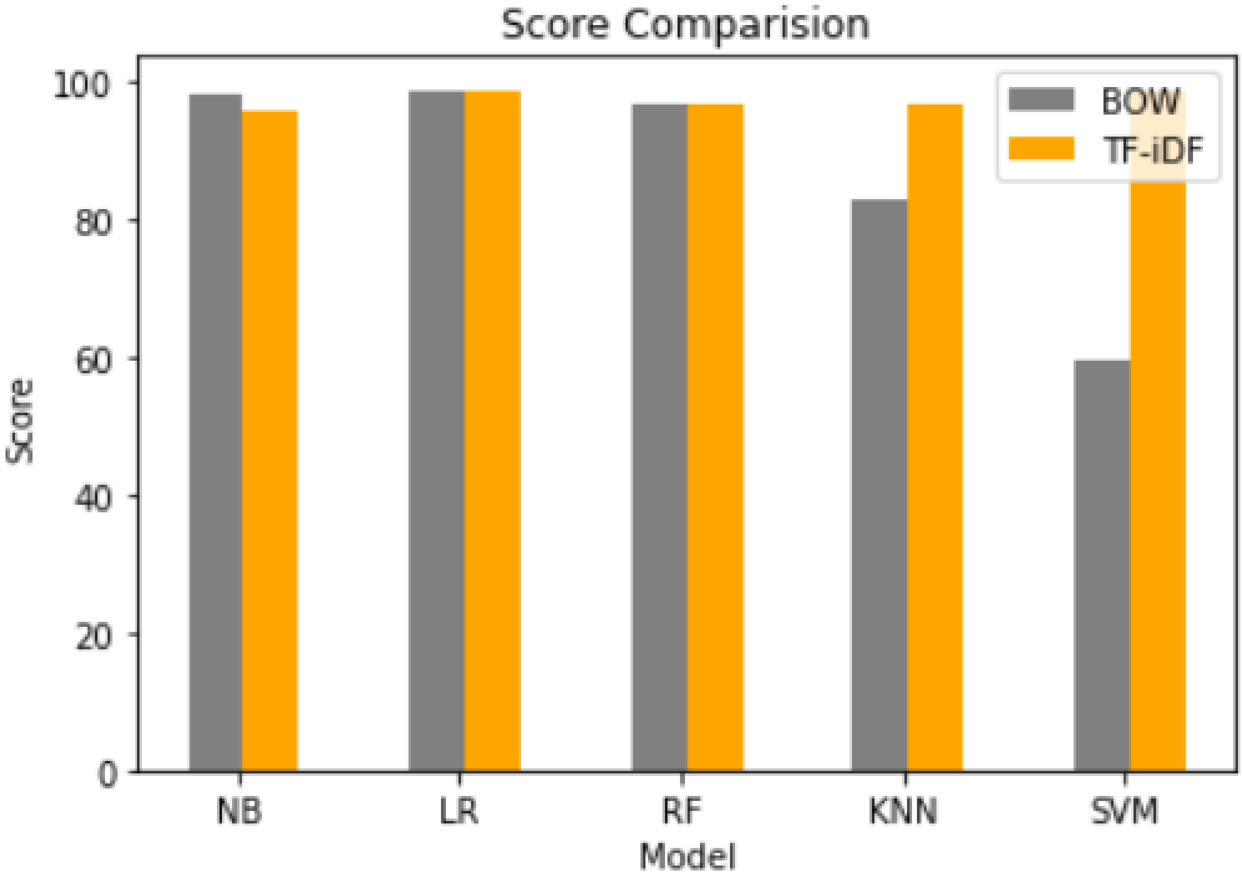
**Objective of the Activity Done:**

**Detailed Report:**

## 5. Results and Discussion

### 5.1 Language Model Selection

While selecting the best language model the data has been converted into both types of vectors and then the models been tested for to determine the best model for classifying spam. The results from individual models are presented in the experimentation section under methodology. Now comparing the results from the models.



From the figure it is clear that TF-IDF proves to be better than BoW in every model tested. Hence TF-IDF has been selected as the primary language model for textual data conversion in feature vector formation.

### 5.2 Proposed Model results

To determine which model is effective we used three metrics Accuracy, Precision, and F1score.

The resulted values for the proposed model are

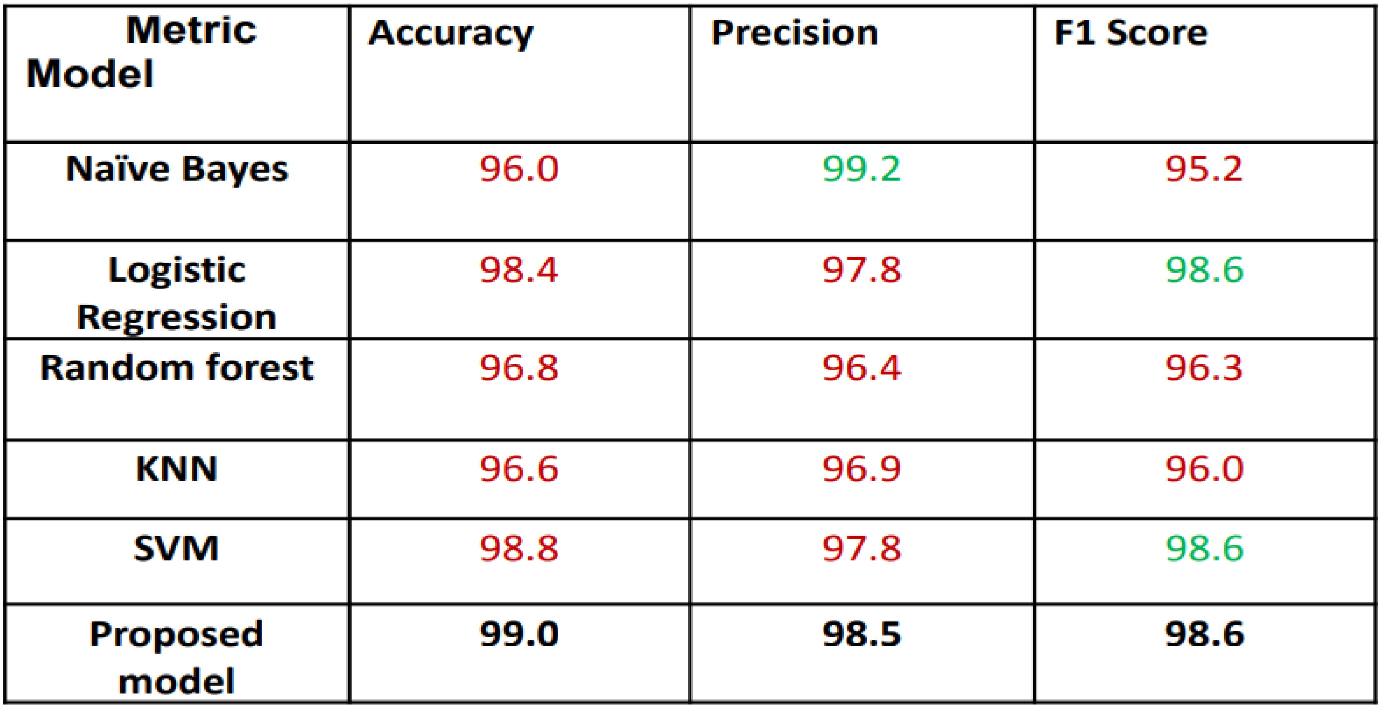
Accuracy – 99.0

Precision – 98.5

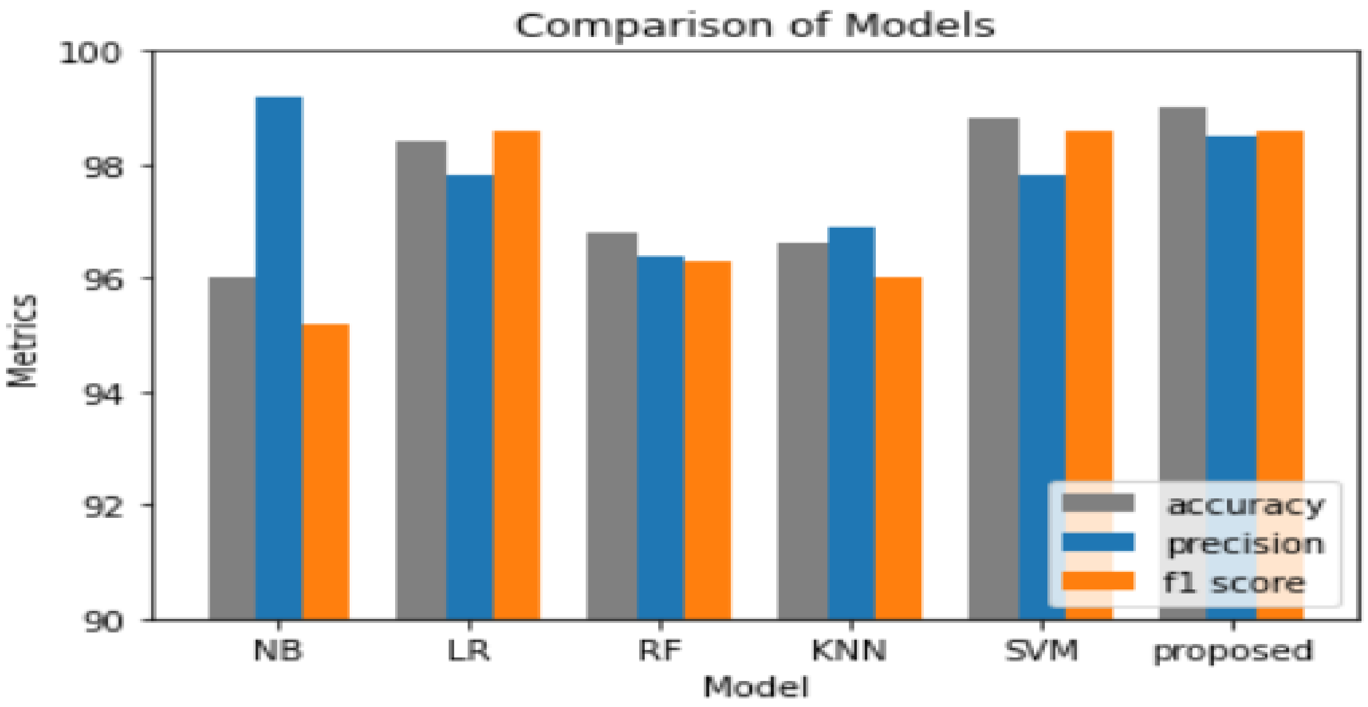
F1 Score – 98.6

### 5.3 Comparison

The results from the proposed model has been compared with all the models individually in tabular form to illustrate the differences clearly.



| Metric<br>Model | Accuracy | Precision | F1 Score |
|---|---|---|---|
| Naïve Bayes | 96.0 | 99.2 | 95.2 |
| Logistic Regression | 98.4 | 97.8 | 98.6 |
| Random forest | 96.8 | 96.4 | 96.3 |
| KNN | 96.6 | 96.9 | 96.0 |
| SVM | 98.8 | 97.8 | 98.6 |
| Proposed model | **99.0** | **98.5** | **98.6** |



Here we can observe that our proposed model outperforms almost every other model in every metric. Only one model(naïve Bayes) has slightly higher accuracy than our model but it is considerably lagging in other metrics.

From the above comparison barchart we can clearly see that all models individually are not as efficient as the proposed method.

### 5.4 Summary

There are two main tasks in the project implementation. Language model selection for completing the textual processing phase and proposed model creation using the individual algorithms. These two tasks require comparison from other models and select of various parameters for better efficiency.

During the language model selection phase two models, Bag of Words and TF-IDF are compared to select the best model and from the results obtained it is evident that TF-IDF performs better.

During the proposed model design various algorithms are tested with different parameters to get best parameters. Models are merged to form a ensemble algorithm and the results obtained are presented and compared above. It is clear from the results that the proposed model outperforms others in almost every metric derived.

## 6. Conclusion and Future Scope

### 6.1 Conclusion

From the results obtained we can conclude that an ensemble machine learning model is more effective in detection and classification of spam than any individual algorithms. We can also conclude that TF-IDF (term frequency inverse document frequency) language model is more effective than Bag of words model in classification of spam when combined with several algorithms. And finally we can say that spam detection can get better if machine learning algorithms are combined and tuned to needs.

### 6.2 Future work

There are numerous appilcations to machine learning and natural language processing and when combined they can solve some of the most troubling problems concerned with texts. This application can be scaled to intake text in bulk so that classification can be done more affectively in some public sites. Other contexts such as negative, phishing, malicious, etc,. can be used to train the model to filter things such as public comments in various social sites. This application can be converted to online type of machine learning system and can be easily updated with latest trends of spam and other mails so that the system can adapt to new types of spam emails and texts.

## ACTIVITY LOG FOR THE WEEK

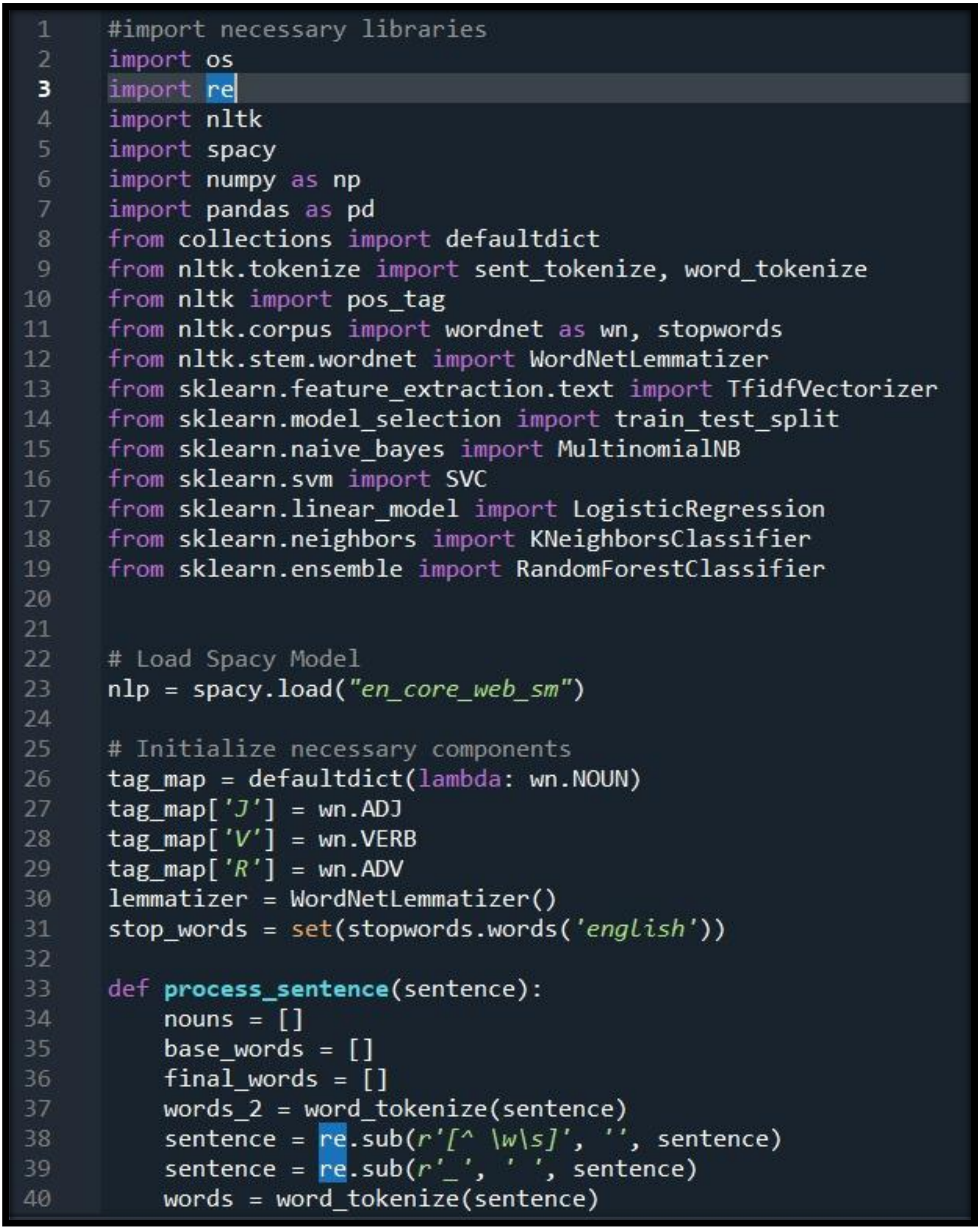
| Day<br>&<br>Date | Brief description<br>of the<br>Daily activity | Learning<br>Outcome | Person In-<br>Charge<br>Signature |
|---|---|---|---|
| Day – 1 | | | |
| Day - 2 | | | |
| Day – 3 | | | |
| Day – 4 | | | |
| Day – 5 | | | |
| Day –6 | | | |

# WEEKLY REPORT

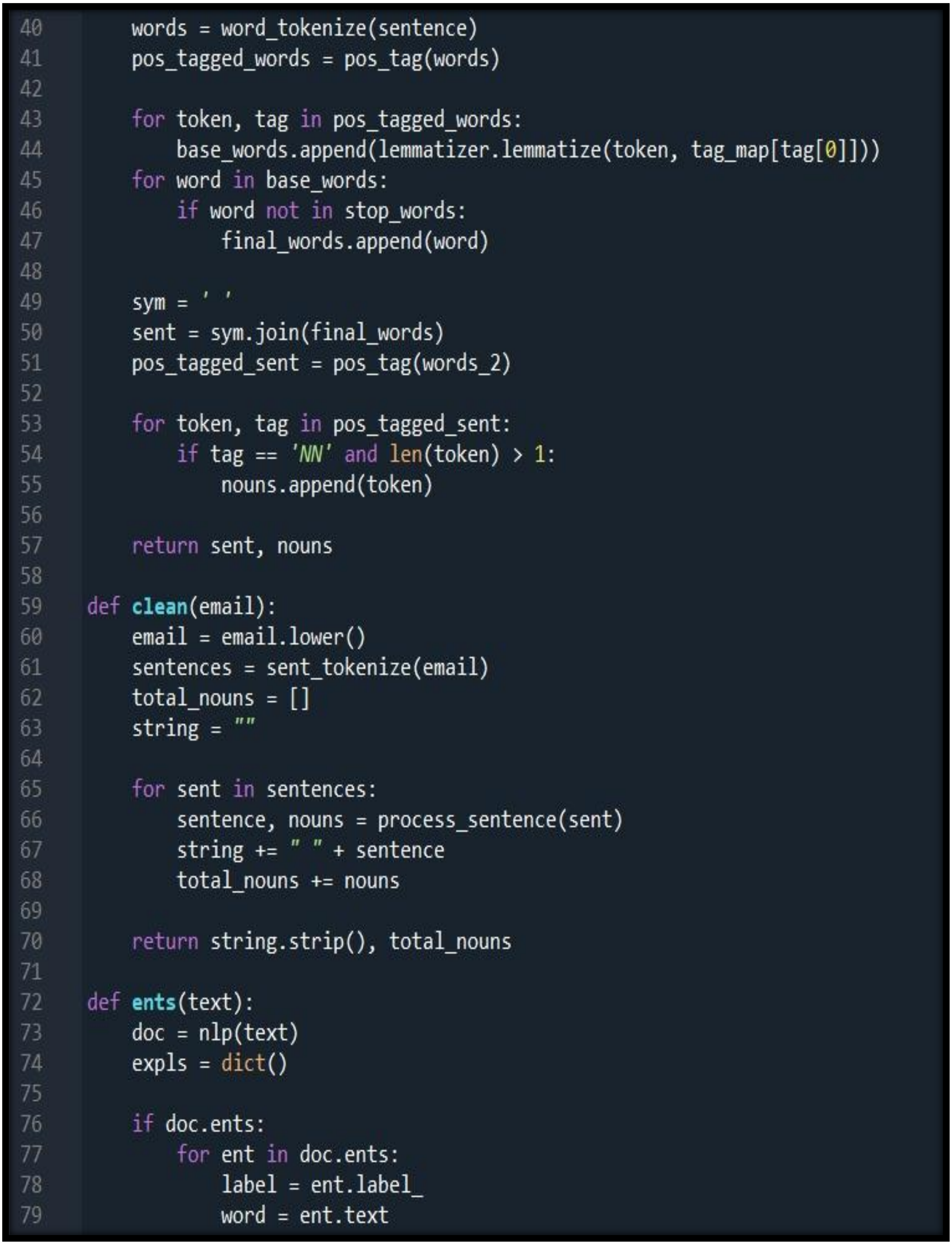
**Objective of the Activity Done:**

**Detailed Report:**

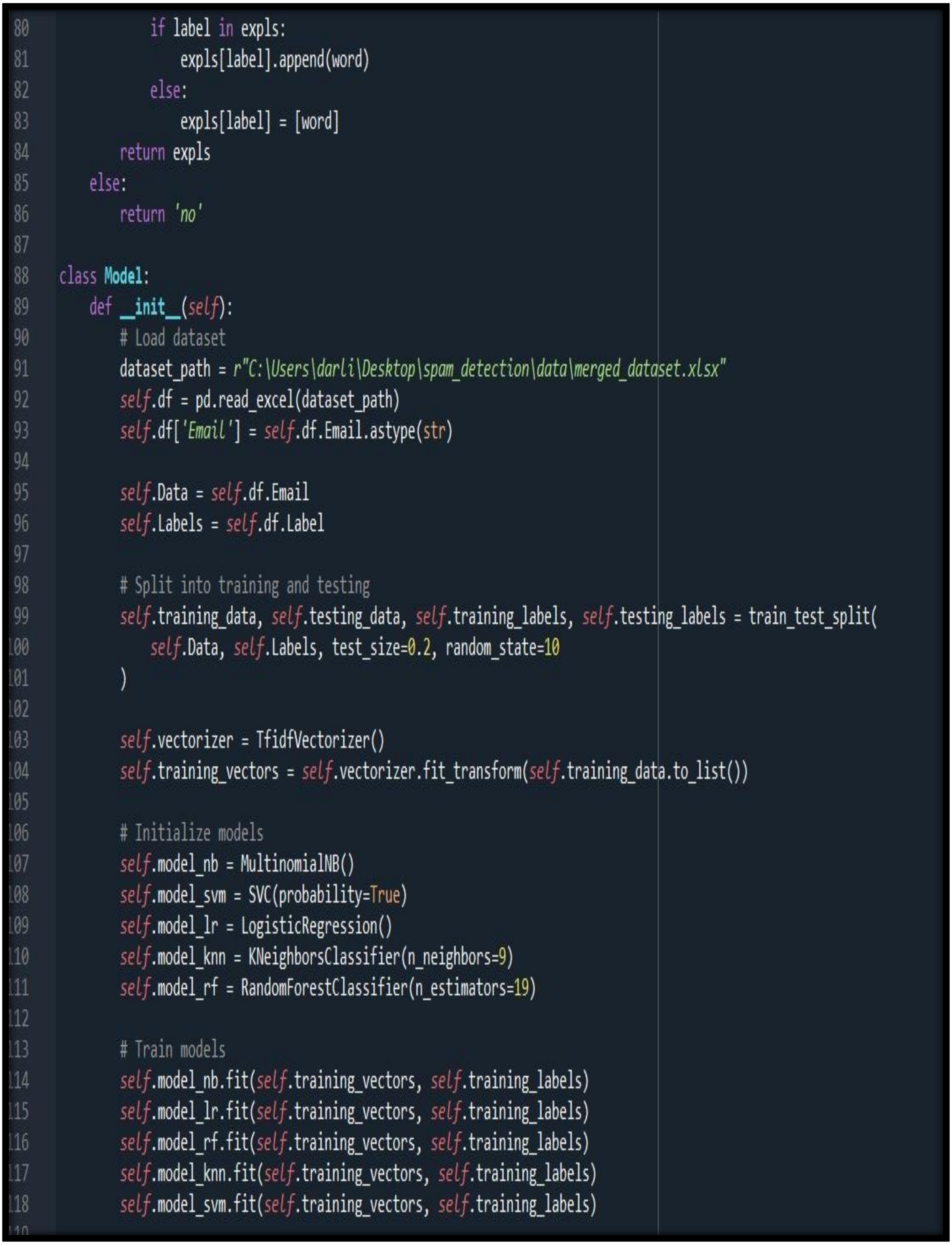
## Source Code.



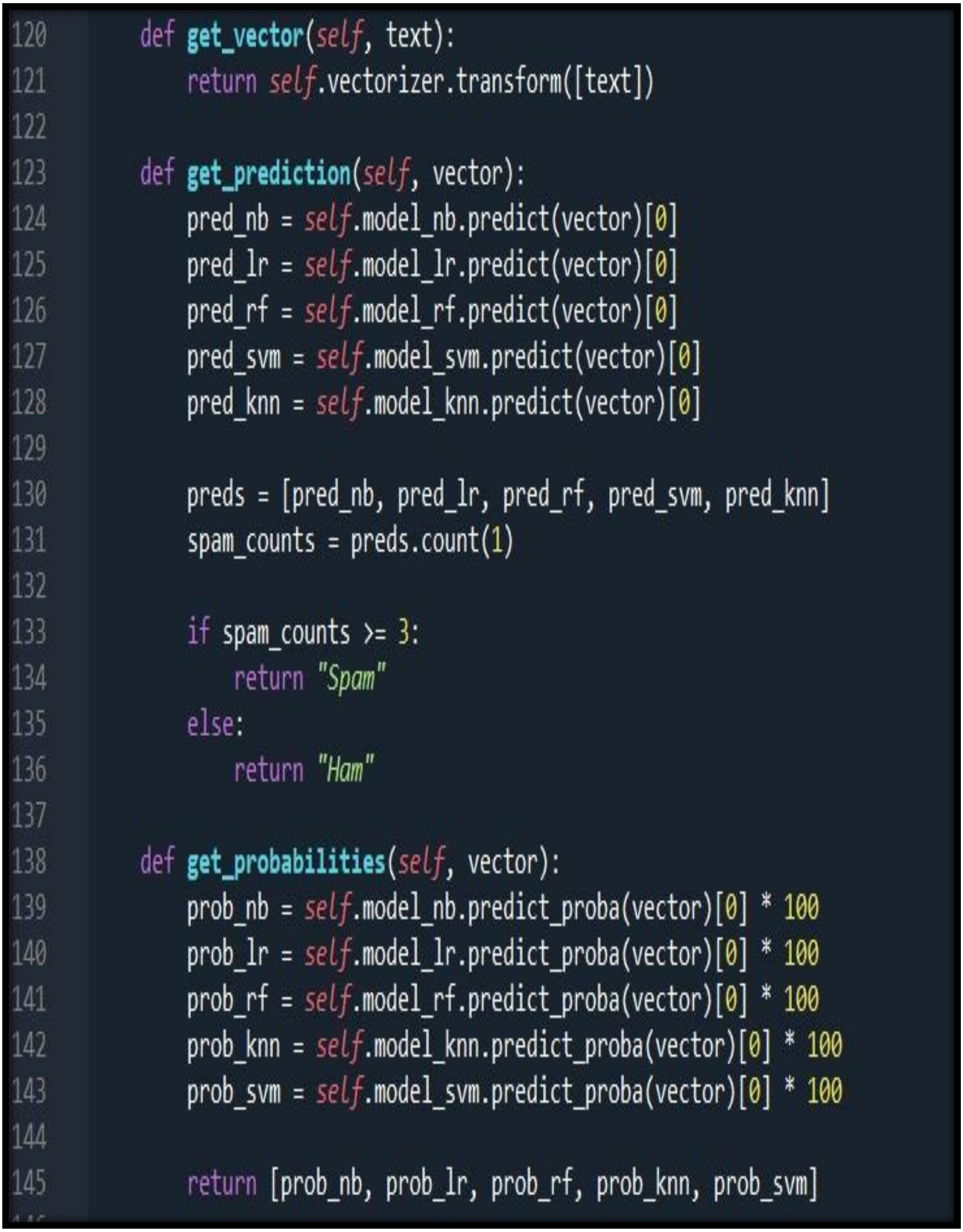
```python
#import necessary libraries
import os
import re
import nltk
import spacy
import numpy as np
import pandas as pd
from collections import defaultdict
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk import pos_tag
from nltk.corpus import wordnet as wn, stopwords
from nltk.stem.wordnet import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier


# Load Spacy Model
nlp = spacy.load("en_core_web_sm")

# Initialize necessary components
tag_map = defaultdict(lambda: wn.NOUN)
tag_map['J'] = wn.ADJ
tag_map['V'] = wn.VERB
tag_map['R'] = wn.ADV
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))

def process_sentence(sentence):
    nouns = []
    base_words = []
    final_words = []
    words_2 = word_tokenize(sentence)
    sentence = re.sub(r'[^ \w\s]', '', sentence)
    sentence = re.sub(r'_', ' ', sentence)
    words = word_tokenize(sentence)
```



```python
    words = word_tokenize(sentence)
    pos_tagged_words = pos_tag(words)

    for token, tag in pos_tagged_words:
        base_words.append(lemmatizer.lemmatize(token, tag_map[tag[0]]))
    for word in base_words:
        if word not in stop_words:
            final_words.append(word)

    sym = ' '
    sent = sym.join(final_words)
    pos_tagged_sent = pos_tag(words_2)

    for token, tag in pos_tagged_sent:
        if tag == 'NN' and len(token) > 1:
            nouns.append(token)

    return sent, nouns

def clean(email):
    email = email.lower()
    sentences = sent_tokenize(email)
    total_nouns = []
    string = ""

    for sent in sentences:
        sentence, nouns = process_sentence(sent)
        string += " " + sentence
        total_nouns += nouns

    return string.strip(), total_nouns

def ents(text):
    doc = nlp(text)
    expls = dict()

    if doc.ents:
        for ent in doc.ents:
            label = ent.label_
            word = ent.text
```

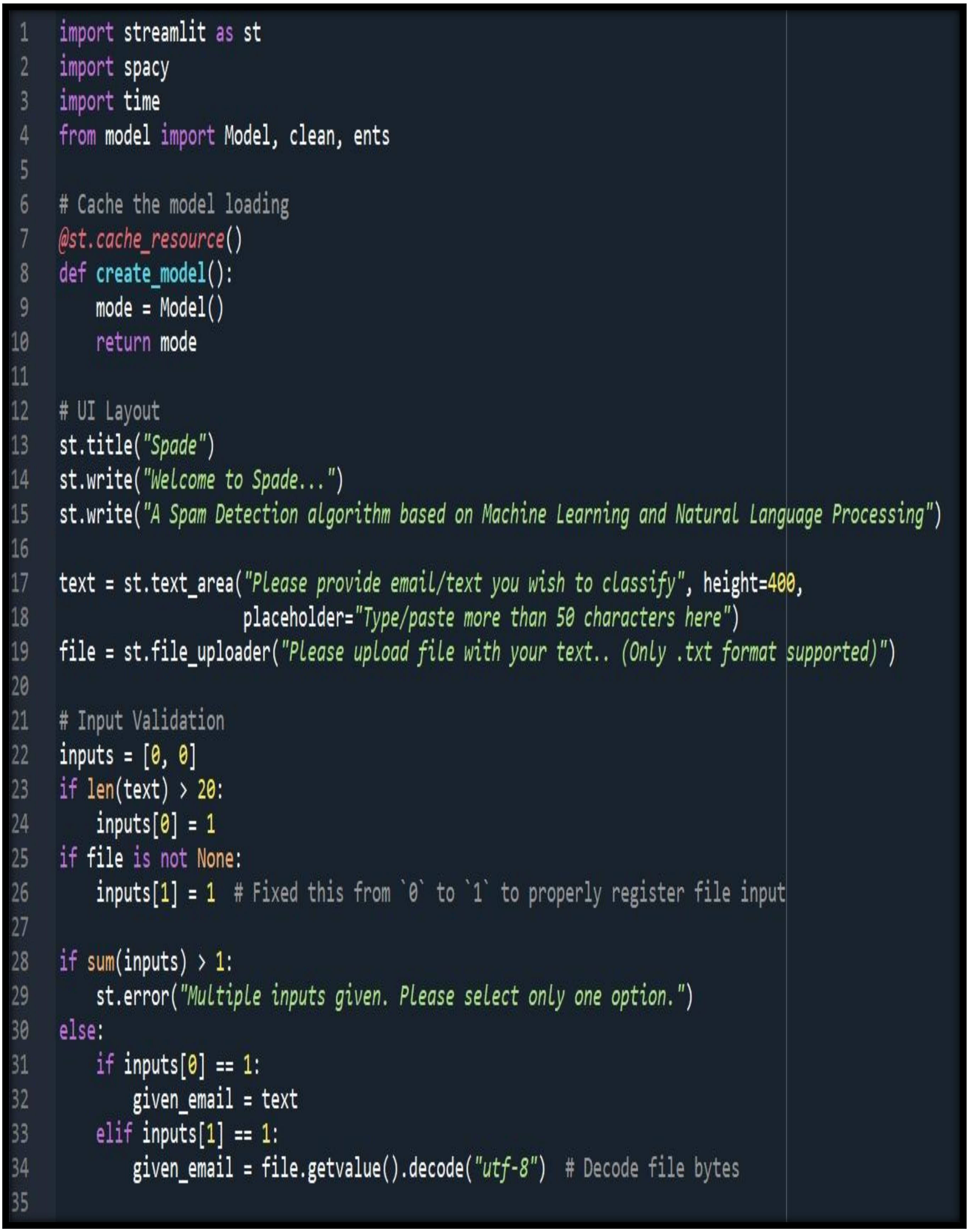


```
            if label in expls:
                expls[label].append(word)
            else:
                expls[label] = [word]
        return expls
    else:
        return 'no'

class Model:
    def __init__(self):
        # Load dataset
        dataset_path = r"C:\Users\darli\Desktop\spam_detection\data\merged_dataset.xlsx"
        self.df = pd.read_excel(dataset_path)
        self.df['Email'] = self.df.Email.astype(str)

        self.Data = self.df.Email
        self.Labels = self.df.Label

        # Split into training and testing
        self.training_data, self.testing_data, self.training_labels, self.testing_labels = train_test_split(
            self.Data, self.Labels, test_size=0.2, random_state=10
        )

        self.vectorizer = TfidfVectorizer()
        self.training_vectors = self.vectorizer.fit_transform(self.training_data.to_list())

        # Initialize models
        self.model_nb = MultinomialNB()
        self.model_svm = SVC(probability=True)
        self.model_lr = LogisticRegression()
        self.model_knn = KNeighborsClassifier(n_neighbors=9)
        self.model_rf = RandomForestClassifier(n_estimators=19)

        # Train models
        self.model_nb.fit(self.training_vectors, self.training_labels)
        self.model_lr.fit(self.training_vectors, self.training_labels)
        self.model_rf.fit(self.training_vectors, self.training_labels)
        self.model_knn.fit(self.training_vectors, self.training_labels)
        self.model_svm.fit(self.training_vectors, self.training_labels)
```

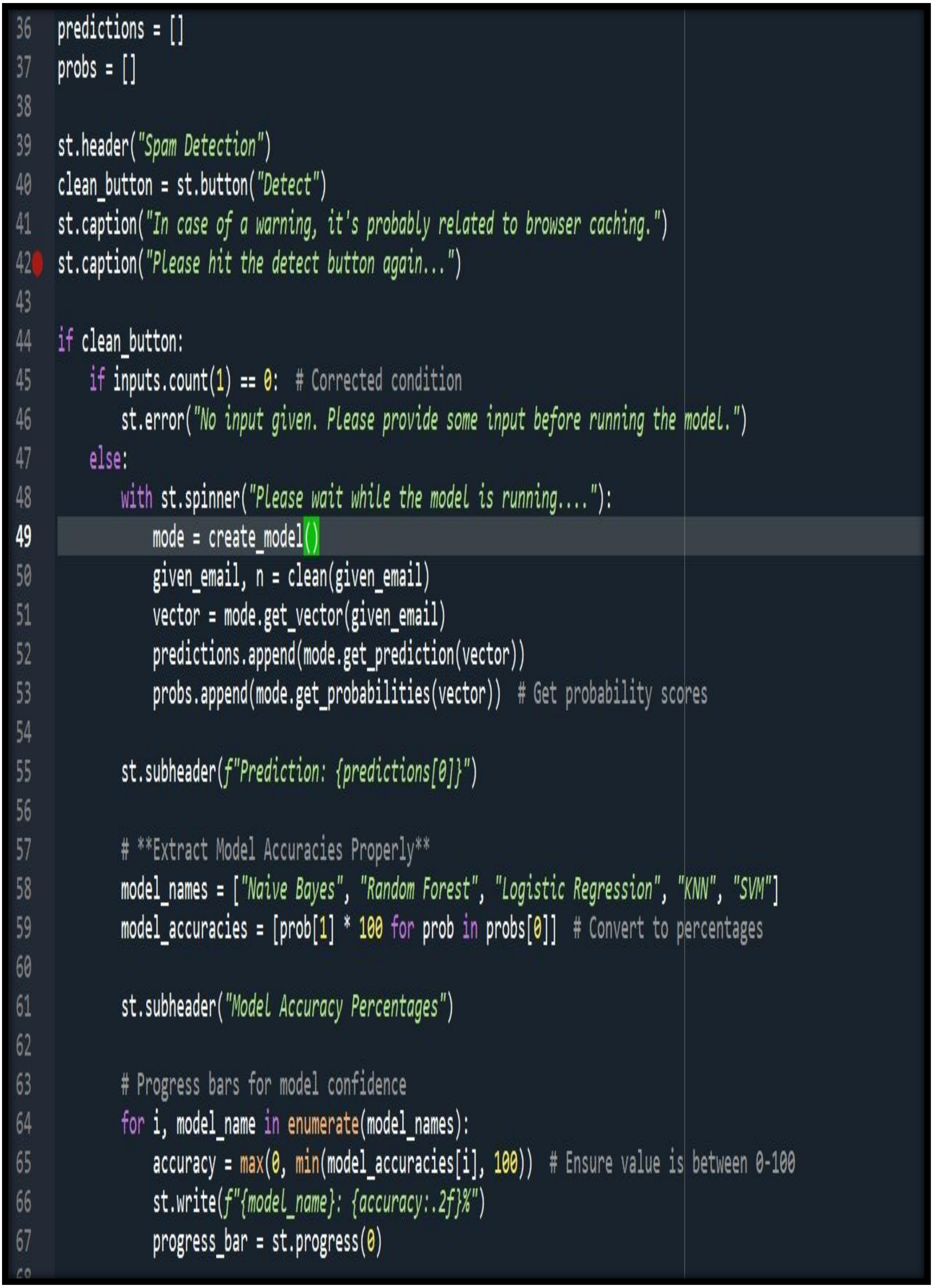


```
    def get_vector(self, text):
        return self.vectorizer.transform([text])

    def get_prediction(self, vector):
        pred_nb = self.model_nb.predict(vector)[0]
        pred_lr = self.model_lr.predict(vector)[0]
        pred_rf = self.model_rf.predict(vector)[0]
        pred_svm = self.model_svm.predict(vector)[0]
        pred_knn = self.model_knn.predict(vector)[0]

        preds = [pred_nb, pred_lr, pred_rf, pred_svm, pred_knn]
        spam_counts = preds.count(1)

        if spam_counts >= 3:
            return "Spam"
        else:
            return "Ham"

    def get_probabilities(self, vector):
        prob_nb = self.model_nb.predict_proba(vector)[0] * 100
        prob_lr = self.model_lr.predict_proba(vector)[0] * 100
        prob_rf = self.model_rf.predict_proba(vector)[0] * 100
        prob_knn = self.model_knn.predict_proba(vector)[0] * 100
        prob_svm = self.model_svm.predict_proba(vector)[0] * 100

        return [prob_nb, prob_lr, prob_rf, prob_knn, prob_svm]
```

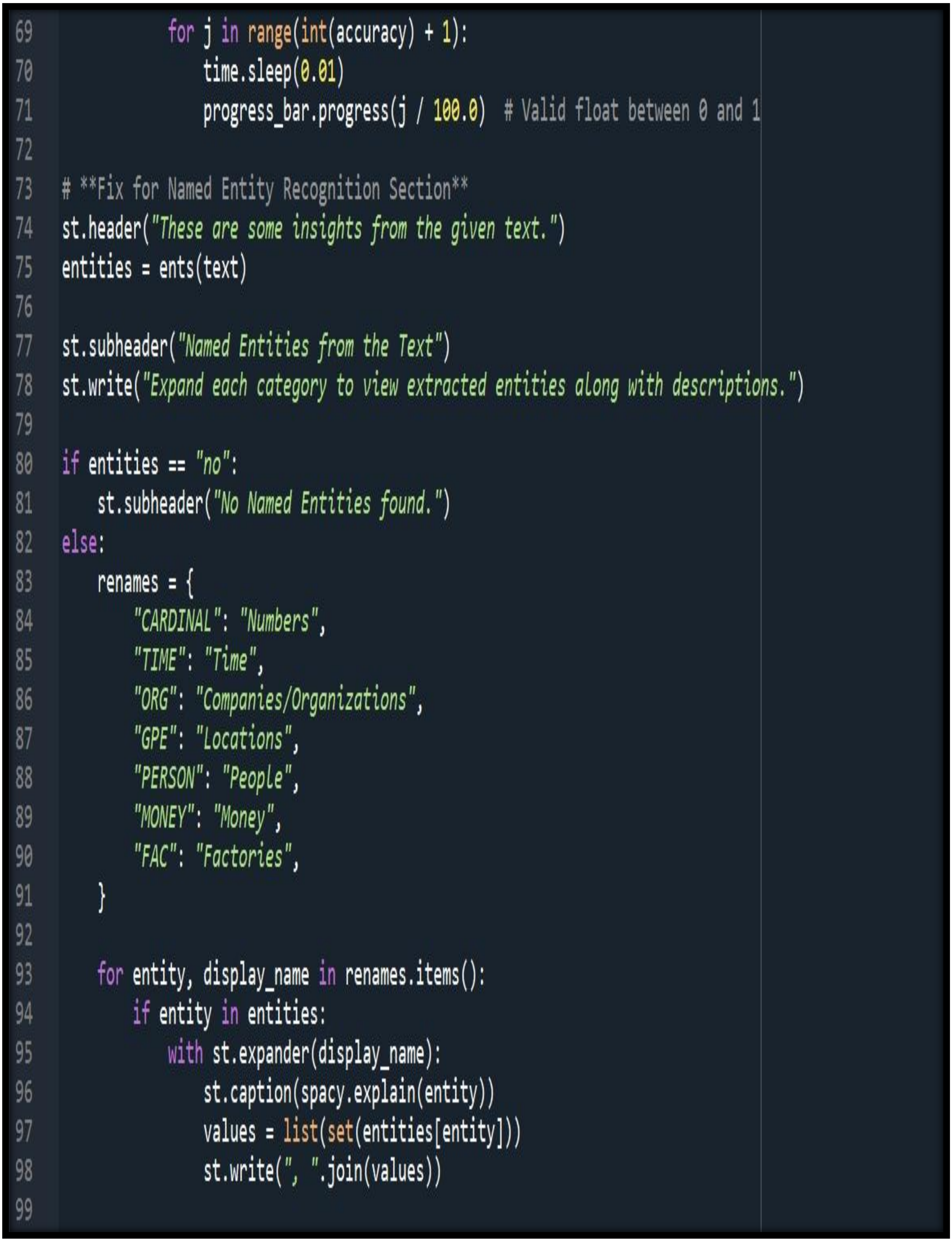
## Deployment code:



```python
import streamlit as st
import spacy
import time
from model import Model, clean, ents

# Cache the model loading
@st.cache_resource()
def create_model():
    mode = Model()
    return mode

# UI Layout
st.title("Spade")
st.write("Welcome to Spade...")
st.write("A Spam Detection algorithm based on Machine Learning and Natural Language Processing")

text = st.text_area("Please provide email/text you wish to classify", height=400,
                    placeholder="Type/paste more than 50 characters here")
file = st.file_uploader("Please upload file with your text.. (Only .txt format supported)")

# Input Validation
inputs = [0, 0]
if len(text) > 20:
    inputs[0] = 1
if file is not None:
    inputs[1] = 1  # Fixed this from `0` to `1` to properly register file input

if sum(inputs) > 1:
    st.error("Multiple inputs given. Please select only one option.")
else:
    if inputs[0] == 1:
        given_email = text
    elif inputs[1] == 1:
        given_email = file.getvalue().decode("utf-8")  # Decode file bytes

```

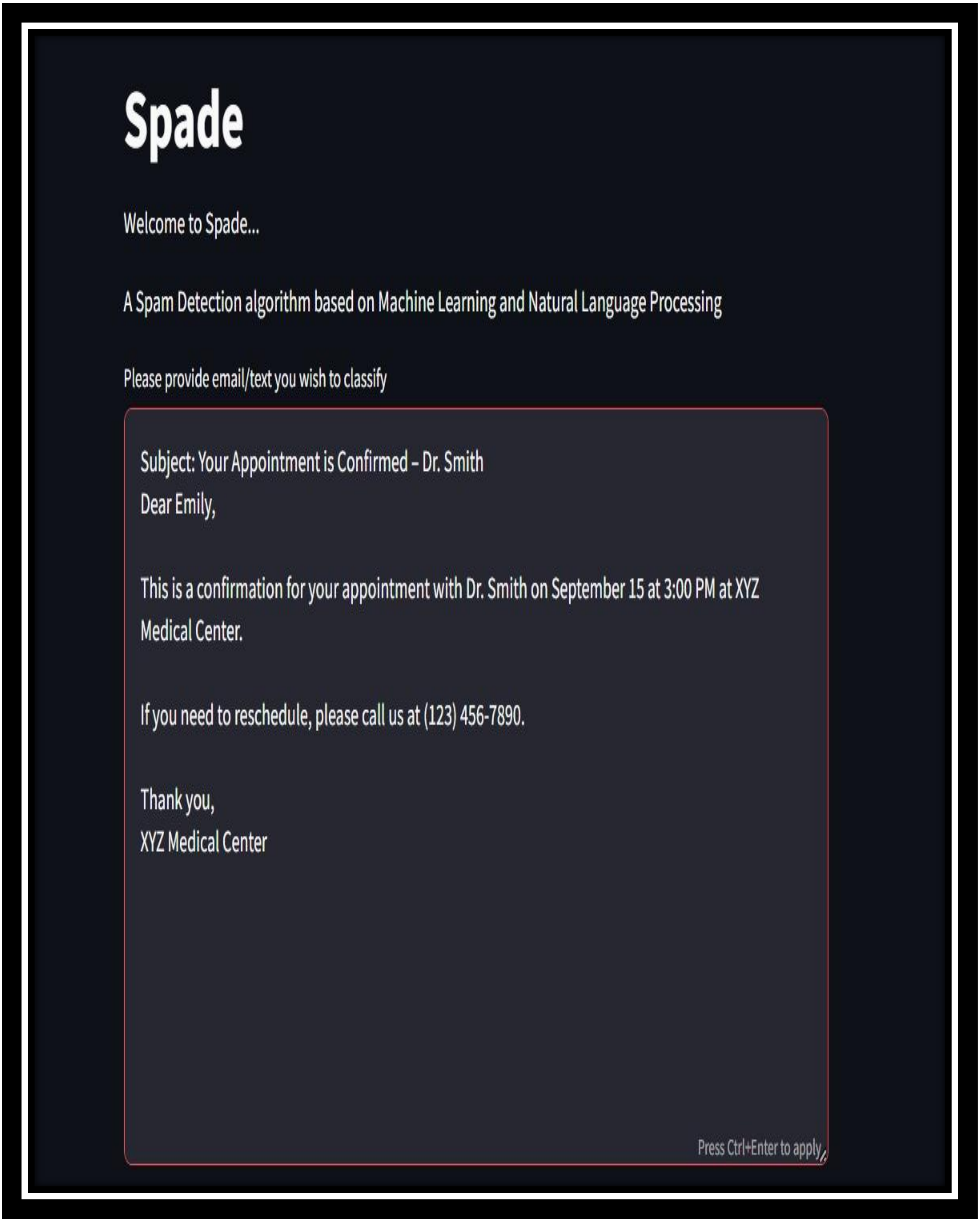


```
predictions = []
probs = []

st.header("Spam Detection")
clean_button = st.button("Detect")
st.caption("In case of a warning, it's probably related to browser caching.")
st.caption("Please hit the detect button again...")

if clean_button:
    if inputs.count(1) == 0:  # Corrected condition
        st.error("No input given. Please provide some input before running the model.")
    else:
        with st.spinner("Please wait while the model is running...."):
            mode = create_model()
            given_email, n = clean(given_email)
            vector = mode.get_vector(given_email)
            predictions.append(mode.get_prediction(vector))
            probs.append(mode.get_probabilities(vector))  # Get probability scores

        st.subheader(f"Prediction: {predictions[0]}")

        # **Extract Model Accuracies Properly**
        model_names = ["Naive Bayes", "Random Forest", "Logistic Regression", "KNN", "SVM"]
        model_accuracies = [prob[1] * 100 for prob in probs[0]]  # Convert to percentages

        st.subheader("Model Accuracy Percentages")

        # Progress bars for model confidence
        for i, model_name in enumerate(model_names):
            accuracy = max(0, min(model_accuracies[i], 100))  # Ensure value is between 0-100
            st.write(f"{model_name}: {accuracy:.2f}%")
            progress_bar = st.progress(0)
```



```
            for j in range(int(accuracy) + 1):
                time.sleep(0.01)
                progress_bar.progress(j / 100.0)  # Valid float between 0 and 1

# **Fix for Named Entity Recognition Section**
st.header("These are some insights from the given text.")
entities = ents(text)

st.subheader("Named Entities from the Text")
st.write("Expand each category to view extracted entities along with descriptions.")

if entities == "no":
    st.subheader("No Named Entities found.")
else:
    renames = {
        "CARDINAL": "Numbers",
        "TIME": "Time",
        "ORG": "Companies/Organizations",
        "GPE": "Locations",
        "PERSON": "People",
        "MONEY": "Money",
        "FAC": "Factories",
    }

    for entity, display_name in renames.items():
        if entity in entities:
            with st.expander(display_name):
                st.caption(spacy.explain(entity))
                values = list(set(entities[entity]))
                st.write(", ".join(values))

```

**User Interface:**



# Spade

Welcome to Spade...

A Spam Detection algorithm based on Machine Learning and Natural Language Processing

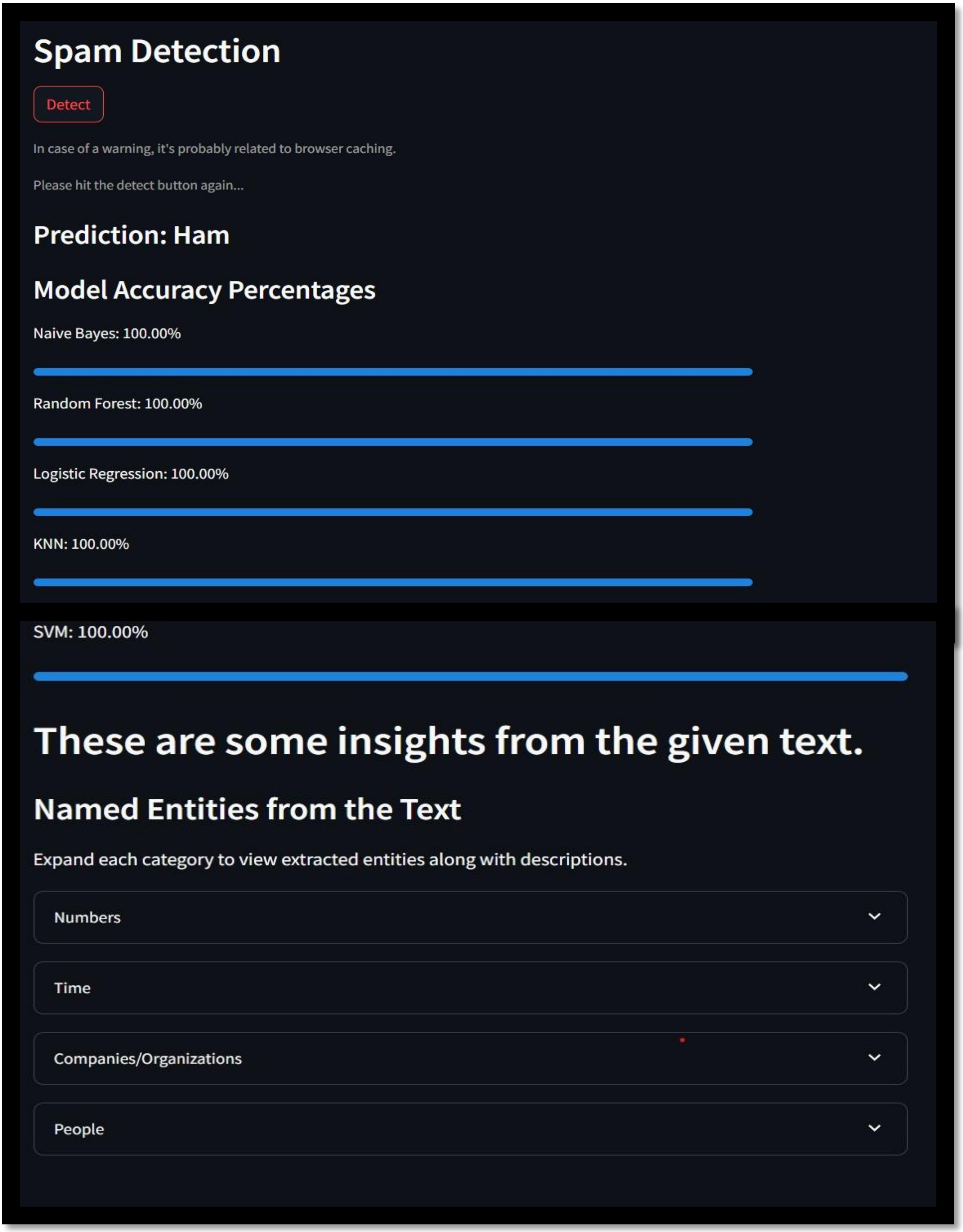
Please provide email/text you wish to classify

Subject: Your Appointment is Confirmed – Dr. Smith
Dear Emily,

This is a confirmation for your appointment with Dr. Smith on September 15 at 3:00 PM at XYZ Medical Center.

If you need to reschedule, please call us at (123) 456-7890.

Thank you,
XYZ Medical Center



# Spam Detection

In case of a warning, it's probably related to browser caching.

Please hit the detect button again...

## Prediction: Ham

## Model Accuracy Percentages

Naive Bayes: 100.00%

Random Forest: 100.00%

Logistic Regression: 100.00%

KNN: 100.00%

SVM: 100.00%

# These are some insights from the given text.

## Named Entities from the Text

Expand each category to view extracted entities along with descriptions.