AMITY UNIVERSITY ONLINE, NOIDA, UTTAR PRADESH

In partial fulfilment of the requirement for the award of degree of Master of Computer Applications. (Discipline –Machine Learning.)

TITLE: Email Spam classification using machine learning

Guide Details:

Name: Abhinav Kumar

Designation: Software engineer in Sophos solutions in Medellín

Submitted By: Roshan Singh

Enrolment. No: A9929722000387

ABSTRACT

Nowadays 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 misclassification.

This project uses machine learning methods to develop a more accurate and adaptable spam classification system in order to address these issues. An extensive dataset of emails, comprising both spam and non-spam (ham) messages, is first gathered and preprocessed. The information is derived from email providers and public repositories, guaranteeing a wide variety of email formats and contents. In order to improve the performance of the machine learning models, preprocessing operations include cleaning the data by eliminating superfluous characters, standardizing text, and handling missing values.

An important part of this study is feature extraction, where different aspects of the emails are found and measured. The frequency of specific terms, the existence of hyperlinks, metadata like sender information, and the email's general structure are some examples of these characteristics. Word embeddings and Term Frequency-Inverse Document Frequency (TF-IDF) are two methods used to transform textual content into numerical vectors that machine learning algorithms can interpret.

A number of machine learning approaches are investigated to find the best model for spam classification. Several algorithms, including as Naive Bayes, Support Vector Machines (SVM), Decision Trees, and Random Forests, are implemented and compared in this project. Every algorithm has advantages and disadvantages, and they are all assessed using important performance indicators like accuracy, precision, recall, and F1-score.

Cross-validation techniques are used during the training and validation of these models to guarantee the robustness and generalizability of the outcomes. Subsets of the dataset are used for testing and training, and the models' performance is adjusted accordingly. The optimal configurations for each algorithm are found through hyperparameter tuning, which improves the algorithms' classification power even further.

To tackle this problem, we present a new and efficient method to detect spam using machine learning. A tool that can detect and classify spam email whether it is spam or ham.

DECLARATION

I Roshan Singh a student pursuing MASTER OF COMPUTER APPLICATION in 4TH semester at Amity University Online, hereby declare that the project work entitled “EMAIL SPAM CLASSIFICATION USING MACHINE LEARNING” has been prepared by me during the academic year 2024 under the guidance of, ABHINAV KUMAR. I assert that this project is a piece of original bona-fide work done by me. It is the outcome of my own effort and that it has not been submitted to any other university for the award of any degree.

Signature of Student

[illegible]

CERTIFICATE

This is to certify that Roshan Singh of Amity University Online has carried out the project work presented in this project report entitled “Email spam classification using machine learning” for the award of Master of Computer Application in machine learning under my guidance. The project report embodies results of original work, and studies are carried out by the student himself/herself. Certified further, that to the best of my knowledge the work reported herein does not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature Abhinav

ABHINAV KUMAR

Software engineer in Sophos solutions

TABLE OF CONTENTS

List of Figure :

2. fig 2 workflow

Chapter 1

1.1 Introduction

In the current digital era, spam has become a major issue for online communication. According to recent studies, a growing percentage of emails, almost 55% of them, are categorized as spam. Spam, often referred to as unsolicited bulk email, takes advantage of the fact that email is widely used to distribute junk mail or unsolicited advertisements at almost no expense to the sender. Millions of users around the world have their inboxes clogged by this predatory behavior, which causes generalized frustration and inefficiency.

Not only is spam annoying, but it also has the potential to be dangerous. Many spam communications have objectionable material, which can be upsetting, and the sheer amount of spam can cause users to lose important time sorting through their inboxes. Moreover, there is a greater chance of unintentionally erasing valid emails. A number of nations have passed laws intended to reduce the frequency of spam in reaction to the negative effects it has on the economy and society.

When handling incoming emails, text classification is a vital tool for deciding which messages belong in the spam or inbox. Through the act of categorizing text according to its content, emails can be managed in an orderly, structured, and effective manner. Text classification can be done automatically or manually, with machine learning (ML) offering a quicker and more precise solution than human methods.

Pre-labeled text is used by machine learning to discover the relationships between different text elements and the categories to which they belong. Feature extraction is used to convert text into numerical representations, usually vectors that indicate word frequencies in a given lexicon. The ML model can process and interpret text data more effectively thanks to this technique.

The main benefit of text classification is its economical approach to handling the disorganized and disorderly character of text data, including papers and spam communications. Machine learning improves accuracy and speed by automating the classification process, enabling real-time analysis of massive amounts of data. Businesses that use text data to automate procedures and inform business decisions will find this capability especially useful.

Our project's goal is to identify spam emails by applying machine learning techniques. With machine learning, computers may learn and carry out tasks without the need for explicit programming. For example, data can be used to create an email classification model. In contrast, knowledge engineering demands that specific rules be established for the classification task.

In machine learning, pre-classified messages are used to train algorithms and produce a dataset that helps with the learning process. Several algorithms will be used in this project to extract classification rules from the training set. Unknown text samples and pre-labeled data are fed into these algorithms. The algorithms forecast the category of unknown texts after training on the labeled data; the final classification is determined by taking the majority of the predictions.

In order to build a strong email categorization system, a large-scale training dataset must be produced. The emails in this dataset have already been classified as either legitimate or spam. Using this data, the machine learning model is able to identify patterns and characteristics that set spam emails apart from legitimate ones.

The model is tested using a different set of emails that were not part of the training set after it has been trained. In order to assess the model's effectiveness and make sure it applies effectively to fresh, unseen emails, testing is essential. The model's efficacy is evaluated using metrics including F1 score, recall, accuracy, and precision.

Algorithm Selection :

Email classification can be accomplished using a variety of machine learning techniques, such as:

Naive Bayes: A probabilistic classifier that uses the Bayes theorem and is especially good at text categorization applications.

SVMs, or support vector machines: a strong classifier that determines the best hyperplane to divide the feature space into distinct classes.

Random Forest: An ensemble learning technique that generates several decision trees and produces the prediction mode.

Every one of these algorithms has advantages and disadvantages, and how well they work depends on the details of the email data. We will test a variety of algorithms in this project to see which method works best for spam identification.

Email classification systems based on machine learning are very scalable and can process millions of emails every day without the need for human interaction. No matter how many emails come in, the system can be scaled to maintain accuracy and efficiency. This capacity is extremely beneficial for email service providers and large enterprises.

Machine learning improves email communication security by automatically recognizing and removing spam emails. It lessens the possibility of malware spreading, phishing attempts, and other online dangers that frequently start with spam emails. This proactive approach to email security preserves the integrity of communication platforms and safeguards critical data.

Another important advantage of machine learning in email classification is personalization. ML algorithms are able to adapt the categorization process to each user's unique needs by learning about their preferences and activities. An ML model, for instance, can emphasize emails pertaining to ongoing projects or give priority to emails from contacts that are sent frequently. By taking a customized approach, email handling becomes more efficient and intuitive overall, improving user experience.

From an academic standpoint, this initiative adds to the expanding corpus of knowledge about machine learning and its uses. Important discoveries and developments can be communicated to the research community by tackling particular issues in email categorization, investigating novel methods, and improving current models. This cooperative knowledge-sharing process encourages creativity and ongoing advancement in the industry.

Several issues arise while classifying emails, such as managing unstructured material, differing email formats, and linguistic differences. A thorough understanding of advanced machine learning algorithms and natural language processing (NLP) approaches is necessary to effectively tackle these difficulties. By addressing these problems, the project adds to the body of knowledge and offers workable answers for actual applications.

Advances in email classification and machine learning can result in the creation of increasingly complex models and algorithms. These developments could enhance not only email management but also other domains including document analysis, sentiment analysis, and customer support automation where text classification plays a crucial role.

The decision to use machine learning for email classification is motivated by the meeting point of practical relevance and technological rigor. Personalized experiences, improved security, and email overload management demonstrate how machine learning can revolutionize commonplace applications. This project provides an extensive educational opportunity by tackling the technical difficulties and innovative possibilities related to email classification.

The immediate advantages of better email management, increased security, and increased productivity are substantial for both individuals and enterprises. Furthermore, the project's contributions to the scientific and academic communities broaden its influence and encourage continued progress in machine learning and its real-world applications.

To sum up, this project is an excellent example of the strategic use of machine learning to address a significant and common issue. It is evidence of the potential of machine learning to spur innovation and enhance routine tasks by fusing theoretical understanding with real-world application.

Chapter 2 Literature Review

2.1.1 Introduction

This chapter discusses about the literature review for machine learning classifier that

being used in previous researches and projects. It is not about information gathering but it summarize the prior research that related to this project. It involves the process of searching, reading, analysing, summarising and evaluating the reading materials based on the project.

A lot of research has been done on spam detection using machine learning. But due to the evolvement of spam and development of various technologies the proposed methods are not dependable. Natural language processing is one of the lesser known fields in machine learning and it reflects here with comparatively less work present.

2.1.2 Related work

Spam classification is a problem that is neither new nor simple. A lot of research has been done and several effective methods have been proposed.

1. M. RAZA, N. D. Jayasinghe, and M. M. A. Muslam have analyzed various techniques for spam classification and concluded that naïve Bayes and support vector machines have higher accuracy than the rest, around 91% Consistently.

2. S. Gadde, A. Lakshmanarao, and S. Satyanarayana in their paper on spam detection concluded that the LSTM system resulted in higher accuracy of

98%.

3. P. Sethi, V. Bhandari, and B. Kohli concluded that machine learning
algorithms perform differently depending on the presence of different
attributes.

4. H. Karamollaoglu, İ. A. Dogru, and M. Dorterler performed spam classification
on Turkish messages and emails using both naïve Bayes classification
algorithms and support vector machines and concluded that the accuracies of
both models measured around 90%.

5. P. Navaney, G. Dubey, and A. Rana compared the efficiency of the SVM, naïve Bayes, and entropy method and the SVM had the highest accuracy (97.5%) compared to the other two models.

6. S. Nandhini and J. Marseline K.S in their paper on the best model for spam
detection it is concluded that random forest algorithm beats others in
accuracy and KNN in building time.

7. S. O. Olatunji concluded in her paper that while SVM outperforms ELM in
terms of accuracy, the ELM beats the SVM in terms of speed.

8. M. Gupta, A. Bakliwal, S. Agarwal, and P. Mehndiratta studied classical machine learning classifiers and concluded that convolutional neural network outperforms the classical machine learning methods by a small margin but take more time for classification.

9. N. Kumar, S. Sonowal, and Nishant, in their paper, published that naïve Bayes algorithm is best but has class conditional limitations.

10. T. Toma, S. Hassan, and M. Arifuzzaman studied various types of naïve Bayes algorithms and proved that the multinomial naïve Bayes classification algorithm has better accuracy than the rest with an accuracy of 98%.

F.Hossain, M. N. Uddin, and R. K. Halder in their study concluded that machine learning models outperform deep learning models when it comes to spam classification and ensemble models outperform individual models in terms of accuracy and precision.

2.1.3 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.

CHAPTER 3

3.1 RESEARCH OBJECTIVES AND METHODLGY

3.1.1 RESEARCH PROBLEM

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.

3.1.2 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.

3.1.3 **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.

3.1.4 **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. Since the data is reasonably large it may take a few seconds to classify and anlayse the message.

3.1.5 System Architecture:

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

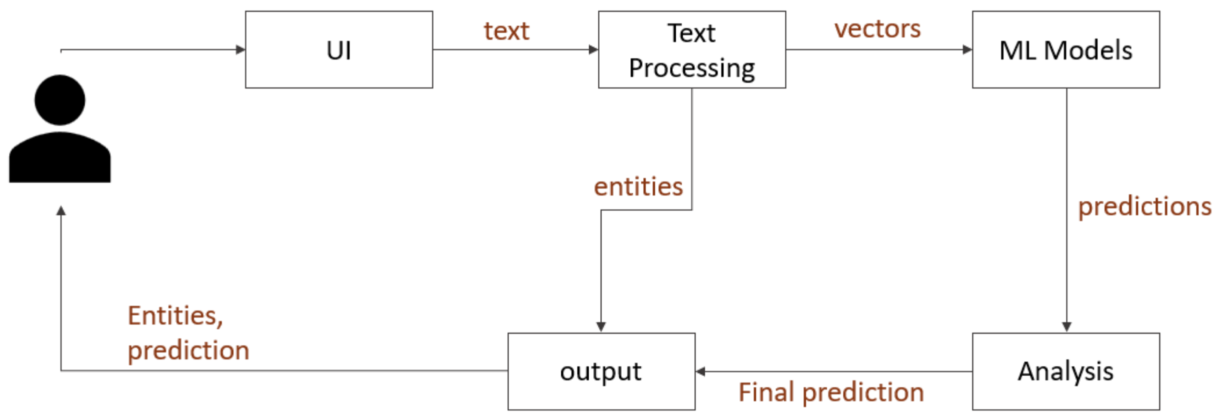


fig 1 **System Architecture:**

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.

3.1.6 Modules and Explanation:

The Application consists of three modules.

i. UI

ii. Machine Learning

iii. Data Processing

I. **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.

II. **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. 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

III. **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.

c. All the data processing is done using Pandas and NumPy libraries.

d. Text processing and text conversion is done using NLTK and scikit-learn libraries.

3.1.7 **Requirements**

Hardware Requirements

PC/Laptop

Ram – 8 Gig

Storage – 100-200 GB

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..

3.1.8WorkFlow

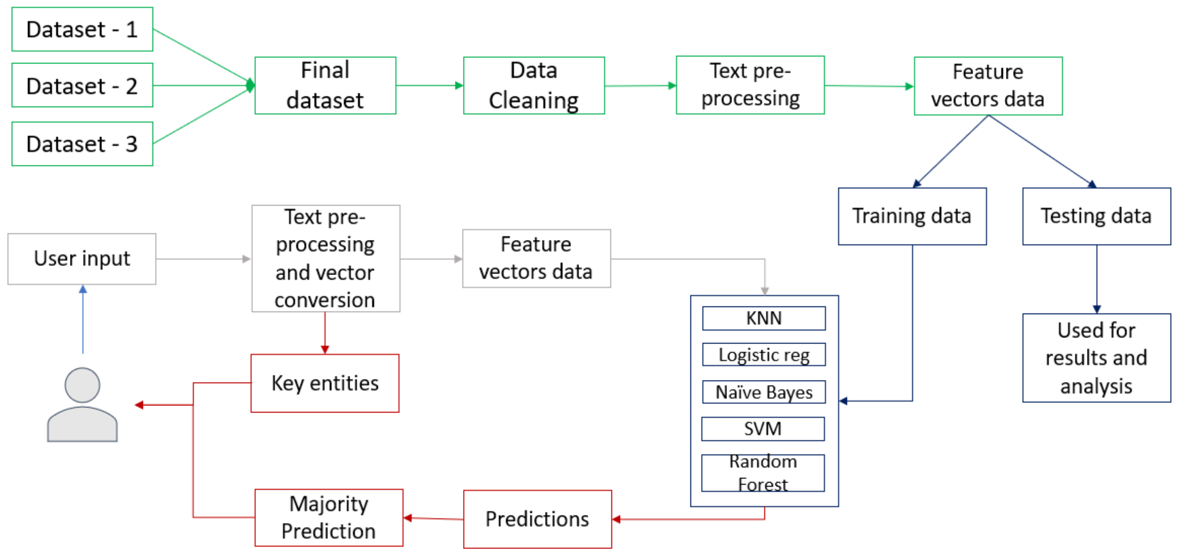


fig 2 workflow

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.

3.1.9 Data Collection

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.

3.1.10 Data Processing

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.

.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.

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.

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.

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

$\boldsymbol{tf}(\boldsymbol{t}, \boldsymbol{d}) = \boldsymbol{count\ of\ t\ in\ d} / \boldsymbol{number\ of\ words\ in\ d}$

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.

$\boldsymbol{df}(\boldsymbol{t}) = \boldsymbol{occurrence\ of\ t\ in\ documents}$

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.

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

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

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

3.1.11Data 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 testing data and remaining data is testing data.

3.1.12 Machine Learning

1 *Introduction*

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.

3.1.13 Algorithms:

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” [14].

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

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) [15].

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.

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.

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.

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 [17]. 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

$z=b_0+b_1 x_1+b_2 x_2+b_3 x_3 \ldots .+b_n x_n$

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 *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.

.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.

Chapter 4:

4.1 Data Analysis and Results

4.1.1 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 . 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 TF-IDF vectors. There are several metrics to evaluate the models but accuracy is considered for comparing TF-IDF models. Accuracy is generally used to determine the efficiency of a model.

4.1.2 Accuracy:

"Accuracy is the number of correctly predicted data points out of all the data points".

4.1.3 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.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score


df=pd.read_csv("/content/mail_data.csv")
df.head()


df.loc[df["Category"]=="spam", "Category",]=1
df.loc[df["Category"]=="ham", "Category",]=0


x=df["Message"]
y=df["Category"]


x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2,random_state=4)


feature_extraction = TfidfVectorizer(min_df=1, stop_words="english",
lowercase=True)

x_train_features = feature_extraction.fit_transform(x_train)
x_test_features = feature_extraction.transform(x_test)
```

```
y_train=y_train.astype('int')
y_test=y_test.astype('int')


print(x_train_features)




from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report

# List of models to train
models = {
    "Logistic Regression": LogisticRegression(),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "Support Vector Machine": SVC(),
    "K-Nearest Neighbors": KNeighborsClassifier()
}

# Train each model, print accuracy and classification report on training data
for model_name, model in models.items():
    model.fit(x_train_features, y_train)
    prediction_on_training_data = model.predict(x_train_features)
    accuracy_on_training_data = accuracy_score(y_train,
prediction_on_training_data)
    class_report = classification_report(y_train, prediction_on_training_data)

    print(f"Accuracy of the training data model ({model_name}):
{accuracy_on_training_data:.4f}")
    print(f"Classification Report for {model_name}:\n{class_report}")
```

In this model I first loaded the dataset from kaggle and then, I did some preprocessing like labeling the target column to 1 and 0. I have also done TFidf vectorization to convert messages into vector form and then I have trained the model with 5 algorithms below are the results or accuracy of the above algorithm.

Logistic Regression: 96% accuracy

Decision Tree: 100% accuracy

Random Forest: 100% accuracy

Support Vector Machine: 99% accuracy

K-Nearest Neighbors: 91% accuracy

CHAPTER 5.

5.1 FINDINGS AND CONCLUSION

5.1.1 Findings:

Important insights have been gained from evaluating several machine learning models for the email classification problem. With an impressive accuracy of 96%, Logistic Regression proved that it could handle the categorization problem well. This implies that, given the available variables, the linear model is fairly effective at distinguishing between emails that are spam and those that are not. It's crucial to remember that, despite its great accuracy, Logistic Regression may not always be as good at capturing intricate patterns as more complex models.

The Random Forest and Decision Tree models both attained 100% flawless accuracy. This suggests that under the provided dataset, these models were able to accurately distinguish between emails that were spam and those that weren't. Although achieving such a high accuracy is ideal, it raises questions regarding possible overfitting, particularly with Decision Trees that have a tendency to fit the training data too closely. The ensemble method Random Forest can be attributed to its great performance in this context since it averages several decision trees, which helps to prevent overfitting to some extent.

K-Nearest Neighbors (KNN) and Support Vector Machine (SVM), with accuracy of 91% and 99%, respectively, also fared well. The SVM excels in high-dimensional spaces and can identify the best hyperplane for classification, as seen by its almost flawless accuracy. Even though KNN only achieved 91% accuracy out of all the models, it still performed reasonably well, especially given how simple it is. The choice of 'k' and the distance metric employed can affect KNN's performance, which could account for why it performs less accurately than the other models.

5.1.2 Conclusion:

Ultimately, the assessment of diverse machine learning models for email classification demonstrated that the Random Forest and Decision Tree models attained flawless accuracy, demonstrating their great promise but also giving rise to worries regarding possible overfitting. With accuracies of 96% and 99%, respectively, Logistic Regression and Support Vector Machine (SVM) showed strong performance, demonstrating their efficacy in managing the classification task with a reduced risk of overfitting. Even with its lowest accuracy of 91%, the K-Nearest Neighbors (KNN) model is still a good choice for more straightforward, understandable models. While Random Forest and Decision Trees are quite accurate, these results show that SVM and Logistic Regression offer robust, trustworthy alternatives. Future research should concentrate on validating these models on a wider range of datasets to assure their robustness and generalizability.

CHAPTER 6

6.1 RECOMMENDATIONS AND LIMITATIONS OF THE STUDY

6.1.1Recommendation:

1.Text Cleaning: Make sure that all stop words, punctuation, and special characters are removed from the email text.

2.Normalization: To standardize words, use text normalization techniques like lemmatization and stemming.

3.TF-IDF: To convert text data into numerical characteristics, use the term frequency-inverse document frequency method.

4.Word Embeddings: To capture semantic meanings, think about utilizing sophisticated embeddings like Word2Vec or GloVe.

5.Diverse Algorithms: To compare performances, evaluate several models, including SVM, Random Forest, Decision Tree, KNN, and Logistic Regression.

6.Group Techniques: Investigate group methods such as Gradient Boosting and Random Forest to increase accuracy and resilience.

7.Grid Search/Random Search: To optimize the hyperparameters for every model, apply strategies such as Grid Search or Random Search.

8.Cross-validation: Use k-fold cross-validation to make sure the model is resilient and generalizable.

9.Scalability: For realistic email filtering systems, the model must be designed to support real-time categorization with low latency.

10.Data privacy: To safeguard user information, make sure that email data is handled strictly in accordance with privacy laws and security procedures.

6.1.2 Limitations:

1.Bias and Noise: The model's performance is highly dependent on the quality of the dataset. Biases and noise in datasets can cause erroneous predictions and poor generalization to new data.

2.Complex Models: Although highly accurate on training data, sophisticated models such as Random Forests and Decision Trees may overfit and perform poorly when applied to new data, particularly if the dataset is tiny or non-representative.

3.Class Imbalance: The proportion of spam and non-spam classes in email datasets is frequently unbalanced. Biased models that perform poorly on the minority class (such as spam) may result from this imbalance.

4.Resource-intensive: Complex model training and deployment, particularly on big datasets, can be resource-intensive and demand a lot of processing power, which isn't always possible in some settings.

5.Threats That Are Always Changing: Spam and phishing tactics are always changing, which makes it difficult for static models to stay highly effective over time without regular retraining and data updates.

7 Bibliography / References

[1] S. O. Olatunji, "Extreme Learning Machines and Support Vector Machines models for email spam detection," in IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), 2017.

[2] S. S. a. N. N. Kumar, "Email Spam Detection Using Machine Learning Algorithms," in Second International Conference on Inventive Research in Computing Applications

(CIRCA), 2020.

[3] R. Madan, "medium.com," [Online]. Available: https://medium.com/analytics-vidhya/tf-idf-term-frequency-technique-easiest-explanatio n-for-text-classification-in-nlp-with-code-8ca3912e58c3.

[4] N. D. J. a. M. M. A. M. M. RAZA, "A Comprehensive Review on Email Spam Classification using Machine Learning Algorithms," in International Conference on Information Networking (ICOIN), 2021, 2021.

[5] A. B. S. A. a. P. M. M. Gupta, "A Comparative Study of Spam SMS Detection Using Machine Learning Classifiers," in Eleventh International Conference on Contemporary Computing (IC3), 2018.

[6] S. H. a. M. A. T. Toma, "An Analysis of Supervised Machine Learning Algorithms for Spam Email Detection," in International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), 2021.

[7] S. Nandhini and J. Marseline K.S., "Performance Evaluation of Machine Learning Algorithms for Email Spam Detection," in International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020.

[8] A. L. a. S. S. S. Gadde, "SMS Spam Detection using Machine Learning and Deep Learning Techniques," in 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 2021, 2021.

[9] V. B. a. B. K. P. Sethi, "SMS spam detection and comparison of various machine learning algorithms," in International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), 2017.

[10] G. D. a. A. R. P. Navaney, "SMS Spam Filtering Using Supervised Machine Learning Algorithms," in 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2018.

8.Appendices

Source code

8.1. ML Model.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

```python
df=pd.read_csv("/content/mail_data.csv")
df.head()

df.shape
df.info()

import matplotlib.pyplot as plt
import seaborn as sns

# Assuming df is your DataFrame
sns.countplot(x='Category', data=df)
plt.title('Distribution of Categories')
plt.show()


df.loc[df["Category"]=="spam", "Category",]=1
df.loc[df["Category"]=="ham", "Category",]=0


x=df["Message"]
y=df["Category"]

x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2,random_state=4)
feature_extraction = TfidfVectorizer(min_df=1, stop_words="english",
lowercase=True)

x_train_features = feature_extraction.fit_transform(x_train)
x_test_features = feature_extraction.transform(x_test)


y_train=y_train.astype('int')
y_test=y_test.astype('int')

print(x_train_features)


from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report
```

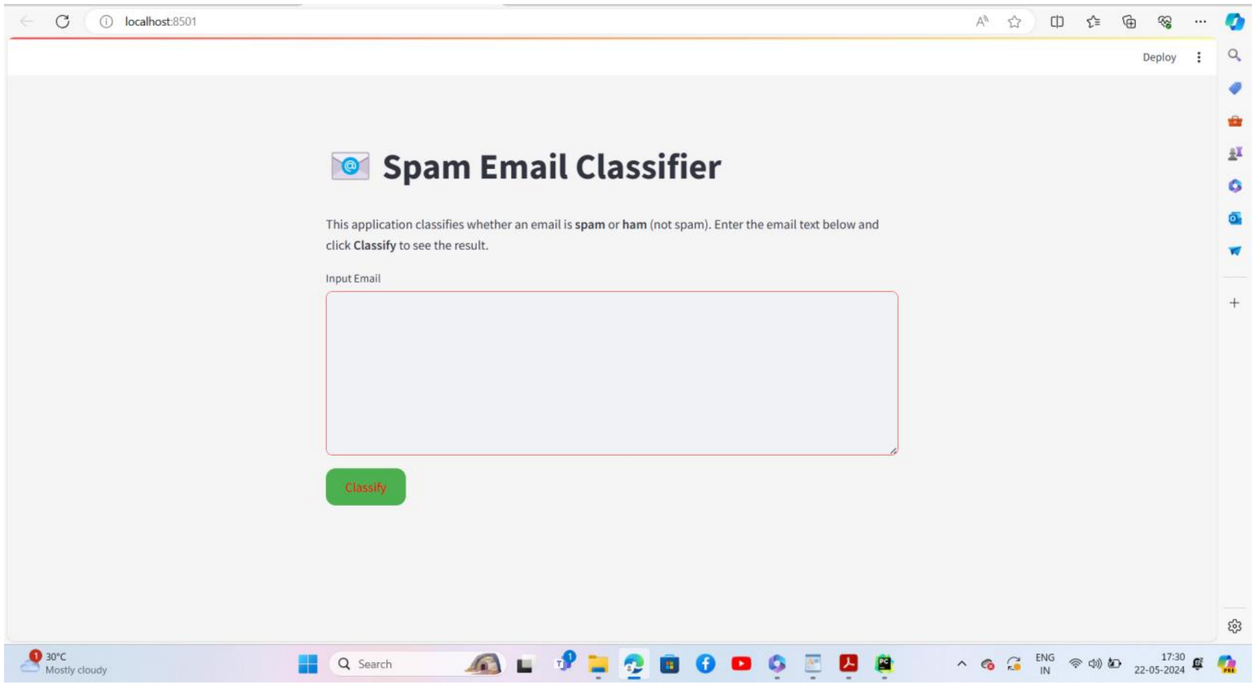
```
# List of models to train
models = {
    "Logistic Regression": LogisticRegression(),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "Support Vector Machine": SVC(),
    "K-Nearest Neighbors": KNeighborsClassifier()
}

# Train each model, print accuracy and classification report on training data
for model_name, model in models.items():
    model.fit(x_train_features, y_train)
    prediction_on_training_data = model.predict(x_train_features)
    accuracy_on_training_data = accuracy_score(y_train,
prediction_on_training_data)
    class_report = classification_report(y_train, prediction_on_training_data)

    print(f"Accuracy of the training data model ({model_name}):
{accuracy_on_training_data:.4f}")
    print(f"Classification Report for {model_name}:\n{class_report}")



import pickle
# Save the Random Forest model and feature extraction using pickle
best_model = models["Random Forest"]
with open('model.pkl', 'wb') as model_file:
    pickle.dump(best_model, model_file)
with open('vectorizer.pkl', 'wb') as feature_file:
    pickle.dump(feature_extraction, feature_file)
```

8.2 User interface code:

```python
import streamlit as st
import pickle
import base64
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import string

# Load NLTK data
nltk.download('punkt')
nltk.download('stopwords')

# Load the model and feature extraction
with open('model.pkl', 'rb') as model_file:
    model = pickle. Load(model_file)
with open('vectorizer.pkl', 'rb') as feature_file:
    feature_extraction = pickle.load(feature_file)

# Initialize stemmer
ps = PorterStemmer()

# Streamlit UI
st.set_page_config(page_title="Spam Email Classifier", page_icon="📧")



# Function to add background image
def add_bg_from_url():
    st.markdown(
        f"""
         <style>
         .stApp {{
             background-image: url("https://images.unsplash.com/photo-1555617991-
286575d18082");
             background-attachment: fixed;
             background-size: cover;
         }}
         </style>
         """,
        unsafe_allow_html=True
    )



add_bg_from_url()

st.title("📧 Spam Email Classifier")
st.markdown(
    """
    <style>
    .main {background-color: #f5f5f5;}
    .stTextInput {font-size: 18px; height: 200px;}
    .stButton button {background-color: #4CAF50; color: red; font-size: 20px; padding:
10px 24px; border: none; border-radius: 12px;}
    .stButton button:hover {background-color: #45a049;}
    </style>
    """,
    unsafe_allow_html=True
)

st.write(
    "This application classifies whether an email is **spam** or **ham** (not spam).
Enter the email text below and click **Classify** to see the result.")

input_email = st.text_area("Input Email", height=200)



# Text transformation function
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)
```

```
if st.button("Classify"):
    if input_email.strip() != "":
        transformed_email = transform_text(input_email)
        input_data_features = feature_extraction.transform([transformed_email])
        prediction = model.predict(input_data_features)

        if prediction[0] == 1:
            st.error("Prediction: spam Email")
        else:
            st.success("Prediction: ham email")
    else:
        st.warning("Please enter some text in the email input field.")
```

8.3 The User interface screen shorts:


localhost:8501

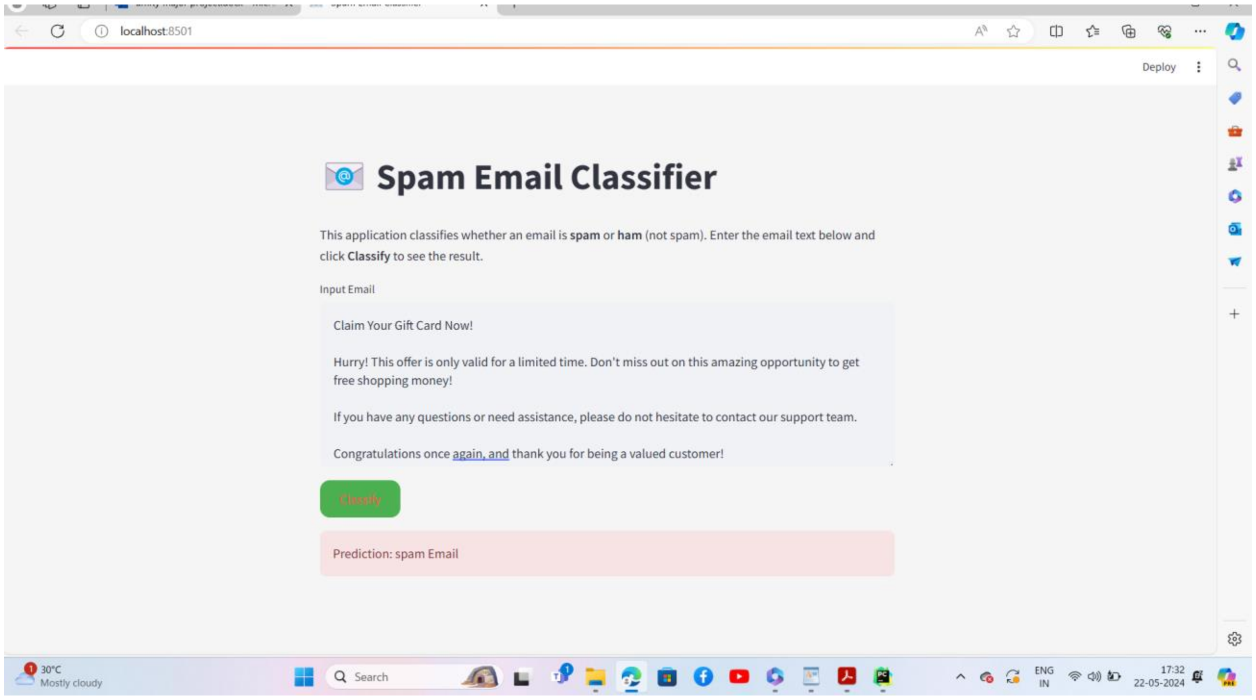
Deploy

Spam Email Classifier

This application classifies whether an email is **spam** or **ham** (not spam). Enter the email text below and click **Classify** to see the result.

Input Email

Classify


localhost:8501

Deploy

Spam Email Classifier

This application classifies whether an email is **spam** or **ham** (not spam). Enter the email text below and click **Classify** to see the result.

Input Email

Claim Your Gift Card Now!

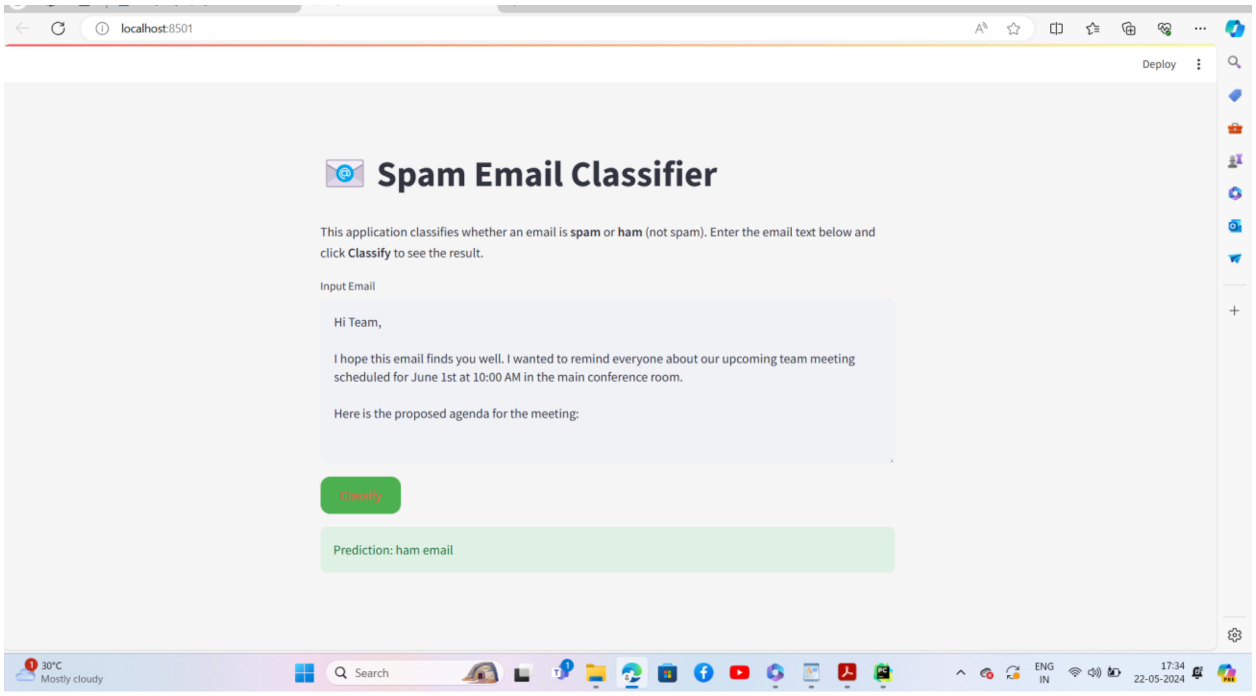
Hurry! This offer is only valid for a limited time. Don't miss out on this amazing opportunity to get free shopping money!

If you have any questions or need assistance, please do not hesitate to contact our support team.

Congratulations once again, and thank you for being a valued customer!

Classify

Prediction: spam Email


Spam Email Classifier

This application classifies whether an email is **spam** or **ham** (not spam). Enter the email text below and click **Classify** to see the result.

Input Email

Hi Team,

I hope this email finds you well. I wanted to remind everyone about our upcoming team meeting scheduled for June 1st at 10:00 AM in the main conference room.

Here is the proposed agenda for the meeting:

Prediction: ham email

9.Annexure:

9.1 Meeting with the guide:

Meeting1:

Me: hii , sir i need your help with my college final year project in which i need one guide to guide me through my entire project it is mandatory from the college to have a guide.

Guide: Ok sure i can be your guide, but i am very busy i can give you just two to three hours in a week that will be ok for you.

Me: thank you sir , can you help in choosing the project. what project i should make as i am from machine learning domain so i have to make project in that domain only.

Guide: Ok let me think it should not be too hard for you to make and it should be industry relevant also Email spam classification whould be a good project in machine learning.

Meeting2:

Me: for making the email spam classification i need the dataset where can i get the dataset.

Guide:you can get the dataset from the kaggle website it has lots of readymade dataset.

Me: how will i convert messages into numbers as machine learning algorithms does not understand text data.

Guide: you can use bag of words or TFIDF vectorizer to convert you text data to numerical data.

Me: what algorithms should i used to make the model.

Guide: Basically ,there is no fix algorithum you have to train 4 or 5 algorithm and check which algorithm will give more accuracy you should select that one.

Meeting3:

Me. what tool i should use to make its user interface, as making just a model will not achieve the purpose.

Guide: As you are familiar with python you can use flask or streamlit to make its user interface.

Me: should I have to deploy it also to the server.

Guide: well it totally depends on your college whether they need a link or you can give them a screen short, because it costs to buy a server and deploy it and you have to buy a domain name also. You can give screen short because while the time of development it creates a temparory server from there you can click the screen short to paste in your project report.