

A Project Report
On
Email Spam Detection Model

*Submitted in partial fulfillment of the
requirement for the award of the degree of*

BACHELOR OF TECHNOLOGY



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

BTECH CSE

Session 2023-24
in

Name of discipline

By
Noman Ali (21SCSE1010302)
Abuzar (21SCSE1010356)
Amit Verma (21SCSE1010873)

Under the guidance of Mr.
Akhilesh Kumar Singh

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING**

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

Jan , 2024



**SCHOOL OF COMPUTING SCIENCE AND
ENGINEERING**
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled “.....**EMAIL SPAM DETECTION USING ML**.....” in partial fulfillment of the requirements for the award of the B. Tech. (Computer Science and Engineering) submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of SEP, 2023 to, JAN, 2024 under the supervision of Prof. Akhilesh Kumar Singh, Department of Computer Science and Engineering, of School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Student Names (Admission No.)

NOMAN ALI (21SCSE1010302)

ABUZAR (21SCSE1010356)

AMIT VERMA (21SCSE1010873)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Guide Names

Designation

CERTIFICATE

This is to certify that Project Report entitled“... **EMAIL SPAM DETECTION USING MACHINE LEARNING.....**” which is submitted by in partial fulfillment of the requirement for the award of degree B. Tech. in Department of**CSE.....** of School of Computing Science and Engineering Department of Computer Science and Engineering Galgotias University, Greater Noida, India is a record of the candidate own work carried outby him/them under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree

Signature of Examiner(s)

Signature of Program Chair

Signature of Dean

Date: Jan , 2024

Place: Greater Noida

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude..... , Department of Computer Science & Engineering, Galgotias University, Greater Noida, India for his constant support and guidance throughout the course of our work. His/Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of , Head, Department of Computer Science & Engineering, Galgotias University, Greater Noida, India for his full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Signature:

Name :

Roll No.:

Date :

Signature:

Name :

Roll No.:

Date :

Signature:

Name :

Roll No.:

Date :

ABSTRACT

Unsolicited emails, popularly referred to as spam, have remained one of the biggest threats to cybersecurity globally. More than half of the emails sent in 2021 were spam, resulting in huge financial losses. The tenacity and perpetual presence of the adversary, the spammer, has necessitated the need for improved efforts at filtering spam. This study, therefore, developed baseline models of random forest and extreme gradient boost (XGBoost) ensemble algorithms for the detection and classification of spam emails using the Enron1 dataset. The developed ensemble models were then optimized using the grid-search cross-validation technique to search the hyperparameter space for optimal hyperparameter values. The performance of the baseline (un-tuned) and the tuned models of both algorithms were evaluated and compared.

The impact of hyperparameter tuning on both models was also examined. The findings of the experimental study revealed that the hyperparameter tuning improved the performance of both models when compared with the baseline models. The tuned RF and XGBoost models achieved an accuracy of 97.78% and 98.09%, a sensitivity of 98.44% and 98.84%, and an F1 score of 97.85% and 98.16%, respectively. The XGBoost model outperformed the random forest model. The developed XGBoost model is effective and efficient for spam email detection.

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.

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE No
	Abstract	5
	List of Figures	8
	List of Tables	9
1	Introduction	10
2	Literature Review	12
	2.1 introduction	12
	2.2 Related work	12
	2.3 Summary	13
3	Objectives and Scope	14
	3.1 Problem statement	14
	3.2 Objectives	14
	3.3 Project Scope	14
	3.4 Limitations	14
4.	Experimentation and Methods	15
	4.1 Introduction	15
	4.2 System architecture	15
	4.3 Modules and Explanation	15
	4.4 Requirements	17
	4.5 Workflow	17
	4.5.1 Data collection and Description	18
	4.5.2 Data Processing	19
	4.5.2.1 Overall Data Processing	19
	4.5.2.2 Textual Data Processing	19
	4.5.2.3 Feature Vector Processing	20
	4.5.2.3.1 bag of words	20
	4.5.2.3.2 TF-IDF	20
	4.5.3 Data Splitting	23
	4.5.4 Machine Learning	23
	4.5.4.1 Introduction	23
	4.5.4.2 Algorithms	23

	4.5.4.2.1 Naïve bayes Classifier	23
	4.5.4.2.2 Random Forest Classifier	24
	4.5.4.2.3 Logistic Regression	25
	4.5.4.2.4 K-Nearest Neighbors	26
	4.5.4.2.5 Support Vector machines	26
	4.5.5 Experimentation	27
	4.5.6 User Interface(UI)	30
	4.5.7 Working Procedure	31
5	Results and Discussion	32
	5.1 Language Model selection	32
	5.2 Proposed Model	32
	5.3 Comparison	32
	5.4 Summary	34
6	Conclusion and Future Scope	35
	6.1 Conclusion	35
	6.2 Future Work	35
	References	36
	Appendices	38
	A. Source code	38
	B. Screenshots	43

LIST OF FIGURE

Fig No	Title	Pg no
4.1	Architecture	15
4.2	Workflow	17
4.3	Enron Data	18
4.4	Ling spam	18
4.5	Naïve Bayes(Bow vs TF-IDF)	27
4.6	Logistic Regression(Bow vs TF-IDF)	28
4.7	Neighbors vs Accuracy(KNN)	28
4.8	KNN(Bow vs TF-IDF)	29
4.9	Random Forest(trees vs scores)	29
4.10	Random Forest(Bow vs TF-IDF)	29
4.11	SVM(Bow vs TF-IDF)	30
5.1	Bow vs TF-IDF(Cumulative)	32
5.2	Comparision of Models	33

Table number	Table Name	Page no
4.1	Term Frequency	22
4.2	Inverse document frequency	22
4.3	TF-IDF	22
5.1	Models and results	33

CHAPTER - 1

INTRODUCTION

Email spam remains a significant challenge in today's digital age, with the constant influx of unsolicited and potentially harmful messages. To mitigate this issue, machine learning algorithms play a crucial role in identifying and filtering out spam emails. This report explores the application of two popular machine learning algorithms, Naive Bayes and K-Nearest Neighbors (KNN) Classifier, in the context of email spam detection. Many current spam-email-detection techniques rely on a single model, which can be prone to errors and overfitting. Ensemble models, which combine the predictions of multiple models, have the potential to improve the accuracy and robustness of spam detection. While ensemble models have been widely used in other areas of machine learning, they have not been widely applied to spam email detection. Hyperparameters, such as the number of decision trees in a random forest or the regularization parameter in an extreme gradient boost algorithm, can greatly affect the performance of a model. However, finding the optimal hyperparameters are often ignored because it is a time-consuming and computationally expensive task. Therefore, this study is aimed at the hyperparameter optimization of the random forest (RF) and extreme gradient boosting (XGBoost) ensemble algorithms. This is in a bid to enhance the predictive accuracy of the two ensemble models and to determine the best-performing model, robust enough for efficient spam email detection. Ensemble algorithms rely on a combination of predictions from two or more base models to obtain an improved prediction performance on a dataset. In this study:

Spam-email-detection models based on the random forest and XGBoost machine-learning algorithms were developed.

The performances of the ensemble models were optimized through hyperparameter tuning.

The performances of the ensemble models were evaluated and compared before and after hyperparameter tuning.

The convergence time of the models were also established.

The other sections of this study are presented thus: In the second section, a brief highlight of related research on spam email classification and detection was presented. The third section described in detail the dataset and preprocessing techniques, methods, and performance evaluation metrics. The results of the experiments are presented in the fourth section. Finally, a conclusion was drawn with a perspective for further studies in the last section.

Email spam detection involves the use of computational techniques to distinguish between legitimate and unwanted emails. Traditional rule-based methods have limitations, prompting the adoption of machine learning algorithms to enhance accuracy and adaptability.

Machine Learning Algorithms for Email Spam Detection

Naive Bayes Classifier:-

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem. It assumes that the features used for classification are conditionally independent, given the class label. In the context of email spam detection, Naive Bayes calculates the probability of an email being spam or not based on the occurrence of certain words or features in the message.

Advantages:

Simple and computationally efficient.

Performs well on large datasets.

Requires less training data compared to some other algorithms.

Challenges:

Assumes independence of features, which may not hold true in all cases.

Limited ability to capture complex relationships between features.

K-Nearest Neighbors (KNN) Classifier

KNN is a non-parametric and instance-based algorithm that classifies data points based on the majority class among their k-nearest neighbors in the feature space. In email spam detection, KNN can be applied by representing emails as feature vectors and determining their proximity in the feature space.

Data Preprocessing:-

Before applying the algorithms, preprocessing steps such as tokenization, stemming, and feature extraction are essential to convert raw email data into a format suitable for machine learning.

Model Training

The selected algorithms, Naive Bayes and KNN, are trained using labeled datasets containing examples of both spam and non-spam emails.

Model Evaluation

The performance of the models is evaluated using metrics such as accuracy, precision, recall, and F1 score. Cross-validation techniques help ensure robustness and generalizability.

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.

Literature Review

Introduction

Exploratory Data Analysis (EDA) is a crucial step in the data analysis process that involves summarizing, visualizing, and understanding the main characteristics of a dataset. EDA helps analysts and data scientists gain insights into the data, identify patterns, detect outliers, and inform subsequent modeling or analysis steps. Here's a breakdown of the key aspects of Exploratory Data Analysis:

Descriptive Statistics:

Central Tendency: Measures like mean, median, and mode are used to describe the central location of the data.

Dispersion: Variability in the data is examined through measures like range, variance, and standard deviation.

Data Visualization:

Histograms: Visual representation of the distribution of a single variable.

Box Plots: Illustrate the summary statistics (median, quartiles) and identify potential outliers.

Scatter Plots: Display the relationship between two variables to identify patterns or correlations.

Pair Plots: A grid of scatter plots showing relationships between pairs of variables in the dataset.

Data Cleaning:

Handling missing values, outliers, or inconsistent data is an important part of EDA. This involves decisions on imputation, removal, or correction of data points.

Correlation Analysis:

Investigating relationships between variables to understand if and how they are related. Correlation coefficients like Pearson's correlation are often used.

Feature Engineering:

Creating new variables or transforming existing ones to better represent the underlying patterns in the data. This might involve scaling, encoding categorical variables, or creating interaction terms.

Identifying Patterns and Trends:

EDA helps in uncovering hidden patterns, trends, or seasonality in the data that might not be immediately apparent.

Outlier Detection:

Identifying and dealing with outliers that may significantly impact the results of statistical analyses or machine learning models.

Data Distribution Analysis:

Understanding the shape of the data distribution, whether it follows a normal distribution or has skewness, is essential for making informed statistical inferences.

Data Transformation:

Preparing data for analysis by transforming it into a suitable format. This might include log transformations, normalization, or other techniques depending on the characteristics of the data.

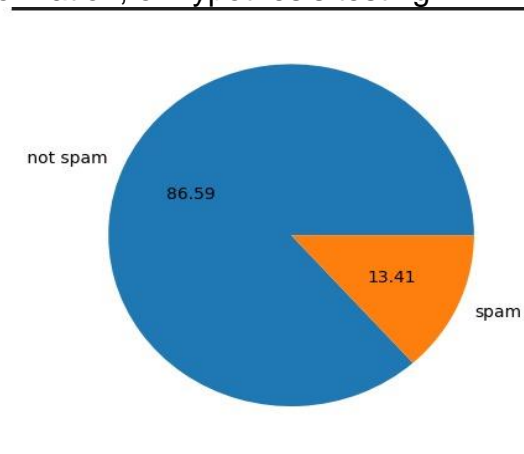
Hypothesis Generation:

Formulating hypotheses about relationships or patterns in the data that can be tested in subsequent analyses.

Interactive Exploration:

Using interactive tools and dashboards to explore the data dynamically, allowing for real-time adjustments and insights.

The ultimate goal of EDA is to provide a comprehensive understanding of the dataset, inform modeling choices, and guide the next steps in the data analysis process. It's an iterative process, where findings from EDA may prompt further data cleaning, transformation, or hypothesis testing.



Methodology

Dataset

The Enron dataset was used in this research because it is the only substantial collection of an actual email that is public and also because of its high level of usage among researchers. The Enron dataset is made up of 6 main directories, each directory has several subdirectories, each containing emails as a single text file. In this study, the Enron1 dataset was used. These emails were converted to a single CSV file by Marcel Wiechmann. The CSV file contained about 33,000 emails. However, during conversion, some of the email messages were not correctly aligned with their labels. The non-aligning messages were removed alongside the orphaned labels through Microsoft Excel. On completion of the removal, the CSV file contained a total of 32,860 emails. Of the total emails, 16,026 (49%) are legitimate emails (ham) and 16,834 (51%) are spam. The CSV file contained five columns labeled message ID, subject, spam/ham, and date. The subject and date columns were not used in this study. The needed columns were readjusted as serial numbers. Column two contains the class label of each of the emails and column three contains the text of each email.

Dataset Processsing:-

To improve the quality of classification, it is important to get rid of unwanted characters or features that constitute noise from the data. The cleaning activities and functions used are presented in Table 1.

```

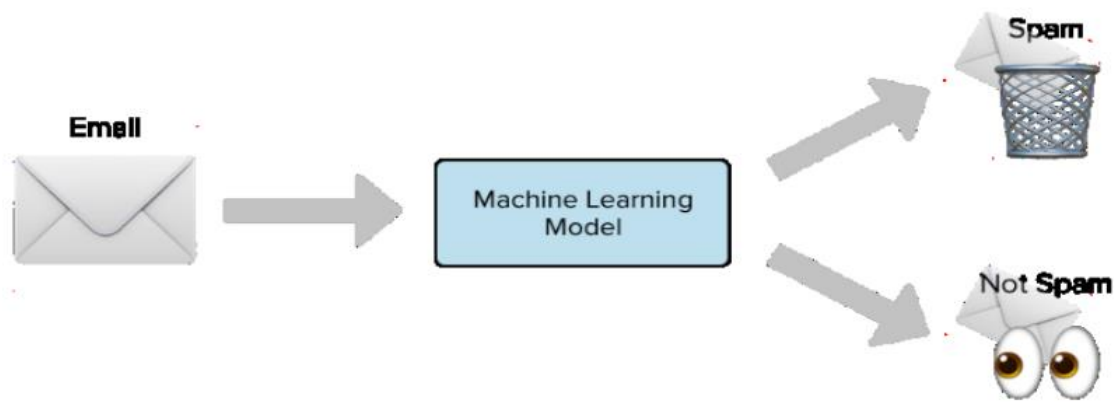
1 class,message,,,
2 ham,"Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...,,,
3 ham,Ok lar... Joking wif u oni,,,,,
4 spam,Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's,,,
5 ham,U dun say so early hor... U c already then say,,,,,
6 ham,"Nah I don't think he goes to usf, he lives around here though",,,
7 spam,"FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, 1.50 to rcv",,,
8 ham,Even my brother is not like to speak with me. They treat me like aids patent.,,,
9 ham,As per your request 'Melle Melle (Oru Minnaminunginte Nuringu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune,,,
10 spam,WINNER!! As a valued network customer you have been selected to receive a 900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.,,,
11 spam,Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030,,,
12 ham,"I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.",,,
13 spam,"SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info",,,
14 spam,"URGENT! You have won a 1 week FREE membership in our 100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403LDNM1A7RW18",,,
15 ham,I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a
16 ham,I HAVE A DATE ON SUNDAY WITH WILL!!!,,
17 spam,"XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap.xxxmobilemovieclub.com?n=QJKGIGHJJCBL",,,
18 ham,Oh k...i'm watching here:),,,
19 ham,Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet,,,
20 ham,Fine if that's the way u feel. That's the way its got a b,,,
21 spam,"England v Macedonia - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87077 Try:WALES, SCOTLAND 4txt/1.20 POBOXox36504M45WQ 16+",,,
22 ham,Is that seriously how you spell his name?,,,
23 ham,I'm going to try for 2 months ha ha only joking,,,
24 ham,So pay first lar... Then when is da stock comin.,,,
25 ham,Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already,,,
26 ham,Ffffffffff. Alright no way I can meet up with you sooner?,,,
27 ham,Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worried. He knows I'm sick when I turn down pizza. Lol,,,
28 ham,Lol your always so convincing,,,
29 ham,Did you catch the bus ? Are you frying an egg ? Did you make a tea? Are you eating your mom's left over dinner ? Do you feel my Love ?,,,
30 ham,"I'm back & we're packing the car now, I'll let you know if there's room",,,

```

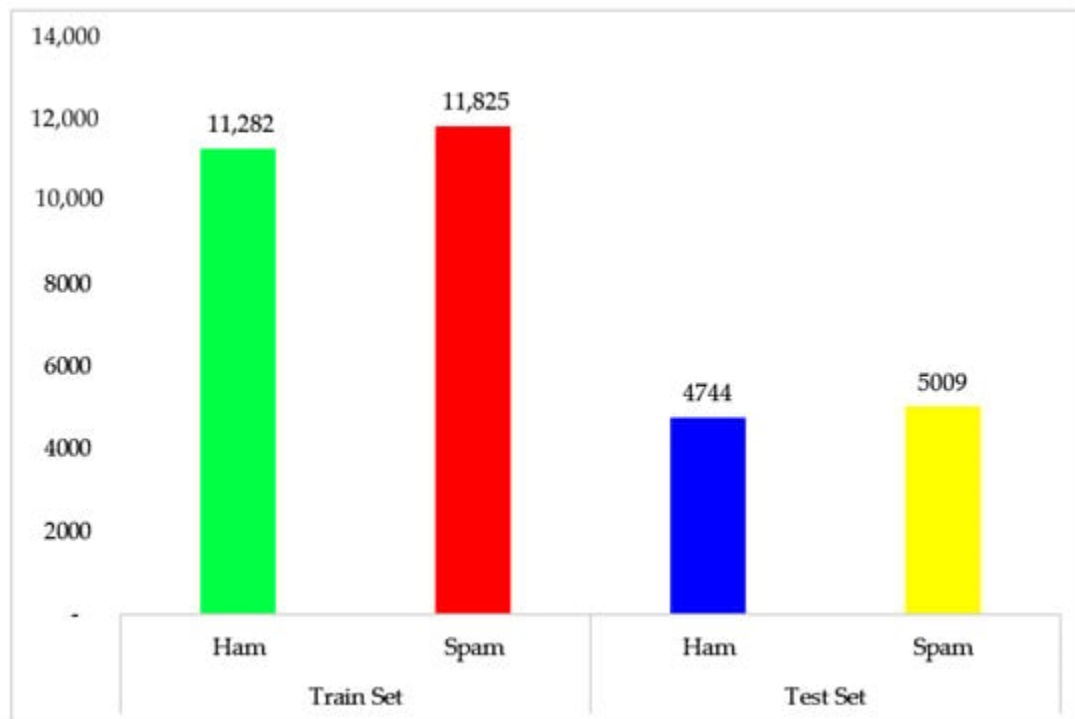
Spam and Ham:

According to Wikipedia “the use of electronic messaging systems to send unsolicited bulk messages, especially mass advertisement, malicious links etc.” are called spam. “Unsolicited” means that those things which you didn’t ask for messages from the sources. So, if you do not know about the sender the mail can be spam. People generally don’t realize they just signed in for those mailers when they download any free services, software or while updating the software.

“Ham” this term was given by Spam Bayes around 2001 and it is defined as “Emails that are not generally desired and is not considered spam”. Machine learning approaches are more efficient, a set of training data is used, these samples are the set of email which are pre classified. Machine learning approaches have a lot of algorithms that can be used for email filtering. These algorithms include “Naïve Bayes, support vector machines, Neural Networks, K-nearest neighbor, Random Forests etc.”



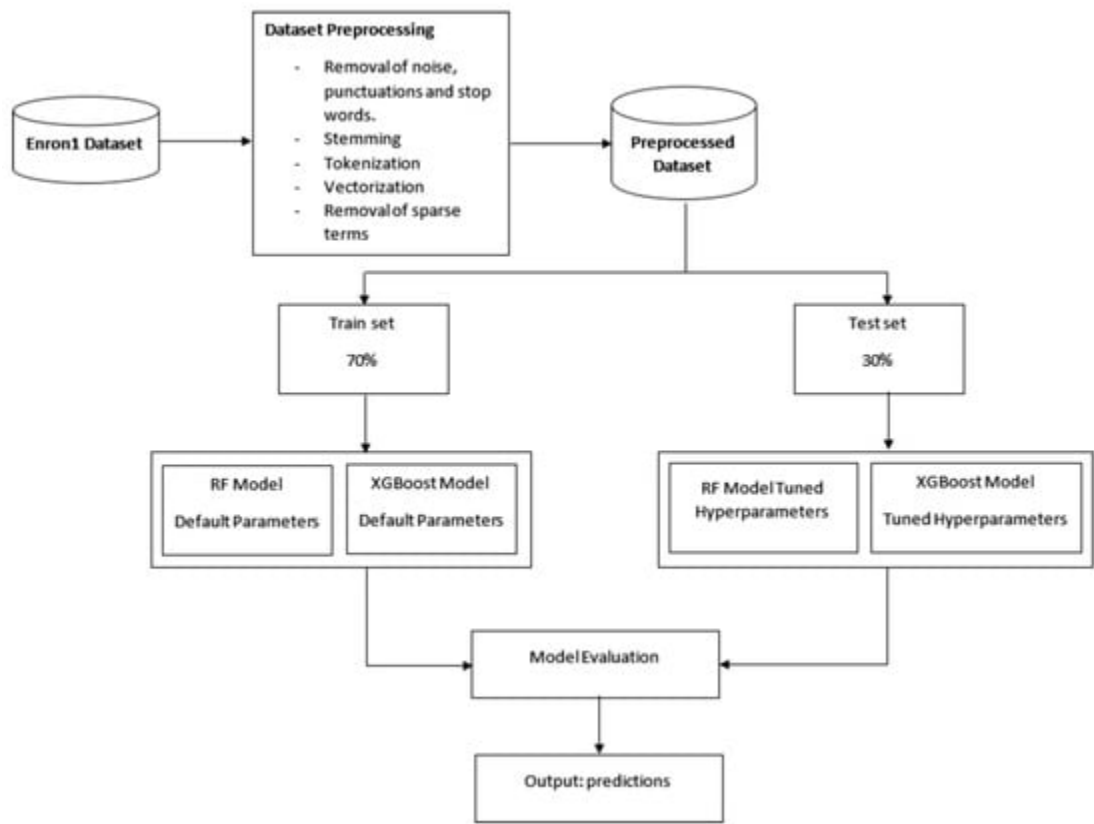
Classification into Spam and non-spam



Distribution of ham and spam emails in train and test dataset.

Baseline models of random forest and extreme gradient boost (XGBoost) were developed by training and testing each model independently with 70% and 30% of the preprocessed dataset. All the parameters were set to their default values during the training and testing of the baseline models. The performance of these models on the test data was recorded as the baseline performance to be improved via hyperparameter

tuning. To reduce the computation time, only the important predictors were passed to the random forest. The random forest has an inbuilt feature for ranking variables or features based on their importance in arriving at a prediction.



The proposed model workflow.

SOME ALGORITHMS THAT ARE USED:

	Algorithm	Accuracy	Precision	Accuracy_scaling_x	Precision_scaling_x	Accuracy_scaling_y	Precision_scaling_y	Accuracy_num_chars	Precisio
0	KN	0.919283	1.000000	0.919283	1.000000	0.919283	1.000000	0.919283	
1	RF	0.973991	1.000000	0.973991	1.000000	0.973991	1.000000	0.973991	
2	ETC	0.978475	1.000000	0.978475	1.000000	0.978475	1.000000	0.978475	
3	xgb	0.978475	0.977273	0.978475	0.977273	0.978475	0.977273	0.978475	
4	GBDT	0.964126	0.974138	0.964126	0.974138	0.964126	0.974138	0.964126	
5	LR	0.974888	0.969231	0.974888	0.969231	0.974888	0.969231	0.974888	
6	AdaBoost	0.967713	0.938462	0.967713	0.938462	0.967713	0.938462	0.967713	
7	DT	0.954260	0.915966	0.954260	0.915966	0.954260	0.915966	0.954260	
8	BgC	0.970404	0.914894	0.970404	0.914894	0.970404	0.914894	0.970404	
9	NB	0.978475	0.914474	0.978475	0.914474	0.978475	0.914474	0.978475	
10	SVC	0.846637	0.418605	0.846637	0.418605	0.846637	0.418605	0.846637	

K-Nearest Neighbors (KNN) is a supervised machine learning algorithm used for classification and regression tasks. It is a simple and intuitive algorithm that is based on the idea that similar data points tend to be close to each other in the feature space. In other words, the class or value of a data point can be predicted by looking at the majority class or average value of its k-nearest neighbors. Here's a step-by-step explanation of how KNN works:

Initialization:

Choose the value of K, which represents the number of neighbors considered when making predictions.

Distance Calculation:

For a given data point that needs a prediction, calculate the distance (typically Euclidean distance) between that point and every other point in the dataset.

Neighbor Selection:

Identify the K data points with the shortest distances to the given data point. These data points are referred to as the "nearest neighbors."

Majority Voting (Classification) or Averaging (Regression):

For classification tasks: Assign the class label to the given data point based on the majority class among its k-nearest neighbors. This is often done through a

voting mechanism.

For regression tasks: Predict the numerical value for the given data point by taking the average of the values of its k-nearest neighbors.

Prediction:

The predicted class label or value for the given data point is the result of the majority voting or averaging process.

Model Evaluation:

Assess the performance of the KNN model using metrics such as accuracy, precision, recall, F1 score, etc. The choice of K can significantly impact the performance of the model.

NAIVE BAYES:

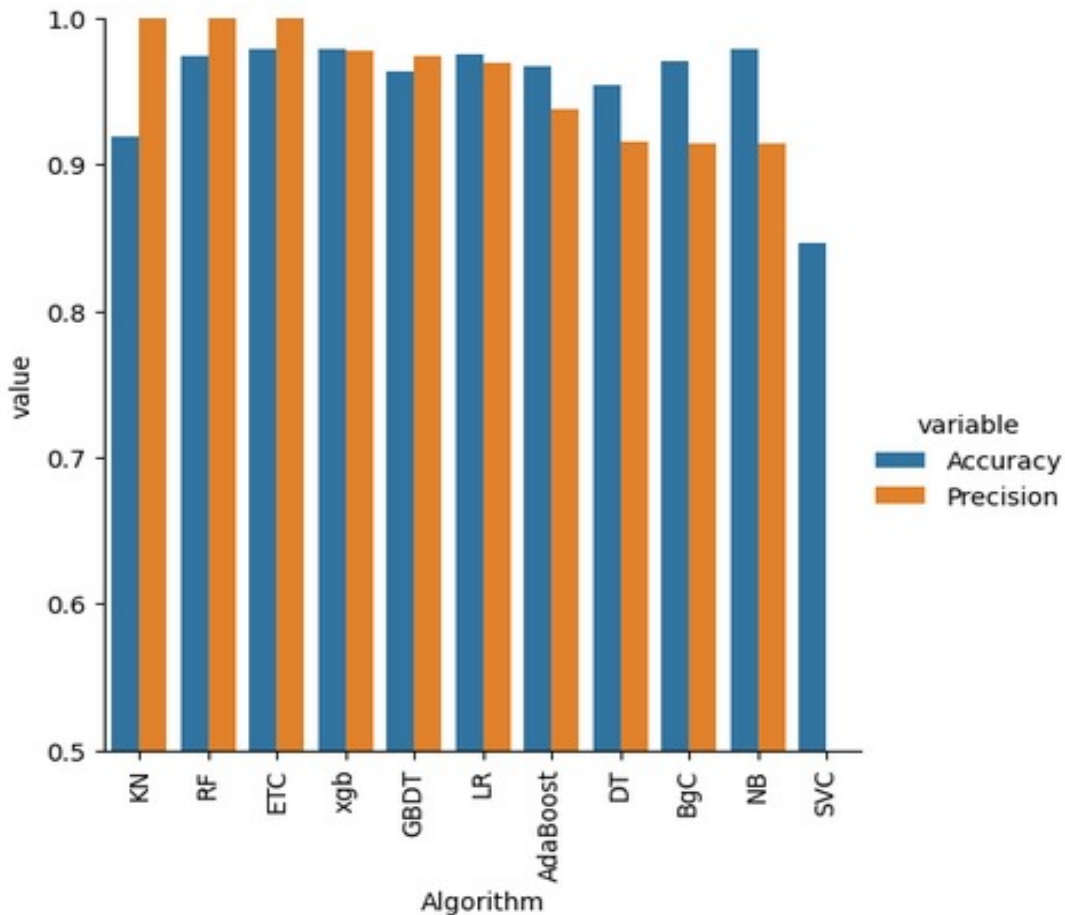
Naive Bayes classifier was used in 1998 for spam recognition. The Naive Bayes classifier algorithm is an algorithm which is used for supervised learning. The Bayesian classifier works on the dependent events and works on the probability of the event which is going to occur in the future that can be detected from the same event which occurred previously. Naïve Bayes was made on the Bayes theorem which assumes that features are autonomous of each other. Naïve Bayes classifier technique can be used for classifying spam emails as word probability plays main role here. If there is any word which occurs often in spam but not in ham, then that email is spam. Naive Bayes classifier algorithm has become a best technique for email filtering. For this the model is trained using the Naïve Bayes filter very well to work effectively. The Naive Bayes always calculates the probability of each class and the class having the maximum probability is then chosen as an output. Naïve Bayes always provide an accurate result. It is used in many fields like spam filtering.

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

$$P(B) = \sum_y P(B|A)P(A)$$

Rusland et al. [63] present research on email spam filtering and perform the analysis using a machine learning algorithm Naïve Bayes. They used two datasets evaluated on the value of accuracy, F-measure, precision, and recall. As we know, Naïve Bayes uses probability for classification, and the probability is counting the frequency and combination of values in a dataset. This research uses three steps for the filtration of emails, i.e., preprocessing, feature selection, and, at last, it implements the features by using the Naïve Bayes classifier. The preprocessing step removes all conjunction words, articles, and stop words from the email body. Then, they used the WEKA tool [64] and made two datasets called spam data and spam base dataset. The average accuracy was 89.59% using two datasets, while the spam data got 91.13% accuracy, The spam base dataset got an accuracy of 82.54%. The average precision results for

spam data were 83%, while, for spam base, the precision result was 88%. They claimed that the Naïve Bayes classifier performs better on spam base data as compared with spam data.



Random Forest:-

Random forest is a supervised ensemble classifier that is used for classification and regression. It is an ensemble learning method that creates a set of decision trees and combines them to make a final prediction. To arrive at a prediction, the random forest follows these steps:

Step 1: Select a random sample of data from the dataset.

Step 2: Build a decision tree using the sample data.

Step 3: Repeat the process a certain number of times.

Step 4: Combine the decision trees by taking the average of their predictions.

Each decision tree in the random forest makes a prediction, and the final prediction is made by taking the average of all the predictions made by the individual decision trees. This helps to reduce overfitting and improve the overall accuracy of the model. The two important hyperparameters that must be defined by the user when generating a random forest are `mtry` and `ntree`. In a random forest, `mtry` is the number of features that are randomly sampled as candidates for splitting at each decision tree node and the `ntree` is the number of decision trees in the random forest. The `mtry` parameter determines how much randomness is injected into the model. A smaller `mtry` value will make the model more deterministic and potentially more accurate but at the cost of a more complex model that is more prone to overfitting. A larger `mtry` value will make the model more robust to noise in the data, but at the cost of accuracy. The `ntree` parameter determines the overall complexity of the model. A larger value of `ntree` will make the model more accurate but at the cost of increased computational resources and longer training time. The number of trees in a forest (`ntree`) is not limited by computational resources, but the performance improvement from having a large number of trees is minimal, according to. However, states that computational resources are the limiting factor for the number of trees in a forest.

The Extra Trees Classifier (ETC) is an ensemble learning algorithm that belongs to the family of tree-based models, similar to Random Forest. Extra Trees, short for Extremely Randomized Trees, introduces additional randomness in the tree-building process compared to traditional decision trees and Random Forests. Here's an explanation of the Extra Trees Classifier:

1. **Randomization in Tree Building:**

Extra Trees introduces randomness not only in the selection of the feature at each node but also in the threshold for splitting. Instead of finding the optimal split point, Extra Trees selects split points randomly.

2. **Multiple Trees:**

Like Random Forest, Extra Trees builds multiple decision trees during training. Each tree is trained on a bootstrap sample (randomly sampled with replacement from the

original dataset).

3. **Voting or Averaging:**

For classification tasks, the final prediction is made by a majority vote among the trees. For regression tasks, the final prediction is often the average of the predictions from individual trees.

4. **Ensemble Learning:**

Extra Trees is an ensemble learning method, which means it combines the predictions of multiple weak learners (trees) to create a stronger and more robust model.

5. **Benefits:**

Extra Trees tends to have a lower variance compared to traditional decision trees, making it less prone to overfitting.

The extra randomization often leads to faster training times.

XGBOOST CLASSIFIER :-

XGBoost (eXtreme Gradient Boosting) is a powerful and efficient machine learning algorithm that belongs to the family of gradient boosting methods. Developed by Tianqi Chen, XGBoost has gained widespread popularity in various machine learning competitions and applications due to its speed, accuracy, and flexibility. Here's an explanation of the XGBoost classifier:

Key Features of XGBoost:

1. Gradient Boosting:

XGBoost is an implementation of gradient boosting, a machine learning technique where multiple weak learners (typically decision trees) are trained sequentially, and each new tree corrects the errors of the combined ensemble so far.

2. Regularization:

XGBoost incorporates regularization techniques to prevent overfitting. This includes

L1 (LASSO) and L2 (ridge) regularization terms added to the objective function, controlling the complexity of the model.

3. Parallel and Distributed Computing:

XGBoost is designed to be highly efficient, supporting parallel and distributed computing. It can leverage the computational power of multiple processors or even distributed computing frameworks like Apache Spark.

4. Handling Missing Values:

XGBoost has a robust mechanism for handling missing values. It can automatically learn the best imputation strategy during training.

5. Feature Importance:

XGBoost provides a feature importance score after training, helping users understand which features contribute the most to the model's predictions. This is useful for feature selection and interpretation.

6. Early Stopping:

XGBoost supports early stopping, allowing the training process to halt once the performance on a validation dataset stops improving, preventing overfitting.

7. Cross-Validation:

The algorithm supports built-in cross-validation to assess model performance more accurately and to help with hyperparameter tuning.

Training Process:

1. Initialization:

XGBoost starts with a simple model (usually a single leaf with the mean of the target

variable).

2. Building Trees:

Trees are added sequentially, with each new tree aiming to correct the errors made by the existing ensemble. Trees are pruned during the building process to control complexity.

3. Objective Function:

XGBoost minimizes an objective function that combines a loss term (related to prediction errors) and regularization terms. The optimization is performed using gradient descent methods.

4. Prediction:

The final prediction is made by combining the predictions of all trees in the ensemble. For regression tasks, it's the sum of the predictions; for classification, it's often the probability distribution or the class with the majority vote.

GRADIENT BOOST CLASSIFIER :-

❖ Gradient Boosting is a powerful ensemble learning technique that builds a series of weak learners (typically decision trees) sequentially, with each new learner correcting the errors of the combined ensemble so far. Gradient Boosting has become one of the most popular machine learning algorithms due to its high accuracy and flexibility. One of the well-known implementations of Gradient Boosting is the Gradient Boosting Classifier. Let's delve into the key concepts:

❖ Key Concepts of Gradient Boosting Classifier:

❖ **Weak Learners:**

Gradient Boosting builds an ensemble of weak learners, usually decision trees with a shallow depth. Each tree contributes to the overall prediction.

❖ **Sequential Training:**

Trees are added sequentially. Each new tree focuses on correcting the errors made by the existing ensemble. The learning process is done in a stepwise manner.

❖ **Loss Function:**

The algorithm minimizes a loss function, which measures the difference between the predicted values and the actual target values. Common loss functions for classification tasks include log loss (cross-entropy) and exponential loss.

❖ **Gradients and Residuals:**

During each iteration, the algorithm calculates the negative gradient of the loss function with respect to the predictions. The new tree is then trained to predict the negative gradient, which effectively models the residuals (errors) of the previous ensemble.

❖ **Shrinkage (Learning Rate):**

A shrinkage parameter (learning rate) is introduced to control the contribution of each new tree to the ensemble. A smaller learning rate requires more trees to achieve the same level of fitting.

❖ **Regularization:**

To prevent overfitting, regularization techniques like tree depth constraints and subsampling (randomly selecting a subset of data for each tree) are commonly

employed.

❖ **Prediction:**

The final prediction is the sum of the predictions from all the trees in the ensemble.

For binary classification, the output is often transformed using a sigmoid function to obtain probabilities.

ADABOOST CLASSIFIES :-

AdaBoost (Adaptive Boosting) is an ensemble learning algorithm that combines the predictions of multiple weak learners (typically simple decision trees) to create a strong classifier. AdaBoost is particularly useful for binary classification tasks, where the goal is to distinguish between two classes (positive and negative). The algorithm assigns weights to observations, with misclassified observations receiving higher weights in subsequent rounds.

Here's an explanation of the AdaBoost classifier:

Key Concepts of AdaBoost Classifier:

1. Weak Learners:

AdaBoost employs a series of weak learners, often decision trees with limited depth (stumps). These weak learners perform slightly better than random guessing.

2. Weighted Data:

Each data point in the training set is assigned an initial weight. Initially, all weights are set equally.

3. Sequential Training:

The algorithm trains a weak learner on the weighted dataset. After each round, the weights are adjusted based on the performance of the weak learner.

4. Weighted Error:

The error of the weak learner is computed, considering the weighted data. Misclassified observations receive higher weights, making them more influential in subsequent rounds.

5. Classifier Weight:

Each weak learner is assigned a weight based on its accuracy. A higher weight is given to more accurate classifiers.

6. Final Prediction:

The final prediction is a weighted sum of the predictions made by all weak learners. The weights of the weak learners depend on their accuracy.

7. Adaptiveness:

The algorithm adapts over multiple rounds by focusing on examples that were previously misclassified. This adaptiveness helps in handling complex decision boundaries.

BAGGING :-

“Bagging classifier is an ensemble classifier that fits base classifiers each on random sub sets of the original data sets and then combined their individual calculations by voting or by averaging) to form a final prediction. “Bagging is a mixture of bootstrapping and aggregating.

Bagging= Bootstrap AGGREGATING

Bootstrapping helps to lessening the variance of the classifier and it also decline the overfitting by just resampling the data from the training data with same cardinality as in original data set. Bagging is very effective method for limited data, and by using samples you are able to get estimate by aggregating the scores .

Multinomial Naive Bayes (MNB) classifier :-

The Multinomial Naive Bayes (MNB) classifier is a variant of the Naive Bayes algorithm specifically designed for text classification problems where the features represent the frequency of words in a document. It is commonly used for tasks such as spam detection, sentiment analysis, and topic classification. Here's an explanation of the Multinomial Naive Bayes classifier:

Key Concepts of Multinomial Naive Bayes:

1. Multinomial Distribution:

Multinomial Naive Bayes assumes that the features follow a multinomial

distribution, which is suitable for discrete data such as word counts in a document.

2. Bag-of-Words Representation:

Documents are represented as a "bag-of-words," where the order of words is disregarded, and only the frequency of each word matters.

3. Feature Vector:

Each document is represented as a feature vector, where each element corresponds to the count of a specific word in the document.

4. Class-Conditional Probabilities:

The classifier calculates the probability of each class given the observed feature vector. It does this by estimating class-conditional probabilities for each word in the vocabulary.

5. Feature Independence Assumption:

Like all Naive Bayes classifiers, MNB makes the "naive" assumption that features are conditionally independent given the class label. This simplifies the calculations but may not hold true in practice.

6. Laplace Smoothing:

To handle the issue of zero probabilities for unseen words in the test data, Laplace smoothing (additive smoothing) is often applied to the probability estimates.

DECISION TREE

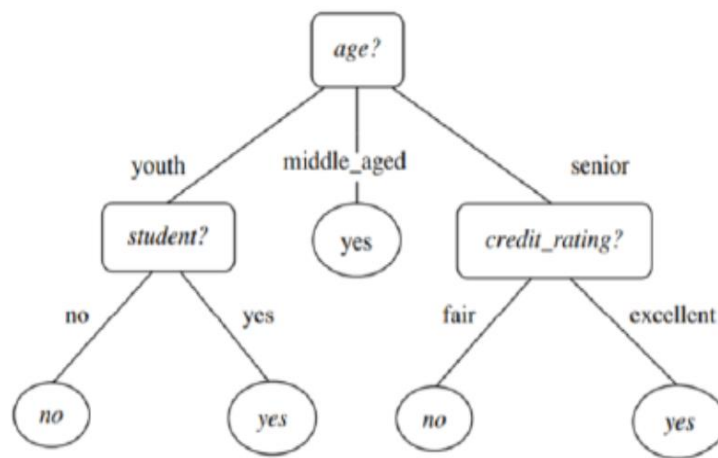
“Decision tree induction is the learning of decision tree from class labeled training tuples”. A decision tree is a flow chart like construction, where;

Internal node or non- leaf node= Test on attribute

Branch = shows outcome of the test

Leaf node= holds a class label

Top node is called root node.



Decision Tree Structure

Takhmiri and Haroonabadi [46] present a different technique to detect spams using a fuzzy decision tree and the Naïve Bayes algorithm. They use the baking voting algorithm to extract patterns of spam behaviour. They do this because obvious characteristics do not exist in the real world. The cross-linking degree for explaining or describing characters is rational and neutral. Decision trees use fuzzy Mamdani rules for the classification of spam and ham email.

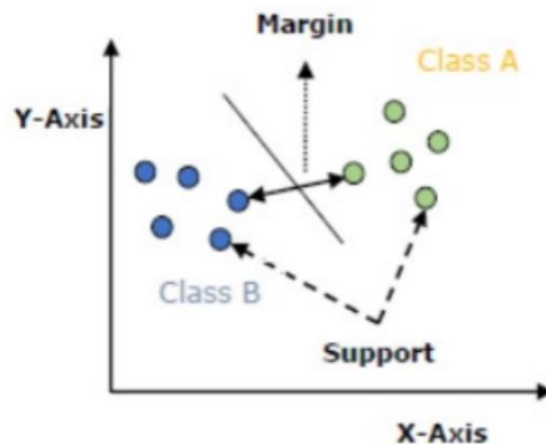
Then, Naive Bayes classifier [47] is used by them on the dataset. Finally, the baking method is used by dividing votes into smaller sections. This solution gives them an optimized weight that can be implemented on obtained percentage that achieve a higher accuracy level. The dataset used in this study contains 1000 emails, from which 350 (35%) were spam and 650 (65%) were ham.

SUPPORT VECTOR MACHINE

“The Support Vector Machine (SVM) is a popular Supervised Learning algorithm, the Support Vector model is used for classification problems in Machine Learning techniques.”

The Support Vector Machines totally founded on the idea of Decision points. The Main resolution of Support Vector Machine algorithm is to create the line or decision boundary. The Support Vector Machine algorithm gives hyperplane as a output which classifies new samples. In 2-dimensional space “hyperplane is line dividing a plane into 2 parts where each class is present in one side.”

Banday and Jan [55] present research in which they define the procedure of statistical spam filters. They design those filters using Naïve Bayes, KNN, support vector machines (SVM), and regression trees [56]. They use all these supervised machine learning algorithms and evaluate the results based on precision, recall, and accuracy. Using these machine learning techniques, they found that classification and regression trees (CART) [57] and Naïve Bayes classifiers are the most effective algorithms for the dataset is approach estimates that, during spam filtering, calculations of false positive are costlier than a false negative



Support Vector Machine

Modules and Explanation

❖ The Application consists of three modules.

- UI
- Machine Learning
- Data Processing

❖ UI Module

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

❖ Machine Learning Module

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

❖ Data Processing Module

- The raw data undergoes several modifications in this module for further process.
- Some of the main functions of this module includes
 - Data cleaning
 - Data merging of datasets
 - Text Processing using NLP
 - Conversion of text data into numerical data(feature vectors).
 - Splitting of data.
- All the data processing is done using Pandas and NumPy libraries.

Requirements

Hardware Requirements

PC/Laptop Ram – 8 Gig

Storage – 100-200 Mb

17

Software Requirements

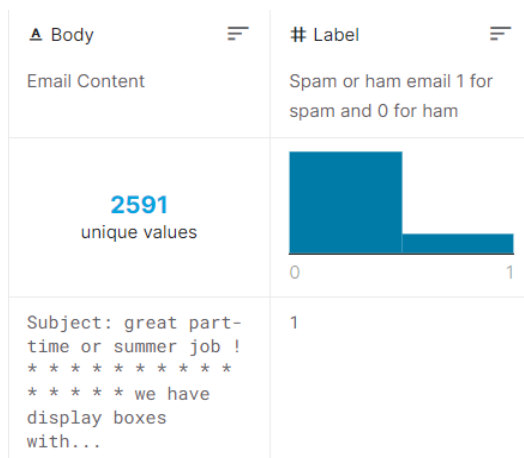
OS – Windows 7 and above

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..

Data Description

Dataset : enronSpamSubset.



Source : Kaggle

Description : this dataset is part of a larger dataset called enron. This dataset contains a set of spam and non-spam emails with 0 for non spam and 1 for spam in label attribute.

Composition :

Unique values : 9687

Spam values : 5000

Non-spam values : 4687

fig no. 4.3 enron spam

Dataset : lingspam.

Source : Kaggle


Description : This dataset is part of a larger dataset called Enron1 which contains emails classified as spam or ham(not-spam).

Composition :

Unique values : 2591

Spam values : 419

Non-spam values : 2172

Body	Label
Email Content	Spam or ham email 1 for spam and 0 for ham
9687 unique values	
Subject: stock promo mover : cwttd * * * urgent investor trading alert * * * weekly stock pick - - ...	1

Data Processing

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).

19

Data cleaning, Data Merging these procedures are completely done using Pandas library.

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 post-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 count vectorization 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 ~~repe~~ance of words to the collection of documents and 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

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

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.

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

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 high recurrence and increases the importance of words/terms that are rare.

$$idf(t) = N/df$$

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

$$tf_idf(t, d) = tf(t, d) * \log(N/(df + 1))$$

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]

Then finding the term frequency

table no. 4.1 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	$\log(3/1)$

Then finding the inverse document frequency

table no. 4.2 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.

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 testing data and remaining data is testing data.

Machine Learning

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.

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 nevercorrect in real-world data, but often works well in practice. so that it is called “Naive” .

$$P(A|B)=(P(B|A)P(A))/P(B)$$

$P(A|B)$ is the probability of hypothesis A given the data B. This is called the posteriorprobability.

$P(B|A)$ 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.

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

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

here z = odds

generally, odds are calculated as

$$\text{odds} = \frac{p(\text{event occurring})}{p(\text{event not occurring})}$$

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

$$\log(\text{odds}) = \text{dot}(\text{features}, \text{coefficients}) + \text{intercept}$$

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

CONCLUSION

In conclusion, machine learning and natural language processing (NLP) techniques can be effectively used for email spam classification. By leveraging the power of supervised learning algorithms such as Naive Bayes, Support Vector Machines, and KNN, and by preprocessing the text data using techniques such as tokenization, stop-word removal, and stemming, it is possible to build accurate and reliable spam filters that can automatically detect and filter out unwanted emails. These techniques can also be extended to handle more complex spamming strategies such as phishing attacks and spear phishing. Overall, in the proposed models Naïve Bayes having the accuracy of 99% SVM having 98% and KNN having 97%. Finally naïve bayes having the highest accuracy so we predict the Naïve bayes model. The use of ML and NLP for email spam classification can save users valuable time and resources and improve the overall productivity and security of email communication.

There is a wide possibility of improvement in our project. The subsequent improvements can be done:

“Filtering of spams can be done on the basis of the trusted and verified domain names.”

“The spam email classification is very significant in categorizing e-mails and to distinct e-mails that are spam or non-spam.”

“This method can be used by the big body to differentiate decent mails that are only the emails they wish to obtain.”

References

- Dixon, S. Global Average Daily Spam Volume 2021. Available online: <https://www.statista.com/statistics/1270424/daily-spam-volume-global/> (accessed on 18 July 2022).
- FBI. Federal Bureau of Investigation: Internet Crime Report 2021. Available online: https://www.ic3.gov/Media/PDF/AnnualReport/2021_IC3Report.pdf (accessed on 6 August 2022).
- Securelist Types of Text-Based Fraud. Available online: <https://securelist.com/mail-text-scam/106926/> (accessed on 4 August 2022).
- Onova, C.U.; Omotehinwa, T.O. Development of a Machine Learning Model for Image-Based Email Spam Detection. *FUOYE J. Eng. Technol.* **2021**, *6*, 336–340. [[Google Scholar](#)] [[CrossRef](#)]
- Bindu, V.; Thomas, C. Knowledge Base Representation of Emails Using Ontology for Spam Filtering. *Adv. Intell. Syst. Comput.* **2021**, *1133*, 723–735. [[Google Scholar](#)] [[CrossRef](#)]
- Kaddoura, S.; Chandrasekaran, G.; Popescu, D.E.; Duraisamy, J.H. A Systematic Literature Review on Spam Content Detection and Classification. *PeerJ Comput. Sci.* **2022**, *8*, e830. [[Google Scholar](#)] [[CrossRef](#)] [[PubMed](#)]
- Méndez, J.R.; Cotos-Yañez, T.R.; Ruano-Ordás, D. A New Semantic-Based Feature Selection Method for Spam Filtering. *Appl. Soft Comput.* **2019**, *76*, 89–104. [[Google Scholar](#)] [[CrossRef](#)]
- Ahmed, N.; Amin, R.; Aldabbas, H.; Koundal, D.; Alouffi, B.; Shah, T. Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges. *Secur. Commun. Networks* **2022**, *2022*, 1862888. [[Google Scholar](#)] [[CrossRef](#)]
- Hosseinalipour, A.; Ghanbarzadeh, R. A Novel Approach for Spam Detection Using Horse Herd Optimization Algorithm. *Neural Comput. Appl.* **2022**, *34*, 13091–13105. [[Google Scholar](#)] [[CrossRef](#)]
- Ismail, S.S.I.; Mansour, R.F.; Abd El-Aziz, R.M.; Taloba, A.I. Efficient E-Mail Spam Detection Strategy Using Genetic Decision Tree Processing with NLP Features. *Comput. Intell. Neurosci.* **2022**, *2022*, 7710005. [[Google Scholar](#)] [[CrossRef](#)]
- Ravi Kumar, G.; Murthuja, P.; Anjan Babu, G.; Nagamani, K. An Efficient Email Spam Detection Utilizing Machine Learning Approaches. *Proc. Lect. Notes Data Eng. Commun. Technol.* **2022**, *96*, 141–151. [[Google Scholar](#)]
- Kontsewaya, Y.; Antonov, E.; Artamonov, A. Evaluating the Effectiveness of Machine Learning Methods for Spam Detection. *Procedia Comput. Sci.* **2021**, *190*, 479–486. [[Google Scholar](#)] [[CrossRef](#)]
- Batra, J.; Jain, R.; Tikkiwal, V.A.; Chakraborty, A. A Comprehensive Study of Spam Detection in E-Mails Using Bio-Inspired Optimization Techniques. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100006. [[Google Scholar](#)] [[CrossRef](#)]
- Dedetürk, B.K.; Akay, B. Spam Filtering Using a Logistic Regression Model Trained by an Artificial Bee Colony Algorithm. *Appl. Soft Comput. J.* **2020**, *91*, 106229. [[Google Scholar](#)] [[CrossRef](#)]
- Sagi, O.; Rokach, L. Ensemble Learning: A Survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1249. [[Google Scholar](#)] [[CrossRef](#)]
- Sheu, J.J.; Chu, K.T.; Li, N.F.; Lee, C.C. An Efficient Incremental Learning Mechanism for Tracking Concept Drift in Spam Filtering. *PLoS ONE* **2017**, *12*, e0171518. [[Google Scholar](#)] [[CrossRef](#)]
- Liu, X.; Zou, P.; Zhang, W.; Zhou, J.; Dai, C.; Wang, F.; Zhang, X. CPSFS: A Credible Personalized Spam Filtering Scheme by Crowdsourcing. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 1457870. [[Google Scholar](#)] [[CrossRef](#)]
- Bahgat, E.M.; Rady, S.; Gad, W.; Moawad, I.F. Efficient Email Classification Approach Based on Semantic Methods. *Ain Shams Eng. J.* **2018**, *9*, 3259–3269. [[Google Scholar](#)] [[CrossRef](#)]
- Agarwal, K.; Kumar, T. Email Spam Detection Using Integrated Approach of Naïve Bayes and Particle Swarm Optimization. In Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018, Madurai, India, 14–15 June 2018; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2019; pp. 685–690. [[Google Scholar](#)]
- Dada, E.G.; Bassi, J.S.; Chiroma, H.; Abdulhamid, S.M.; Adetunmbi, A.O.; Ajibuwa, O.E. Machine Learning for Email Spam Filtering: Review, Approaches and Open Research Problems. *Heliyon* **2019**, *5*, e01802. [[Google Scholar](#)] [[CrossRef](#)] [[PubMed](#)].

