International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 13 Issue 4, April-2024, Impact Factor: 8.375

Email Spam Detection using Machine Learning

Anisha Rai[1], Sakshi[2], Gyanvi[3], Pooja[4], M. Ejaz A. Lodhi[5]

[1,2,3,4,5] Department of Electronics and Communication Engineering, Indira Gandhi Delhi Technical University For Women, Delhi, India

***Corresponding Author: ejaz.iitk@gmail.com**

ABSTRACT

Email spam discovery refers to the process of labeling and filtering out spontaneous, undesirable, or hateful emails from arriving a user's inbox. The aim of Searching out or separate authentic emails from spam, guaranteeing that consumers accept only the emails they want or need. Mail spam discovery strategies confront special challenges due to the different nature of mail substance, counting connections, designing, and changing dialect styles. Tragically, existing investigate does not comprehensively address these particular challenges. This highlights a critical opportunity for future inquire about in this space, and this overview can serve as a profitable reference for directing the heading of up and coming investigate endeavors in e-mail spam location.

Keywords : Naïve Bayes, SVM, Logistic Regression, Gaussian NB, Decision Tree, KN eighbors.

INTRODUCTION

E-mail is one of the foremost commonly and broadly utilized communication mediums. The term 'email' alludes to both the client action and all shapes of electronic mail informing in numerous parts of the world. It has advanced into a essential channel for communication as well as the advancement of items, conveyance of keeping money upgrades, spread of rural data, arrangement of flight upgrades, and offers from internet administrations. Mail is additionally broadly utilized in coordinate showcasing, known as e-mail promoting. In any case, at times, e-mail promoting gets to be a source of irritation for clients. These sorts of emails are commonly alluded to as spam emails. Spam comprises one or more undesirable messages that clients did not ask, sent or disseminated as portion of a bigger set of messages, all sharing considerably comparative substance. The targets of e-mail spam incorporate promoting different items, sending political messages, disseminating inappropriate grown-up substance, and advancing web administrations.

Spam or Non-Spam:

The utilization of electronic informing frameworks to send spontaneous bulk messages, particularly mass notices, noxious joins, etc., is called spam." "Spontaneous" implies accepting messages from sources without earlier ask. Subsequently, in the event that you're new with the sender, the e-mail may well be considered spam. Individuals regularly do not realize they've marked up for these mailers when they download free administrations, program, or upgrade existing computer program. The term "Ham" was coined by Spam Bayes around 2001 and is characterized as "Emails that are not by and large craved and are not considered spam."

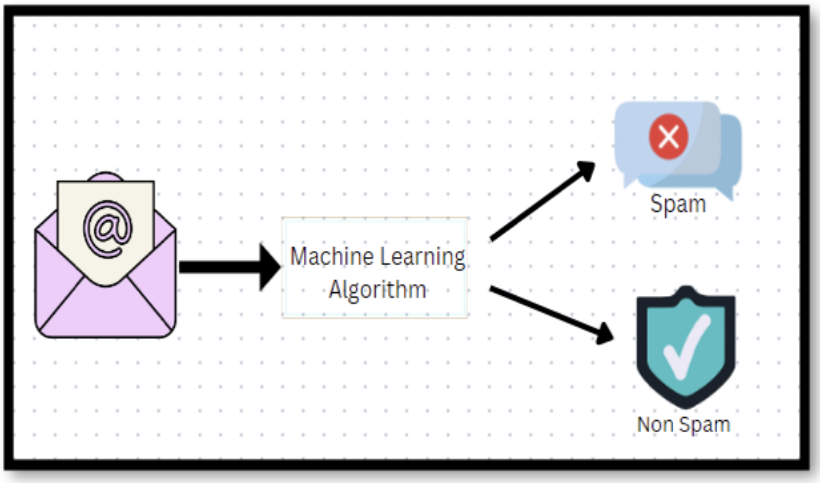

Figure 1: Classification into Spam and Ham Mails

ALGORITHM

Naive Bayes

The Naive Bayes theorem is like a super useful tool in machine learning. It helps us guess things based on what we know already and what we see. Imagine using it to figure out if an email is normal or spam, or even to understand if a message is positive or negative! It's kind of like having a smart helper that uses clues to make smart guesses. This theorem is really handy in lots of situations. This theorem finds extensive application across various domains, from sentiment analysis to email filtering. Its strength lies in its ability to efficiently analyze and categorize data, making it an indispensable tool in classification tasks. Naive Bayes' simplicity, combined with its high effectiveness, renders it an invaluable asset in the machine learning toolkit, enabling informed decision-making in a wide array of real-world scenarios.

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

1. *The chance that event A will occur given that event B has already occurred is expressed as P(A/B). In classification, this represents the probability of a certain class given the features.*
2. *The likelihood that event B will occur given that event A has already happened is expressed as P(B/A). In classification, this represents the likelihood of observing the features given the class.*
3. *P(A) represents the event A previous occurrence probability. In classification, this represents the probability of a certain class occurring without considering any features.*
4. *P(B) represents the event B previous occurrence probability. In classification, this is the probability of observing the features.*

Data Preprocessing

Before we can apply any machine learning algorithm, it's crucial to preprocess the data. This entails doing things like encoding category data, normalizing numerical characteristics, and handling missing values. Through these steps, we ensure that our dataset is in a suitable format for training our Naive Bayes model.

Bag of Words

One of the fundamental methods of natural language processing is the Bag of Words (BoW) representation. It converts text data into a numerical representation by compiling a list of distinct terms and tallying how many times they appear in every document. This enables us to apply machine learning algorithms to text data for tasks like sentiment analysis and text categorization.

We can turn a set of documents into a matrix by using a procedure we'll go over today. Each document will be represented as a row, each word (or token) as a column, and the accompanying (row, column) values will represent how often each word or token appears in that document.

Figure 2: Bag of Words

BoW from scratch

To implement Bag of Words from scratch, we begin by constructing a vocabulary from the entire corpus. Next, we tokenize each document, converting it into a vector of word counts. This process involves careful handling of text preprocessing steps like punctuation removal and stemming. Ultimately, we obtain a BoW representation for each document. Scikit-learn provides a powerful and efficient way to implement Bag of Words. By utilizing the Count Vectorizer module, we can effortlessly convert text data into BoW vectors. This approach streamlines the process and allows us to focus on the subsequent steps of our analysis.

	are	call	from	hello	home	how	me	money	now	tomorrow	win	you
0	1	0	0	1	0	1	0	0	0	0	0	1
1	0	0	1	0	1	0	0	1	0	0	2	0
2	0	1	0	0	0	0	1	0	1	0	0	0
3	0	1	0	2	0	0	0	0	0	1	0	1

Figure 3: implementation of a Bag of Wordsproblem for a document dataset

Trainingand testing sets

The training and testing sets of our dataset must be separated in order to assess the effectiveness of our Naive Bayes model. The model is trained on the training set, and its generalization abilities are evaluated on the testing set. This step is essential in preventing overfitting and ensuring our model's reliability. It provides a way to measure how accurately the model performs on completely new data. This is essential for determining how well the model can apply what it has learned to real-world situations. Without this division, there's a risk that the model might merely memorize the training data, which can lead to difficulties in handling unfamiliar Data - a problem known as overfitting.

GaussianNB Algorithm

In the process of using GaussianNB for email spam detection, we start with a collection of emails labeled as either spam or non-spam. These emails undergo a cleaning process where we standardize their format, remove unnecessary elements like punctuation and common words. Subsequently, we transform the text into a format that a computer can

understand, essentially converting words into numerical values. This enables us to teach the computer using these examples, allowing it to discern patterns indicative of spam. After training, we put it to the test with new, unseen emails to evaluate its accuracy. If it performs well, we can deploy it to automatically sift through incoming emails, flagging or sorting potential spam, thus enhancing our email security and efficiency. Keep in mind, that while GaussianNB is a good starting point, more sophisticated techniques can be explored for even better results.

SVC (Support Vector Classifier)

The Support Vector Classifier (SVC) is a classification algorithm based on the Support Vector Machine (SVM) framework. It seeks to identify the ideal hyperplane in a high-dimensional feature space for dividing various data point classes. Unlike traditional SVMs, SVC is specifically designed for binary classification tasks. It identifies the support vectors, which are the data points closest to the decision boundary, and utilizes them to determine the position of the hyperplane. By finding this optimal separation, SVC excels in scenarios where the data is not linearly separable, and it can also incorporate kernel functions to handle non-linear relationships in the data. This makes SVC a versatile and powerful tool for tasks like email spam detection, sentiment analysis, and various other classification tasks. It is known for its effectiveness, even in cases where the data is complex and not easily separable.

k-Nearest Neighbors (k-NN)

A straightforward but efficient machine learning technique for classification and regression problems is the k-Nearest Neighbors (k-NN) method. A data point's label in classification is determined by comparing it to the labels of its closest neighbors in the feature space. The number of neighbors to take into account is indicated by the "k" in k-NN. The method determines the label of a new point by examining the five nearest data points, for instance, if k is set to 5. The distance metric (commonly Euclidean distance) is used to measure proximity. k-NN is considered a lazy learner because it doesn't build a model during training; instead, it stores the entire dataset and computes predictions at runtime. This makes it relatively simple to implement, but it can be computationally expensive for large datasets.

Decision Tree

A strong machine-learning approach for both classification and regression applications is the decision tree. It builds a structure resembling a tree, with each node standing for a feature and each branch for a choice based on that feature. The algorithm partitions the data iteratively until it reaches leaf nodes, which represent the ultimate anticipated results.

Logistic Regression

Logistic Regression is a key machine learning method for tasks like deciding between two categories. Despite its name, it's mainly for classifying, not doing math like traditional regression. It predicts the chance of something belonging to a specific group. The result is adjusted using the logistic function, keeping it between 0 and 1This method excels at decisions like spam or non-spam emails. Its simplicity and interpretability make it a popular choice in various fields. However, it assumes a linear relationship between features and outcomes, which may not always hold true.

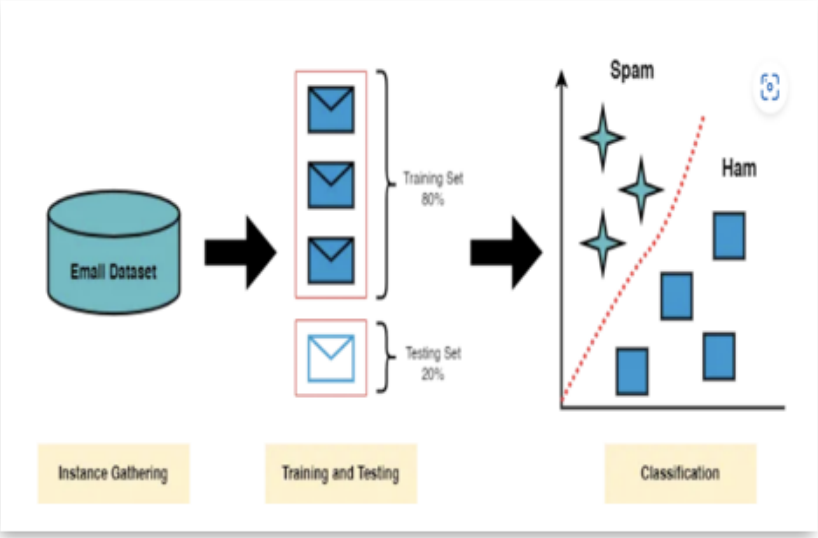

Figure 4: Ham/Spam Email Detection

EVALUATING OUR MODEL

After the predictions are made on our test set, next objective is to assess the performance of our model. There are several ways to accomplish this, some of which include, The classifier's **accuracy** is measured by how frequently it makes the right prediction. It is calculated as the ratio of the total number of forecasts (i.e., test data points) to the number of correct predictions. The **precision** reveals the percentage of messages that we thought were spam that were in fact spam. It's a ratio of "true positives," or words that are correctly categorized as spam, to "all positives," or all terms classified as spam, regardless of whether that was the right classification, in other words, it is the ratio of

[True Positives/(True Positives + False Positives)]

Recall (sensitivity) indicates the percentage of messages that we genuinely considered to be spam that were actually classified as such. It is a ratio of all the words that were truly spam to all the true positives, or words that are actually classed as spam, in other words, it is the ratio of

[True Positives/(True Positives + False Negatives)]

The harmonic mean of recall and precision is known as the **F1 Score**. It offers an impartial assessment of a model's efficacy that accounts for both false positives and false negatives.

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

DATASET DESCRIPTION

The dataset is imported from Kaggle containing both email spam and legitimate (non-spam) emails. I carefully prepared this dataset to suit various types of machine learning algorithms designed for email classification. The performance of these algorithms relies heavily on the quality and composition of the dataset. It's important to note that spam detection algorithms require a well-curated dataset for accurate results. To ensure the robustness of our experiments, we utilized different publicly available datasets commonly used in various studies. The dataset includes information such as the total number of emails, as well as the counts of spam and legitimate emails.

Result

Algorithm	Accuracy	Precision	Recall	F1 Score
Naive Bayes	93.7%	89.8%	89.8%	89.8%
SVM	95.6%	90.4%	97.0%	93.2%
Logistic Regression	94.3%	89.8%	92.2%	91.0%
Gaussian NB	90.9%	84.0%	87.4%	85.7%
Decision Tree	88.6%	80.3%	87.4%	85.7%
K-NN	39.7%	33.9%	1.2%	50.6%

Figure 5: Comparison Table

Figure 6: Comparison Plot

We compared our result with the result of another paper, whose comparison table is shown below:

Algorithm	Accuracy 1	Accuracy 2
Naïve Bayes	**93.7%**	**79.5%**
Logistic Regression	**94.3%**	**93.1%**
SVM	**95.6%**	**90.7%**

- Accuracy 1 – Our paper's result
- Accuracy 2 – Another paper's result (*Performance Evaluation of Machine Learning Algorithms for Email Spam Detection*) (Nandhini.S, 2020)
- Naïve Bayes Algorithm - In Naïve Bayes Algorithm, our Accuracy is Lead by 14.2% (93.7% - 79.5%).
- Logistic Regression Algorithm - In the Logistic Regression Algorithm, our Accuracy is Lead by 1.2% (94.3% - 93.1%).
- SVM Algorithm – In SVM Algorithm, our Accuracy is Lead by 4.9% (95.6% - 90.7%).
- Hence, it is observed that our algorithms work more accurately in compared to the other referenced paper and data.

CONCLUSION

In this study, we embarked on the task of email spam detection employing a diverse set of machine learning algorithms including Naive Bayes, Support Vector Classifier (SVC), GaussianNB, Logistic Regression, KNeighbors, and Decision Trees. Each algorithm was rigorously evaluated for accuracy, precision, recall, and F1 score, vital metrics in email filtering systems. Our findings revealed that Support Vector Classifier (SVC) emerged as the most adept algorithm,

exhibiting an impressive accuracy of 95.69%, a precision of 90.45%, a recall of 97.05%, and F1score of 93.26%. This signifies its robustness in accurately identifying spam emails, making it a top contender for practical implementation. Following closely, the Logistic Regression demonstrated commendable performance, further underlining its potential in email spam detection.

While the Naïve Bayes algorithm displayed respectable results, it lagged slightly behind SVC and logistic regression regarding accuracy, precision, recall, and F1score. Decision Trees, and KNeighbors although effective in various contexts, proved to be comparatively less suitable for this specific task, showcasing the importance of selecting the right algorithm for the given problem domain.

REFERENCES

[1]. (Nagre, 2018)Mobile SMS Spam Detection using Machine Learning Techniques 2018 JETIR December 2018, Volume 5, Issue 12.
[2]. (Existing spam filtering methods considering different techniques, 2021) International conference of technology advancement and innovation.
[3]. (Chae, 2017)Artificial intelligence based method for filtering spam message in email service, 2nd international conference on Anti cybercrimes.
[4]. S. Nandhini and J. Marseline K.S., "Performance Evaluation of Machine Learning Algorithms for Email Spam Detection," 2020 International Conference on Emerging Trends in Information Technology and Engineering, Vellore, India, 2020.
[5]. Dataset https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset