WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

E-Mail Classification Using Machine Learning

[1]Prajapati Ravi Santram, [2]Aarti Chavan, [3]Khan Adil Parvez

[1]Student, Department of Information Technology, Priti Degree College, Kalyan, Mumbai, India
[2]Asst. Professor, Department of Information Technology, Priti Degree College, Kalyan, Mumbai, India
[3]Lecturer, Department of Robotics & Automation, Abdul Razzak Kalsekar Polytechnic, Mumbai, India

ABSTRACT:

Nowadays, a large percentage of people rely on available emails or messages sent by strangers. The ability for anyone to leave an email or message gives spammers a golden opportunity to spam about our various interests, filling the inbox with a variety of ridiculous emails. They greatly affect our internet speed and steal useful information, such as our contact list details. Identifying these spammers and also the spam content can be a hot topic of research and tedious tasks. Email spam is a process of sending messages en masse through the mail. Since the cost of the spam is mostly borne by the recipient, it is effectively postage-free advertising. Spam email is a type of commercial advertising that is economically viable because email can be a very cost-effective medium for the sender. With this proposed model, the specified message can be detected as spam or not using Bayes theorem and Naive Bayes classifier and also IP addresses of the sender are often detected. In this paper we will discuss and apply these algorithms on our dataset and the algorithm which gives the best accuracy and precision will be considered.

Keywords: Machine Learning Algorithms, Naïve Bayes, Support vector machines (SVM), Term Frequency, Inverse Document Frequency.

I. INTRODUCTION

In a more digitizing world, building and constructing models that are designed to separate data into categories has become a relevant design space for different models and algorithms. The goal of this project is to construct a machine learning algorithm that separates emails into two different categories. Furthermore an algorithm will be constructed that will improves over time and become better at categorising emails. This will be done by first constructing a dataset reflecting real world data, since it will be nearly impossible to collect enough data from a normal email this will be solved by collecting the data from two different online forums. The data collected will be used to train and tune the different machine learning algorithms. The 4 different methods were: k-nearest neighbors, adaptive boosting, random forest and artificial neural network. The best preforming algorithm will be incorporated in the final product. The final product will be a feedback loop machine learning algorithm. Where new data will be presented to the algorithm and tested. If the algorithm is sure that a new data point is the right class it would incorporate it into the dataset and over time the dataset would grow larger and the algorithm would adapt to new data and trends. The goals of the project is to understand what makes a good machine learning algorithm for email classification and to understand how it can be incorporated into something that grows over time. The project main limitation is time since only 4 different methods will be tested and the growing algorithm will not be able to run for a substantial time.

II. LITERATURE SURVEY

Consider a situation in which by the means of the internet, you are receiving spam emails very frequently whether it may be promotions for their products to purchase them. These are negative marketing activities and fraud activities. As a receiver, we are helpless to control these spam emails. It also consumes a lot of memory and valuable time. So, there is a high need of having some mechanism to reduce these types of spam emails. In this paper, we are presenting a machine learning-based spam detection mechanism. This model consumes a dataset containing approximately 6000 emails. Using this mechanism, so much time and memory will be saved. With the help of this dataset, features can be extracted and it plays a major role in identifying the accuracy, precision, computational power, and misclassification rate of the particular algorithm.

In Paper [1], Email Spam Detection using integrated approach of Naïve Bayes and Particle Swarm Optimization[1] Naïve Bayes algorithm is a Bayes theorem based statistical machine learning based approach having properties of strong independence, probability distribution and ability to handle large datasets. In NB, probability distribution is evaluated from the frequency distribution of dataset. Particle Swarm Optimization (PSO) is swarm intelligence based concept derived in 1995 by Eberhart and Kennedy PSO work on the property of stochastic distribution and initially find the local search solution, then individual particle share their solution and global solution is obtained. NB having probability distribution property determines the possible class for the email content from the spam class or non-spam class on the basis of keywords present in the email textual data. PSO is used to further optimize the parameters of NB approach to improve the accuracy, search space and classification process.

In paper [2], In this paper, they used a Logistic regression model to classify Spam mail. It is a machine-learning classification algorithm. We use this model for the prediction of the output of a categorical dependent variable. This logistic model can estimate the probability of two class responses like ham/spam. A spam mail dataset is taken which contains 5572 emails including both ham and spam emails. The dataset is divided into two sets, one for training and another for testing. They have taken 80 percent of emails for Training and 20 percent for testing and the output of the model will be shown with the help of 0 and 1. By using this logistic regression algorithm to achieve an accuracy of 96.59 percent.

In paper [3], A dataset is taken from spam assassin which contains nearly 5000 emails that will be read by a python package "Pyzmail". during this text pre-processing phase, email structures are going to be extracted and the contents will be converted to plain text for analysis. during this model, two feature sets are prepared i.e., stopwords with N-gram and tf-IDF (term frequency-inverse document frequency) and therefore the most frequent word count with count vectorization. N-gram and tf-IDF are created by exploring the text structure to take advantage of the contextual features. the foremost frequent word count with count vectorization is based on counting the most frequently occurring words from email content. A pipeline is made to feed data with the feature set and it is also easier to compare results. This pipeline consists of three algorithms i.e., Naïve Bayes, Logistic Regression, and Support vector machines. The evaluation criteria are supported by Accuracy, Precision, Recall, and F1 Score. Among the models which are using feature set 1 i.e., N-gram and tf-IDF, Logistic Regression got the very best precision i.e.,98.33%. Among the models which are using feature set 2 i.e., the most frequent word count with count vectorization, Logistic Regression got the very best precision i.e., 99.33%

In paper [4], Intelligent Model for Classification of SPAM and HAM. In this paper they have used machine learning and non machine learning approaches. Machine learning approaches like support vector mechanism, neural network etc. Non machine learning approaches like strong key word searching and whitelisting and blacklisting of words. The sets so formed are further used as training set and the classification set. The process is to use the first set as training set and the remaining N-1 sets as the sets to be classified. In the next iteration the second set is used as the training set and the remaining sets are sets to be classified. The process is repeated until all the sets are used as training sets. The emails are classified based on the spam percentage each mail gains.

III. METHODOLOGY

In 1998 the Naïve Bayes classifier was proposed for spam recognition. Bayesian classifier is working on the dependent events and the probability of an event occurring in the future that can be detected from the previous occurring of the same event . This technique can be used to classify spam e-mails; words probabilities play the main rule here. If some words occur often in spam but not in ham, then this incoming e-mail is probably spam. Naïve bayes classifier technique has become a very popular method in mail filtering software. Bayesian filter should be trained to work effectively. Every word has certain probability of occurring in spam or ham email in its database. If the total of words probabilities exceeds a certain limit, the filter will mark the e-mail to either category. Here, only two categories are necessary: spam or ham. Almost all the statistic-based spam filters use Bayesian probability calculation to combine individual token's statistics to an overall score, and make filtering decision based on the score. The statistic we are mostly interested for a token T is its spamminess (spam rating) , calculated as follows: Most of the spam filtering techniques is based on text categorization methods. Thus filtering spam turns on a classification problem. In our work, rules are framed to extract feature vector from email. As the characteristics of discrimination are not well defined, it is more convenient to apply machine learning techniques. Three machine learning algorithms, C 4.5 Decision tree classifier, Multilayer There are

number of rules framed by considering the various features that will aid to identify the spam messages effectively. Each rule performs a test on the email, and each rule has a score. When an email is processed, it is tested against each rule. For each rule found to be true for an email, the score associated with the rule is added to the overall score for that email. Once all the rules have been used, the total score for the email is compared to a threshold value. If the score exceeds the threshold, then the email is marked as spam and the others are classified as legitimate mail. In this work, the rules used are.

$$S[T] = \frac{C_{Spam}(T)}{C_{Spam}(T) + C_{Ham}(T)}$$

Where CSpam(T) and CHam(T) are the number of spam or ham messages containing token T, respectively. To calculate the possibility for a message M with tokens {T1,,TN}, one needs to combine the individual token's spamminess to evaluate the overall message spamminess. A simple way to make

classifications is to calculate the product of individual token's spamminess and compare it with the product of individual token's hamminess

$$(H[M] = \prod_{I=1}^{N} (1 - S[T_I]))$$

Pre processing the data

Remove punctuations. Remove stop words :- Stop words like "and", "the", "of", etc are very common in all English sentences and are not very meaningful in deciding spam or legitimate status, so these words have been removed from the emails.

In pre processing we remove all the unwanted words and punctuations such as comma, full stops, extra spaces and other repeated words. This will help to reduce the data to be processed. This helps to process the data much faster than before due to small amount of data.

```
def process_text(text):
    '''
    What will be covered:
    1. Remove punctuation
    2. Remove stopwords
    3. Return list of clean text words
    '''

    #1
    nopunc = [char for char in text if char not in string.punctuation]
    nopunc = ''.join(nopunc)

    #2
    clean_words = [word for word in nopunc.split() if word.lower() not in stopwords.words('english')]

    #3
    return clean_words
```

Creating Model and Training

- Naïve Bayes Classifier

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
messages_bow = CountVectorizer(analyzer=process_text).fit_transform(df['v2'])
X_train, X_test, y_train, y_test = train_test_split(messages_bow, df['v1'], test_size = 0.20, random_state = 0)
classifier = MultinomialNB()
classifier.fit(X_train, y_train)
```

Fig 5.3.1 Naive Bayes Model

Here we split the data into train and test data and then we convert our data into the desired matrix format. To do this we will be using Count Vectorizer(). There are two steps to consider here: Firstly, we have to fit our training data (X_train) into Count Vectorizer() and return the matrix. Secondly, we have to transform our testing data (X_test) to return the matrix. Note that X_train is our training data for the 'v2' column in our dataset and we will be using this to train our model. X_test is our testing data for the 'v2' column and this is the data we will be using(after transformation to a matrix) to make predictions. Import the Multinomial classifier and fit the training data into the classifier using fit(). Name your classifier 'classifier'. Now that our algorithm has been trained using the training data set we can now make some predictions on the test data stored in 'X_test' using predict().

- Logistic Regression model

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
messages_bow = CountVectorizer(analyzer=process_text).fit_transform(df['v2'])
message_train, message_test, spam_nospam_train, spam_nospam_test = train_test_split(messages_bow, df['v1'], test_size = 0.20,
Spam_model = LogisticRegression(solver='liblinear', penalty='l1')
Spam_model.fit(message_train, spam_nospam_train)
pred = Spam_model.predict(message_test)
```

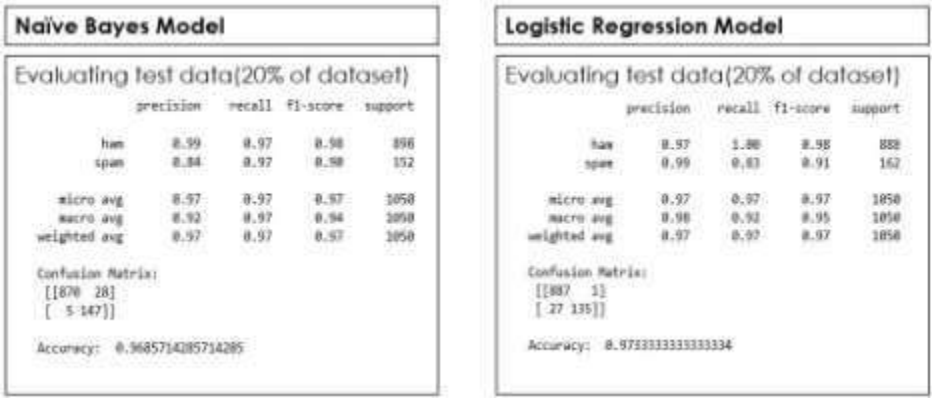
Fig 5.3.2 Logistic regression Model

Here we split the data into train and test data and then we convert our data into the desired matrix format. To do this we will be using Count Vectorizer(). There are two steps to consider here: Firstly, we have to fit our training data (message train) into Count Vectorizer() and return the matrix. Secondly, we have to transform our testing data (message test) to return the matrix. Note that message train is our training data for the 'v2' column in our dataset and we will be using this to train our model. message test is our testing data for the 'v2' column and this is the data we will be using(after transformation to a matrix) to make predictions.

Import the Logistic Regression algorithm and fit the training data into the model using fit(). Name your model 'spam model'. Now that our algorithm has been trained using the training data set we can now make some predictions on the test data stored in 'message test' using predict().

IV. RESULTS

Comparing Results for Naïve Bayes and logistic Regression model with 80-20 split ratio.



6.1. SCREENSHOTS

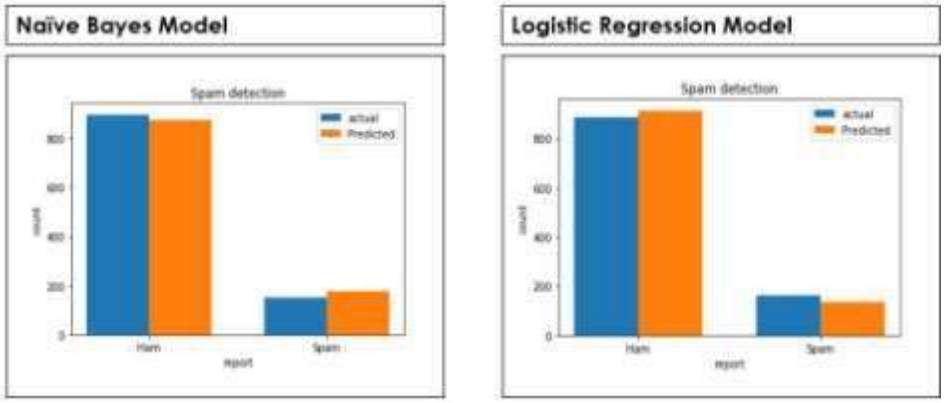


Fig 6.1 80-20 split ration comparison

V. CONCLUSION

We proposed a novel algorithm for enhancing the accuracy of the Naive Bayes Spam Filter. The algorithm was implemented as an enhancement for Naive Bayes Classifier and also tested with logistic regression model. Naive Bayes has a very fast processing speed and allows for a small training set, hence is suitable for real-time spam filtering. We are also using Intelligent Text Modification method to identify messages containing leetspeak and diacritic. We are able to classify email as spam or ham. By creating an addition to Naive Bayes Classifier. We also found that our new addition helped improve ham classification due to the high recall and precision rates. We demonstrated that our algorithm consistently reduced the amount of spam emails misclassified as ham email.

VI. REFERENCES

[1]. Nikhil Govil, Kunal Agarwal, Ashi Bansal, Astha Varshney. “A Machine Learning based Spam Detection Mechanism”, IEEE Xplore Part Number: CFP20K25-ART; ISBN:978-1-7281-4889-2.

[2]. Chode Abhinav, K Jayachandra, Kommu Pranith Kumar, V Sowmya “Spam Mail Detection using Machine Learning”, International journal for applied science & engineering technology research.

[3]. Manoj Sethi, Sumesha Chandra, Vinayak Chaudhary, Yash, “Email Spam Detection using Machine Learning”, International Research Journal of Engineering and Technology (IRJET).

[4]. Nikhil Kumar, Sanket Sonowal, Nishant, “Email Spam Detection Using Machine Learning Algorithms”, IEEE Xplore Part Number: CFP20N67-ART; ISBN: 978-1-7281-5374-2.

[5]. Jyoti Dake, Gunjan Memane, Prerana Katake, Samina Mulani “Email Spam Detection and Prevention using Machine Learning”, International Journal of Advanced Research in Computer and Communication Engineering.

[6]. “An Efficient Spam Filtering using Supervised Machine Learning Techniques” in IJSRCSE, Vol.6, Issue.2, pp.33-37, April (2018).

[7]. Vinodhini. M, Prithvi. D, Balaji. S “Spam Detection Framework using ML Algorithm” in IJRTE ISSN: 2277- 3878, Vol.8 Issue.6, March 2020.

[8]. Deepika Mallampati, Nagaratna P. Hegde “A Machine Learning Based Email Spam Classification Framework Model” in IJITEE, ISSN: 2278- 3075, Vol.9 Issue.4, February 2020.

[9]. Linda Huang, Julia Jia, Emma Ingram, Wuxu Peng, “Enhancing the Naive Bayes Spam Filter through Intelligent Text Modification Detection”, 2018 17th IEEE International Conference on Trust, Security, and Privacy in Computing and Communications.

[10]. http://www.securelist.com/en/threats/spam?chapter=97.

[11]. Deepika Mallampati, K.Chandra Shekar and K.Ravikanth “Supervised Machine Learning Classifier for Email Spam Filtering”, © Springer Nature Singapore Pte Ltd. 2019 and Engineering, https://doi.org/10.1007/978- 981-13-7082-341.

[12]. Suryawanshi, Shubhangi & Goswami, Anurag & Patil, Pramod. (2019). Email Spam Detection: An Empirical Comparative Study of Different ML and Ensemble Classifiers. 69-74. 10.1109/IACC48062.2019.8971582.

[13]. Karim, A., Azam, S., Shanmugam, B., Krishnan, K., & Alazab, M. (2019). A Comprehensive Survey for Intelligent Spam Email Detection.

IEEE Access, 7, 168261-168295. [08907831]. https://doi.org/10.1109/ACCESS.2019.2954791.

[14]. W.A, Awad & S.M, ELseuofi. (2011). Machine Learning Methods for Spam E-Mail Classification. International Journal of Computer Science & Information Technology. 3. 10.5121/ijcsit.2011.3112.

[15]. K. Agarwal and T. Kumar, "Email Spam Detection Using Integrated Approach of Naïve Bayes and Particle Swarm Optimization," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 685-690