SPAM CLASSIFICATION USING NAIVE BAYES ALGORITHM

Shikhammad Ayyubov, Nurlan Akbarov and Asim Rzayev

Abstract

E-mail is one of the foremost popular and regularly used ways of communication thanks to its worldwide accessibility, fast message transfer, and low sending cost. The flaws within the e-mail protocols and therefore the increasing amount of electronic business and financial transactions directly contribute to the rise in email-based threats. Email spam is one of the big problems of the nowadays internet, bringing financial damage to companies and annoying individual users. Spam emails are invading users without their consent and filling their mailboxes. They consume more network capacity also as time in checking and deleting spam emails. The overwhelming majority of Internet users are outspoken in their dislike for spam, although enough of them answer commercial offers that spam remains an efficient source of income for spammers. While most of the users want to do the right thing to avoid and get rid of spam, they need clear and simple guidelines on how to behave. In spite of all the measures taken to eliminate spam, they're not yet eradicated. Also when the countermeasures are over-sensitive, even legitimate emails will be eliminated. Among the approaches developed to prevent spam, filtering is one of the foremost important techniques. Many sorts of research in spam filtering are centered on the more sophisticated classifier-related issues. In recent days, Machine learning for spam classification is a crucial research issue. The effectiveness of the proposed work explores and identifies the use of different learning algorithms for classifying spam messages from email. A comparative analysis among the algorithms has also been presented.

1. INTRODUCTION

The use of the internet has been extensively increasing over the past decade and it continues to be on the ascent. Hence it is suitable to say that the Internet is gradually becoming the main part of everyday life. Internet usage is expected to continue growing and e-mail has become a powerful tool intended for idea and information exchange. Minor time delay during transmission, security of the data being transferred, low costs are a few of the numerous advantages that e-mail enjoys over other physical methods. However, there are a few issues that spoil the efficient usage of emails. Spam email is one of them. In recent years, spam email is so cheap to send, that unwanted messages are sent to a large number of users unsystematically. When a large number of spam messages are received, it is necessary to take a long time to identify spam or non-spam email and their email messages may cause the mail server to crash. To solve the spam problem, there have been several attempts to detect and filter spam emails on the client side. In previous research, many Machine Learning (ML) approaches are applied to the problem, including Bayesian classifiers as Naive Bayes and Support Vector Machine (SVM), and etc. In these approaches, Bayesian classifiers obtained good results by many researchers so that it is widely applied to several filtering software. However, almost all approaches learn and find the distribution of the feature set in only the spam and the non-spam messages. Today, there are many types of spam email, for example, advertisements for the purpose of making money or selling something, urban legends for the purpose of spreading hoaxes or rumors, etc. Moreover, there are HTML emails containing web bugs which is a graphic in an email message designed to monitor who is reading the message. Therefore, some spam emails are judged to be non-spam emails even if we use the existing filtering techniques. In general, the sender of a spam message pursues one of the following tasks: to advertise some goods, services, or ideas, to cheat users out of their private information, to deliver malicious software, or to cause a temporary crash of a mail server. From the point of view of content, spam is subdivided not just into various topics but also into several genres, which result from simulating different kinds of legitimate mail, such as memos, letters, and order confirmations.

2. DATA DESCRIPTION

This data is a selection of mail messages, suitable for use in testing spam filtering systems. Pertinent points:

- All headers are reproduced in full. Some address obfuscation has taken place, and hostnames in some cases have been replaced with "spamassassin.taint.org" (which has a valid MX record). In most cases though, the headers appear as they were received.

- All of these messages were posted to public fora, originated as newsletters from public news websites.

- relying on data from public networked blacklists like DNSBLs, Razor, DCC or Pyzor for identification of these messages is not recommended

- Copyright for the text in the messages remains with the original senders.

Corpus description. It's split into three parts, as follows:

- spam: 500 spam messages, all received from non-spam-trap sources.

- easy_ham: 2500 non-spam messages. These are typically quite easy to differentiate from spam, since they frequently do not contain any spammish signatures (like HTML etc).

- hard_ham: 250 non-spam messages which are closer in many respects to typical spam: use of HTML, unusual HTML markup, coloured text, "spammish-sounding" phrases etc.

- easy_ham_2: 1400 non-spam messages. A more recent addition to the set.

- spam_2: 1397 spam messages. Again, more recent.

Total count: 6047 messages, with about a 31% spam ratio.

The corpora are prefixed with the date they were assembled. They are compressed using "bzip2". The messages are named by a message number and their MD5 checksum.

The "obsolete" dir contains old versions of the corpus, for reference, in the case is needed need to correlate test results using these older versions against the source messages. The messages in those corpora are generally included in the fresher corpora.

4. METHODOLOGY

For analyzing dataset and to predict the performance, supervised machine learning algorithms were adopte. Different algorithms use different biases for generalizing different representations of the data. Therefore, they tend to error on different parts of the instance space. The combined use of different algorithms could lead to the correction of the individual uncorrelated errors. There are two main paradigms for handling an ensemble of different classification algorithms: Classifier Selection and Classifier Fusion. In this paper, the Naive Bayes algorithm was used because of its surprisingly good result and high speed despite the simple algorithm.

4.1 NAIVE-BAYES CLASSIFIER

Naive Bayes-classifier may be a simple probabilistic classifier supported by applying Bayes theorem with strong independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model”. The Naive-Bayes inducer computes conditional probabilities of the category es given the instance and picks the class with the very best posterior. Depending on the precise nature of the probability model, Naive Bayes classifiers are often trained very efficiently during a supervised learning setting. The basic concept of it's to seek out whether an email is spam or not by watching which words are found within the message and which words are absent from it. Naive Bayes classifiers can handle an arbitrary number of independent variables whether continuous or categorical. Given a set of variables, X={X1, X2…..Xd}, we can construct the posterior probability for the event Cj among a set of possible outcomes C = {c1, c2……cd}

$$p(X \mid Cj)\alpha \prod_{k=1}^{d} p(\, xk \mid Cj\,)$$

Now let’s rewrite the formula like this,

$$p(Cj \mid X) \alpha \, p(Cj) \prod_{k=1}^{d} p(xk \mid Cj)$$

Using Bayes rule, we can label the new case with a class Cj that achieves the highest posterior probability.

5. RESULT EVALUATION

The data set was separated into two parts, one part is used as a training data set to produce the prediction model, and the other part is used as a test data set to test the accuracy of our model. The Training data set contains feature values as well as classification of each record.

5.1 MEASURING THE PERFORMANCE

The meaning of a good classifier can vary depending on the domain in which it is used. For example, in spam classification it is very important not to classify legitimate messages as spam as it can lead to. e.g. economic or emotional suffering for the user.

Not-Spam emails wordcloud:

build
come
tri
run
compani
current
group
ilug
set
chang
see
list maintain
say
problem
work
make
url
one
peopl
want
said
time
way
know
thing
go
need
even
file
mail
list
use
new
system
still
well
linux
user
look
think
irish
first
inform list
user group

Spam emails wordcloud:

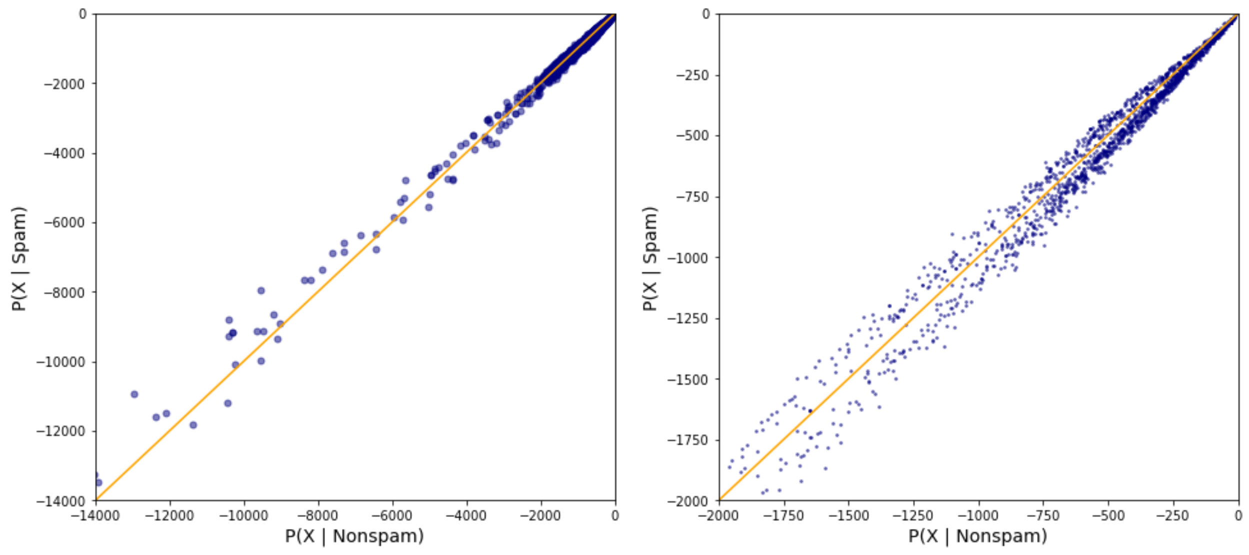
MAIL LIST
USE
EMAIL ADDRESS
FREE
PRODUCT
MARKET
SITE
PLEAS CLICK TIME
HELP
SERVIC
COMPANI
ORDER REPORT
INFORM
WANT
NEED
OFFER
WORK
PEOPL
PROGRAM
MAKE
INTEREST
SEE
INTERNET
ONE
NEW
SYSTEM
AVAIL
CREDIT CARD
ACCOUNT
PERSON
YEAR
MAY
RECEIV EMAIL
BROADCAST EMAIL
LOOK
CUSTOM
SEARCH ENGIN
WISH RECEIV
REMOV MAIL
BUSI
FILL FORM
UNIT STATE
GIVE
CALL
START
U

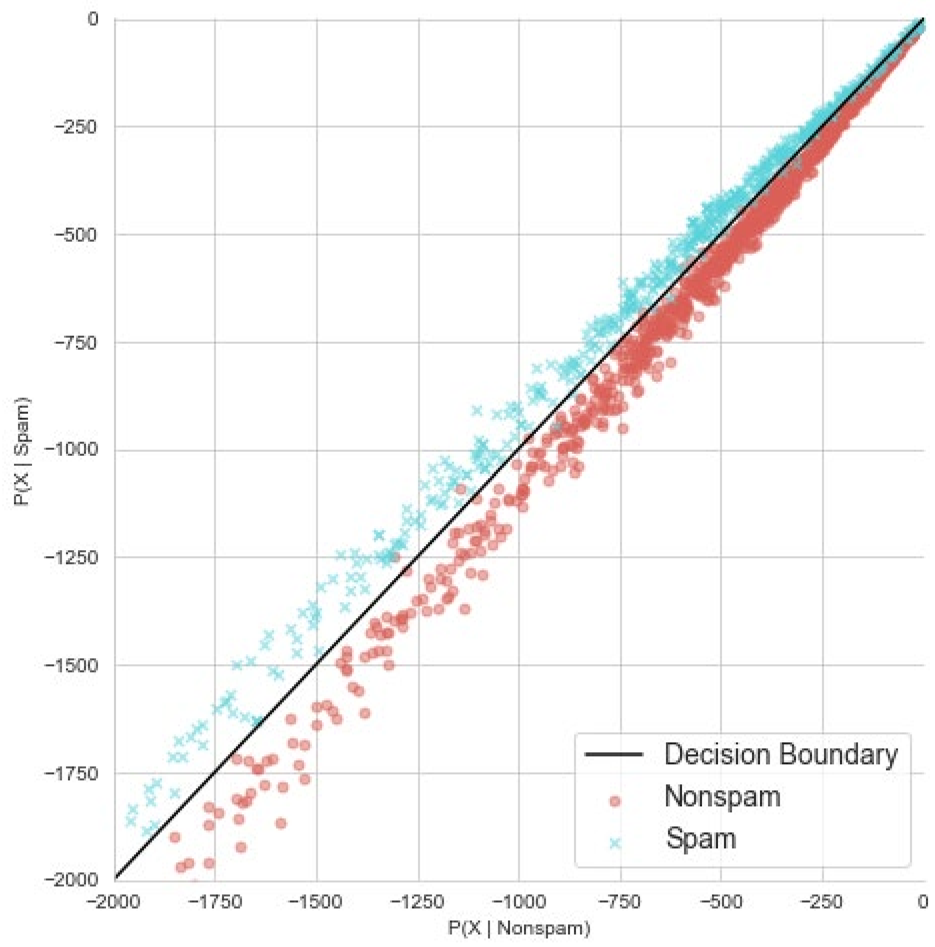
5.2 PRECISION, RECALL and RESULT

A well employed metric for performance measurement in information retrieval is precision and recall. These measures are diligently utilized in the context of spam classification. Recall is that the proportion of relevant items that are retrieved, which during this case is that the proportion of spam messages that are literally recognized. In the spam classification context, precision is the proportion of the spam messages classified as spam over the entire number of messages classified as spam. Thus if only spam messages are classified as spam then the precision is 1. As soon as an honest legitimate message is assessed as spam, the precision will drop below 1. The precision calculates the occurrence of false positives which are good messages classified as spam. When this happens p drops below 1. Such misclassification might be a disaster for the user whereas the sole impact of a coffee recall rate is to receive spam messages within the inbox.Hence it is more important for the precision to be at a high level than the recall rate. A problem when evaluating classifiers is to

find a good balance between the precision and recall rates. Therefore it is necessary to use a strategy to obtain a combined score. One way to achieve this is to use weighted accuracy.
For Naive Bayes Classifier on this data set, fraction classified incorrectly is 2.26%. Accuracy of the model is 97.74%, recall score is 0.966 and precision score is equal to 0.968, and F1-score is 0,97.

Decision boundary:





6. CONCLUSION AND FUTURE WORK

Thus through this paper, a comprehensive analysis of Naive Bayes classifiers using Python programming language was implemented on a dataset. In our work in order to increase the performance of the Naive Bayes FBL algorithm would have been used to produce better results.

REFERENCES

Data source: https://spamassassin.apache.org/old/publiccorpus/
https://www.kdnuggets.com/2020/07/spam-filter-python-naive-bayes-scratch.html
https://in.springboard.com/blog/email-spam-filtering-using-naive-bayes-classifier/
https://towardsdatascience.com/how-to-build-and-apply-naive-bayes-classification-for-spam-filtering-2b8d3308501