Spam/Ham Classification of a Personal Email with Naïve Bayes Classifier

Jongwon Lee http://www.jongwon.net/

Informatics, Indiana University, Bloomington,

Indiana, USA

Abstract

We approach the problem of classifying spam/ham of a personal email with Naïve Bayes classification. We use open source email dataset from enron to train and test the Naïve Bayes classification model then use the model to classify the author's personal emails. For pre-processing we use NLTK lemmatizer, filter human names, and examine word cloud to update stop words. We improve the performance of the model by using GridsearchCV and find the optimal hyperparameters. We evaluate and visualize the model with Naïve Bayes Score and ROC/AUC.

Keywords

Natural Language, Naïve Bayes Classification, CountVectorizer, NLTK, Lemmatizer, stop word, wordcloud, GridsearchCV, ROC, AUC

Introduction

"Generally, spam can be defined as unwanted and unsolicited messages sent to usually a large number of recipients." [1] Naïve Bayes algorithm is a simple machine learning approach for classification problems. However, classifying spam/ham of a personal email is still not solved. The significance and riskiness about spam mails are still an issue. According to Radicati Group's report [2] in the first quarter of 2017, average 269 billion e-mails are being sent a day. According to Express UK report [3], spammers sent e-mails that contain fake i-Tunes login link to carefully selected vulnerable targets.

On top of the threat of spam mails, emails from shopping malls that the users subscribe will send promotions that the users are not interested. However, blocking that email by the address is not ideal because it will block necessary emails from that shopping malls such as order confirmation. Therefore, email services need to add the spam/ham classification algorithm that can classify unwanted promotions to spam and necessary emails to ham. Currently, email services such as Gmail only has the option to block the sender in order to stop those promotions from the website that the customer uses.

This project mainly aims to optimize the Naïve Bayes classification model. The model in this project got 0.9439 Naïve Bayes Score and 0.9663 AUC Score. Moreover, this project wants to answer how effective the model based on a broader dataset is on a narrower personal email dataset. In other words, it tries to justify if a model constructed with the dataset from other users' email does not work on another user's email. Thus, we should construct the classification model for each user. Lastly, it suggests implementing this model and the methods of pre-processing the data on Google Chrome Extension as a practical way to solve the problem.

This project also wants to solve whether pre-processing the data with NLTK stop word, human names, month names and non-words would improve the performance of Naïve Bayes model.

Code

Code used for this project can be found here. https://github.com/FA20-BL-CSCI-P556/final-project-group-9/blob/master/ham_spam.ipynb

Dataset

Two dataset are used in this project. First data is enron spam/ham email dataset consisting 1500 spam and 3672 ham email data from 1999 to 2002. This data is used to create and optimize the Naïve Bayes classification model. The second data is the author's personal email data that is labeled with spam/ham from 2020. We apply the model generated from the first data on the second data.

Pre-processing/Data Cleaning

We used NLTK Lemmatizer in order to group different tense/form of words into a single item. For example, 'send', 'sent', 'sends' would be transformed to 'send' and treated as an identical item.

We included only alphabets since texts like emails will also include non-alphabetical characters such as numbers. We assumed frequency of appearance of numbers will not affect the classification, yet this may not be true.

We filtered human names and month names as well, again assuming these will not affect or negatively affect the classification.

We used NLTK stop word to filter out unnecessary features such as 'I', 'when', 'there' that are known to not contribute to improving classification problems. "The reason why stop words are critical to many applications is that, if we remove the words that are very commonly used in a given language, we can

focus on the important words instead." [4] Thus we got the corpus(machine-readable text) consisting of only alphabets and lemmatized words, and NTLK stop word filtered out.

Definition

This Project uses a term non-word: "A pseudoword or non-word is a unit of speech or text that appears to be an actual word in a certain language, while in fact it has no meaning in the lexicon. Such word is composed of a combination of phonemes which conform to the language's phonotactic rules.".

CountVectorizer

Sklearn's countvectorizer is used for feature extraction(vectorization). We extracted top 500 features from the pre-processed email data and vectorized it. Initially, features included non-words such as 'wa', 'le', 'hou'. We later use word cloud to filter those non-words.

Naïve Bayes Classification

Naïve Bayes classification on spam/ham classification assumes each features are conditionally independent. It is a standard and strong algorithm for classifying texts into categories. With the test size of 30, alpha = 1, fit_prior = True, our initial Naïve Bayes score was 0.9214. AUC Score was 0.9609. In order to figure out the best hyperparameters for Naïve Bayes model, we used GridsearchCV. According to GridsearchCV, alpha = 0.5 and fit_prior = False was better hyperparameters. It enhanced Naïve Bayes score to 0.9246 and AUC score to 0.9616.

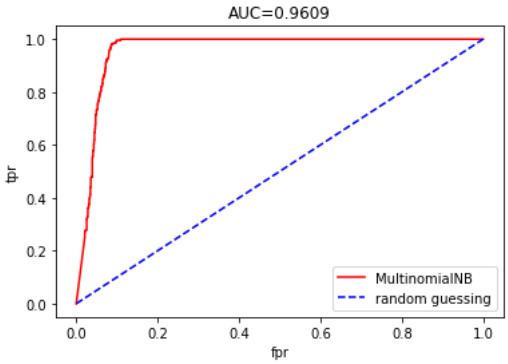

Fig1: Alpha = 1, Fit_prior = True

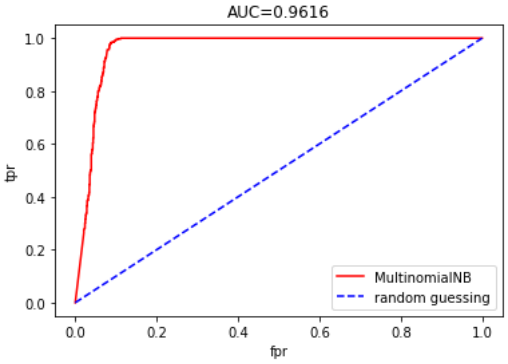

Fig2: Alpha = 0.5, Fit_prior = False

This result implies that the better hpyerparameteres are alpha = 0.5 and fit_prior = False. Fit_prior = False for Naïve Bayes test decides whether to learn class prior probabilities or not. The reason why fit_prior being set to False performs (although very slight) is not discovered in this project.

Word Cloud

The size of the word on a word cloud depends on the frequency of words in a given text as a weighted list. [5] Thus, larger sized words appear on a word cloud implies that word frequently appears in the dataset. With this characteristic, we can not only understand which words are significant for the model but also detect non-words and filter the non-words out.

Instead of going though 500 features and selecting non-word features such as 'wa', 'le', 'hou' we created word cloud and easily detected non-words that were important for the model. We expected filtering out those non-words would improve the performance of the model.

Fig3: Word Cloud for the initial Corpus

We can see words such as 'hou', 'ect', 'enron', 'cc', 'wa' are non-words that are important feature of the model. We decide to filter out those words from the corpus. We then created the word cloud of the updated_corpus.

Fig4: Word Cloud for the updated Corpus

Now, we can assume our features mostly consist of real-words. Ironically, Naïve Bayes score with the updated corpus dropped to from 0.926 to 0.921. This implies filtering out non-word is not effective on improving the performance of the model.

Without the help of word cloud, it is tricky to filter out non-words that are in the corpus and may negatively affect the model.

Fig5: Word Cloud for the spam labeled updated Corpus

Fig6: Word Cloud for the ham labeled updated Corpus

On top of filtering out non-words, with the visualization of word cloud, we can double check if our dataset(corpus) is pre-processed properly and valid. Vocabularies such as ‘pill’, ‘company’, ‘business’, and ‘product’ appearing on the word cloud of spam corpus but not on the ham justifies the pre-processing was successful. We can also easily and effectively understand which vocabulary consists of spam/ham corpus.

Skipping pre-processing to improve NB score

Since filtering out non-words reduced the performance of the NB model, we decided to train and test the model without preprocessing the data (raw data). The performance significantly improved, NB score from 0.9246 to 0.9439 and AUC score from 0.9616 to 0.9663. Moreover, preprocessing takes longer time than training and testing with Naïve Bayes model. Therefore, at least for spam/ham classification the least amount of pre-processing could be done. Human names like ‘john’, month names like ‘jan’, non-words such as ‘ha’, stop words such as ‘when’, ‘I’ all contribute classifying spam/ham emails correctly.

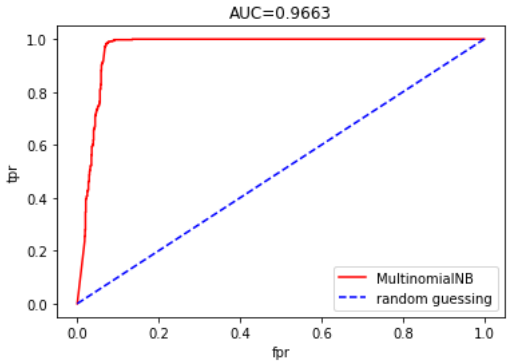

Fig7: Alpha = 0.5, Fit_prior = False, Raw Data

Applying the model on a personal email

With the Naïve Bayes model trained with the enron spam/ham email dataset, we tested a personal email dataset on the author’s inbox. Naïve Bayes score was 0.5, which means the model fails to classify spam/ham.

Fig8: Word Cloud for the spam labeled personal emails

Fig9: Word Cloud for the ham labeled personal emails

Word cloud of enron dataset (fig5 and fig6) and word cloud of personal emails (fig7 and fig8) clearly consists of different vocabularies. Enron spam dataset includes words like 'pill' while personal email spam labeled dataset includes words like 'subscription'. We can tell the spam corpus and feature words highly differs by the era of when the dataset was gathered and who the receiver is. 'pill' would be a strong feature for classifying enron dataset as spam while it never appears on the author's personal spam labeled emails. Therefore, the model must be trained periodically and individually in order to effectively classify spam/ham. Enron dataset was collected from 1999-2002 and author's personal email was collected on 2020. Goal of spam mails will be different by era and thus corpus will contain different vocabularies. If training model for each customer is too costly for the email service provider, grouping the service users by their interest, age, job will also solve the problem.

However, when we tested the raw(not pre-processed) personal email with NB model trained with enron dataset, NB score improved from 0.5 to 0.54. This also contributes to justify pre-processing does not help improve the performance of NB classifier.

Conclusion

This project derives a Naïve Bayes classification model with the help of CountVectorizer, NTLK Lemmatizer, stop word, wordcloud. The model's Naïve Bayes score 0.9439 and AUC score was 0.9663.

Although the model had success with the enron spam/ham dataset, its performance was not amazing when applied to another dataset which consists of the author's personal emails. This project thus concludes, spam/ham classification model must be created for each individual in order to successfully classify the emails.

Moreover, pre-processing task which involves lemmatizer, using only alphabets, filtering human/month names and applying stop words did not improve Naïve Bayes score. This implies in practical usage of spam/ham classification, there is no need to invest on pre-processing. This is also crucial considering pre-processing takes significant time compared to training and testing model.

Future Research

Several limitations needs to be mentioned. This project was not able to get a personal email data that is large enough such as 3000 mails due to the technical issue. The reason why Naïve Bayes parameter fit_prior being set to false performs better overall was not able to be discovered.

We would like to suggest developing a Google Chrome extension as a way to implement this practically. This will contribute to not only stop overflowing personal emails with daily promotion mails but also prevent spams that try to steal personal information. The customer would need to label some emails as spams in order to construct the model.

References

[1] Cormack GV (2006). *Email spam filtering: a systematic review*. Found Trends Inf Retr 1(4):335–455. https://doi.org/10.1561/1500000006

[2] Radicati Group (2020). *Email statistics report*. [http://www.radicati.com/wp/wp-content/uploads/2016/01/Email_Statistics_Report_2016-2020_Executive_Summary.pdf]

[3] Dassanayake, D (2018). *iPhone WARNING - If you get this 'Apple' e-mail do NOT click on it*. [https://www.express.co.uk/life-style/science-technology/933544/Apple-iPhone-warning-scam-e-mail-iTunes-iCloud-App-Store]

[4] Ganesan, Kavita (2019). *What are Stop Words?*. *https://kavita-ganesan.com/what-are-stop-words/#.X9rRlxP0lbU*

[5] Lamantia, Joe (2008). *Text Clouds: A New Form of Tag Cloud?*. Retrieved 2008-09-11.