E4525 Bonus Project: Email Spam Detector

Xijia Tao

Spring 2020

1 Introduction

In this project, we built a spam detector based on simple Naive Bayes classifier techniques. The model with features including structural information, subject and message body achieves 0.994 5-Fold Cross-Validation training AUC and 0.995 testing AUC.

2 Data Source and Pre-processing

The data come from TREC 2005 email spam corpus[1] and 2007 email spam corpus[2]. We mainly focus on TREC 2005 email spam corpus, building email spam detector on its training data and testing the best model picked up by cross-validation on its testing data. After that, we also attempt to predict on 2007 email spam corpus, in order to see whether the spam detector is time consistent.

In pre-precessing step, we generate following features.

- email id: message identifier with format xxx/yyy
- parts (binary): whether it is multi-part
- attachments (binary): whether it contain attachments
- html (binary): whether it contain html section
- subject (string): the subject of the email
- body (string): the playload of text/html part, if it is multi-part. the playload of the email, if it is not multi-part
- links (binary): whether it contain links

We further use bag-of-word to transform the text features, subject and body. We use CountVectorizer in sklearn to build a vector representation of text

	Trec05 Training (5-fold CV)		Trec05 Testing		Trec07	
	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy
Structural	0.678	0.635	0.677	0.635		
Subject	0.982	0.922	0.984	0.924		
Message Body	0.985	0.932	0.987	0.936		
Combined	**0.994**	**0.978**	**0.995**	**0.98**	0.790	0.757

Table 1: Results of Naive Bayes models using different features

respectively for subject and body. CountVectorizer simply constructs a mapping between word and position in a vector, and stores the number of word appearance in the vector. After CountVectorizing, text messages in subject and body are transformed into sparse matrix format.

One thing need to be mentioned here is that we overestimate the cross-validation AUC and accuracy, because building CountVectorizer uses information in validation data. We passed all the training data to CountVectorizer to build dictionary, and thus all the words in validation have already appeared in the dictionary. However, when we used the CountVectorizer built on the training data to transform the testing data, the testing data may contain words that never appeared in our training data.

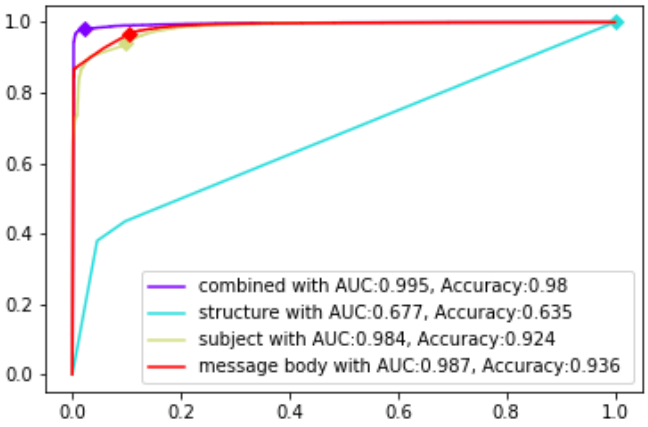
3 Models and Results

We split TREC 2005 dataset into 85% training and 15% testing dataset.

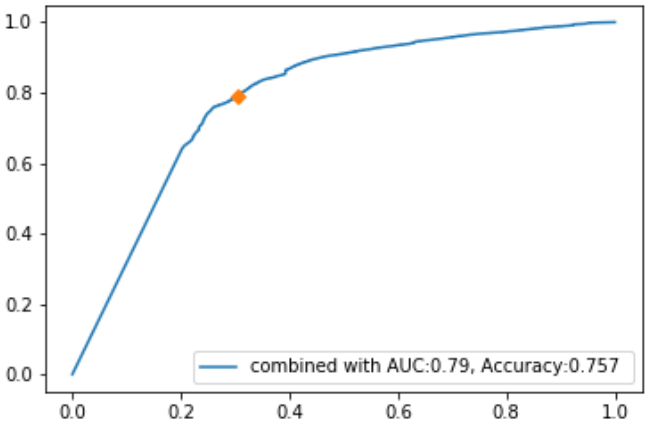
Four Naive Bayes models using different features are trained on TREC 2005 dataset and results are displayed in Table 1. As the name hints, Structural Information Model contains four binary structural features (parts, attachments, html, links). Subject Model contains subject feature. Message Body Model contains body feature. Combined Model contains all the features

According to Table 1, all models involving text features exhibit great spam detect ability. Combined Model is the best one and its AUC achieves 0.994 and 0.995 in training and testing data, which is incredible. Comparing with 5-Fold cross-validation training AUC and Testing AUC, we found that these models perform almost the same in training and testing data, which indicates that there is no over-fitting.

Because the excellent performance of Naive Bayes model, there is no need to build other fancy models. Naive Bayes's good performance is based on the natural characteristics and structure of how a passage is composed of



(a) ROC curve of TREC05 testing data



(b) ROC curve of TREC07 data

Figure 1: ROC curves

words. Conditioned on a topic, how many times a word appears are almost independent among words, which satisfies the assumption of Naive Bayes. Other fancy models may perform better by over-fitting the data, Naive Bayes with correct structural assumptions will definitely perform better in testing data and is more robust.

Combined model retrained on the whole TREC 2005 dataset also attempts to detect spams in TREC 2007 dataset. We found that it doesn't perform well. After comparing with vocabularies apprearing in these two datasets, We found that the reason for poor performance is that ratio of common vocabulary is too small. In other words, the vocabulary people use in emails change so fast, therefore 05 version of spam detector can not detect spams in year 2007. Between 39634 vocabularies in 05's email subject dictionary and 25701 in 07's, there are only 7151 words in common, which account for 28%. Between 408571 vocabularies appearing in 05's email body and 383595 in 07's , there are only 72546 words in common, which account for 19%. This finding tells us that timely updating is necessary in building spam detector. To update data, email apps, like Gmail, would let users to notify them if spams are not correctly detected.

References

[1] TREC05. https://plg.uwaterloo.ca/ gvcormac/treccorpus/

[2] TREC07. https://plg.uwaterloo.ca/ gvcormac/treccorpus07/