Spam Email Classification

Yuepeng Jiang

PID: A53323055

yuj009@eng.ucsd.edu

Abstract

This article is a project work for ECE 225A Probability and Statistic for Data Science leaded by Professor Alon Orlitsky at UC San Diego. Our dataset is from Kaggle under: https://www.kaggle.com/veleon/ham-and-spamdataset. We firstly analyze the dataset and decide to use words in the email content as useful features. After finishing data preprocessing, we build the feature vectors and employ three different modes to classify a certain email as a spam email or a ham email.

Keywords

Feature Vector, Naive Bayes Model, Logistic Regression, Support Vector Machine

1 Introduction

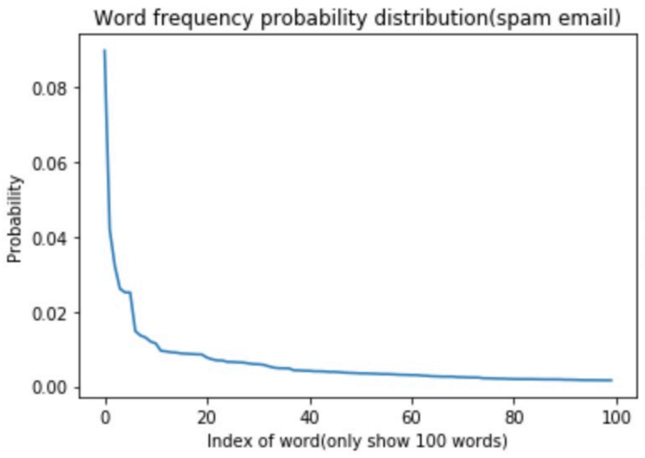
In recent times, the spam email, unwanted commercial bulk email, has become a huge problem on the internet. Spam emails prevent the users from making full and good use of time, storage capacity and network bandwidth. It can also result in untold financial loss to some users who have a poor judgement. This situation often occurs in China because many old people there tend to believe the advertises in the commercial emails. What's worse, some of them are unable to tell whether an email is a fraudulent email or a real email, which results in a huge financial loss to them. So I believe that building a good email classifier that can help people automatically recognize the spam emails is of great importance.

Therefore, for this project, we have decided to build a such spam email classifier. After observation and feature vector construction, we use three models to fit our training dataset. Next, we make predictions of our test set and analyze their performance respectively. The three models we used are Naive Bayes, Support Vector Machine, and Logistic Regression.

2 Dataset Description

We use a dataset from Kaggle which contains 3006 emails in total, 248 spam emails and 2548 ham (non-spam) emails. We separate our dataset into training set, cross-validation set, and test set by 60%, 20%, 20% respectively. The predictions we make are based on the email contents.

As for the concrete statistics in our dataset, the number of total ‘unique’ words are 54872, ‘unique’ words in spam emails are 44304 and ‘unique’ words in ham emails are 52133. What ‘unique’ means in this dataset is defined in Part3 - Data Preprocessing. Figure1 and Figure2 show our selected 100 words’ probability distribution of spam emails and non-spam emails respectively. To clarify, the index in each figure refers to the same word.



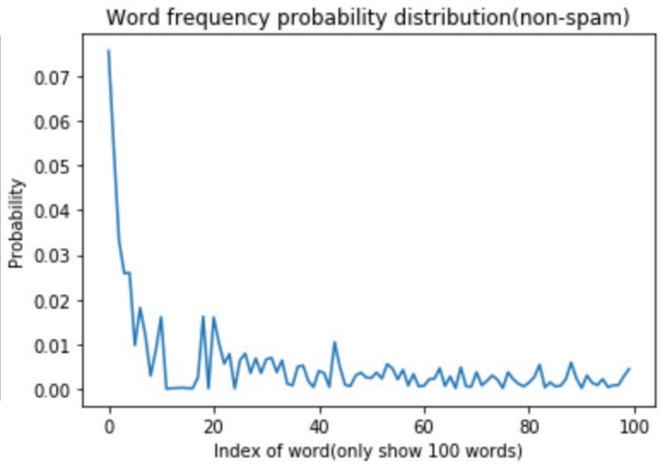


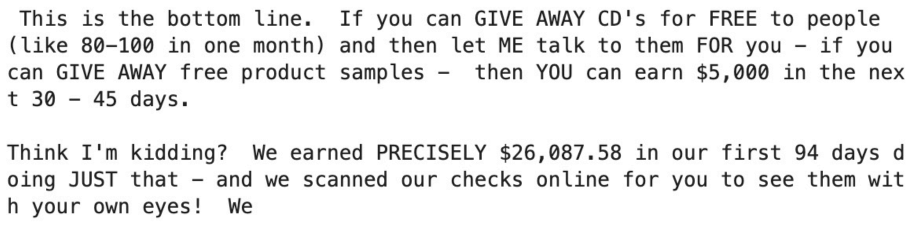
Figure 1: word probability in spam emails

Figure 2: word probability in spam emails

3 Data Preprocessing

A big part in this project is data preprocessing. Since emails’ contents can vary a lot, it’s very important to preprocess them before plugging them into the models. In this part, what we have done is listed below:

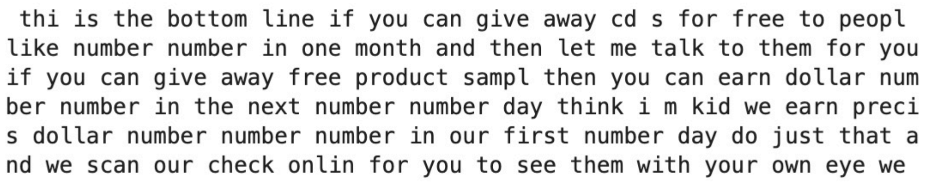
- Convert all words to its lowercase.
- For http links, we just replace it by the word ‘httpaddr’.
- For each ‘$’ sign, we replace it with the word ‘dollar’.
- For each numbers, we replace them with the word ‘number’.
- We treat different forms of a certain word as just one ‘unique’ word. For example, we consider the word set {discount, dicounts, dicounting, discounted} as the word ‘discount’.
- Remove all punctuations.



```
 This is the bottom line.  If you can GIVE AWAY CD's for FREE to people
(like 80-100 in one month) and then let ME talk to them FOR you - if you
can GIVE AWAY free product samples -  then YOU can earn $5,000 in the nex
t 30 - 45 days.

Think I'm kidding?  We earned PRECISELY $26,087.58 in our first 94 days d
oing JUST that - and we scanned our checks online for you to see them wit
h your own eyes!  We
```

Figure 3: Email content example before processing(500 chars)



```
 thi is the bottom line if you can give away cd s for free to peopl
like number number in one month and then let me talk to them for you
if you can give away free product sampl then you can earn dollar num
ber number in the next number number day think i m kid we earn preci
s dollar number number number in our first number day do just that a
nd we scan our check onlin for you to see them with your own eye we
```

Figure 4: Email content example after processing

4 Probabilistic Classification Model - Naive Bayes Model (NBM)

In this section, we apply NBM to make classifications.

Abstractly, Naive Bayes is a conditional probability model: for a certain word, the probabilities that it comes from spam emails or ham emails are different. Say that an email contains a set of words $\{x_1,x_2,...x_n\}$, the probability can be expressed as

$$p(C_k \mid x_1, \dots, x_n)$$

Where 'C' stands for the type of email, and k=0,1 representing the ham email and spam email respectively. Besides, we assume that each word is independent on other words, so using Bayes formula, it can be further expressed as

$$\begin{aligned} p(C_k \mid x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

Firstly, we get the word's frequency probability distribution of spam emails and ham emails from our training set. Then we then use above Naive Bayes Model to make predictions for our test set. The result is shown below. To clarify, precision is the ratio of the number of spam emails you correctly predict over the number of spam emails you predict, and recall is the ratio of the number of spam emails you correctly predict over the number of true spam emails in the dataset.

Precision	1.0
Recall	0.871
Error	0.020
Accuracy	98.01%

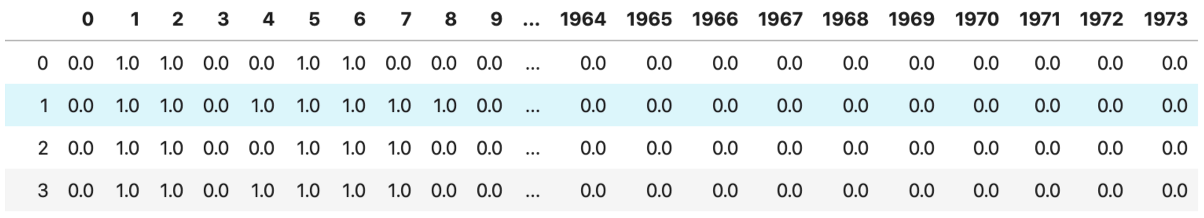
Table 1: Result of NBM

5 Regression Classification Model

5.1 Feature Vector Construction

Before using any regression classification model, the first thing we need to do is convert the emails' contents to feature vectors.

Firstly, we get a word vocabulary of spam emails in our training set, namely a collection of all unique words appearing in that dataset. Secondly, we only keep words with frequency larger than 11. After that, we get a vocabulary of size 1974. Then we scan each email, if its content has the word in the vocabulary, then the value in that position is 1 and 0 otherwise. Thus, an email represents a feature vector of size '1x1974', with 0 or 1 in each position. At last, we build a feature vectors of size '3006x1974'.



	0	1	2	3	4	5	6	7	8	9	...	1964	1965	1966	1967	1968	1969	1970	1971	1972	1973
0	0.0	1.0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	1.0	1.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 5: Feature Vectors

5.2 Logistic Regression

Logistic regression model is a widely used method for 0-1 classification problem. Since our model may overfit the training set, we add a regularization term for this model. The cost function is shown below.

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^i \, log h_\theta(x^i) + (1-y^i) log\left(1-h_\theta(x^i)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j{}^2$$

Where h is the sigmoid function

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T}}$$

The goal is to minimize this function by changing the parameter vector θ which has a size of '1x1974'.After we run our program with different λ, we choose the best λ that gives the lowest error for cross-validation set. In our training set, the best λ is 2 (index is 5).

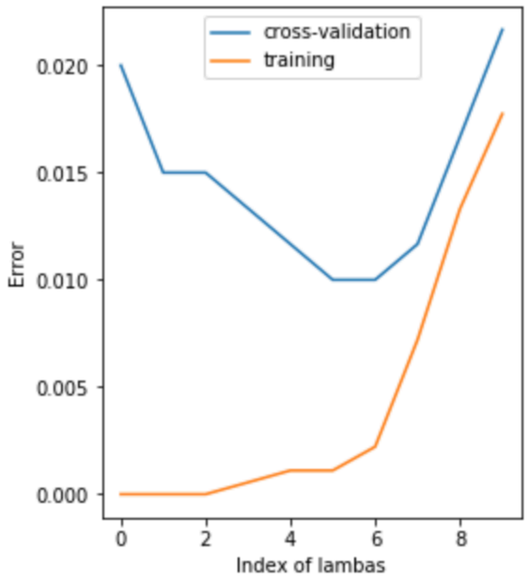


Figure 6: Error of Different λ

Using the best λ, the result of our logistic regression model are shown below.

Precision	0.989
Recall	0.968
Error	0.0066
Accuracy	99.34%

Table 2: Result of LR

5.3 Support Vector Machine (SVM)

Support Vector Machine is an alternative classification model to logistic regression. Given a set of training examples, each marked as a positive label or negative label, a SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier by optimize the soft-margin formulation:

$$\arg \min_{\theta, \alpha, \zeta_i > 0} \frac{1}{2} \|\theta\|_2^2 + C \sum_i \zeta_i$$

such that

$$\forall_i y_i(\theta \cdot X_i - \alpha) \geq 1 - \zeta_i$$

One advantage for SVM is the kernel trick. Since our feature vector's dimension is large(1974 dimension) and the size of our training set is similar to it, we choose linear kernel for our SVM. The performance of SVM for our test set is shown below.

Precision	0.967
Recall	0.957
Error	0.0116
Accuracy	98.84%

Table 3: Result of SVM

6 Result and Conclusion

Model	NBM	LR	SVM
Precision	1.0	0.989	0.967
Recall	0.871	0.968	0.957
Error	0.02	0.0066	0.0116
Accuracy	98.01%	99.34%	98.84%

Table 3: Result of three models

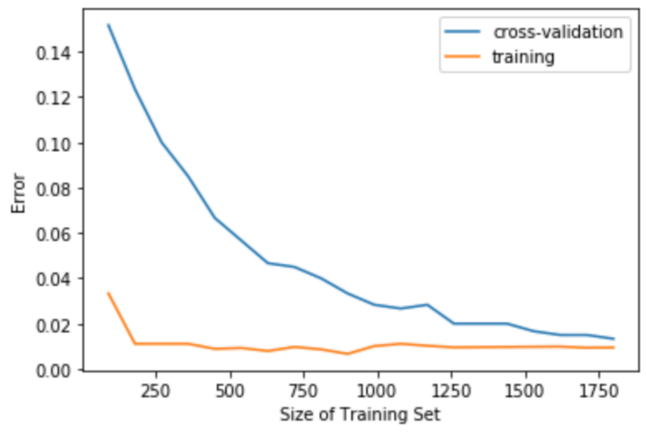


Figure 7: Learning Curve of SVM

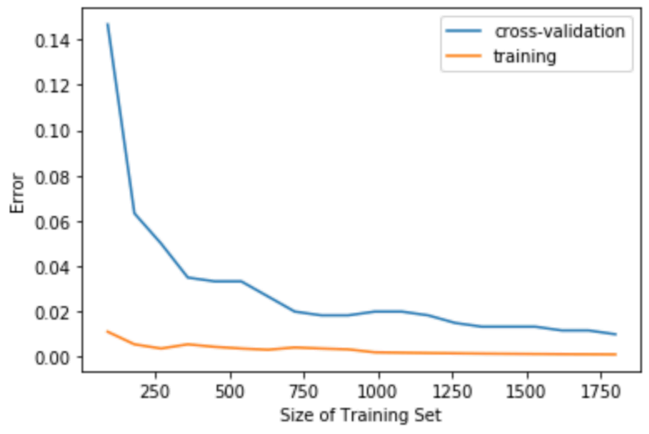


Figure 8: Learning Curve of LR

Analyzing experiment results above, we can draw some conclusions:

- NBM, SVM and LR are all good models for this spam email classification problem
- LR with the best λ has the best performance in this dataset
- The precision of Naive Bayes is 1.0 which means that if you want to make sure that the emails you predict as spam emails are also actually spam emails, then you should use Naive Bayes Model.
- LR model has the best recall value, meaning that if you want to correctly predict as many as possible spam emails, you should choose LR model.
- In the learning curve of SVM and LR, when the size of the training set is small, error tends to be higher than that when the size is large, which means the model may slightly underfit the training set. I think this is probably because of the way I build the feature vectors. In the way I

build the feature vectors, I ignore the word's frequency and just make them all to be 1, so the model needs more data to better fit the parameter θ.

7 Further Improvement

Though the accuracy is very good, our model should be generative, meaning that it should also classify the spam emails from other dataset. Thus, we think that there are some improvement we can make in order to have a better performance on other dataset.

- As discussed above, the first improvement I think is considering the word's frequency in feature vectors
- Add more features. For example, in many junk emails, there are a lot of '!' signs. So add the '!' sign as a new feature may help reduce the error. Also, we can consider the email's subject since it is a good indicator for classification.
- Get more data! This dataset is not large enough to build a general spam email classifier.