Spam Detection

A Bayesian approach to filtering spam

Kunal Mehrotra **Shailendra Watave**

Abstract

The ever increasing menace of spam is bringing down productivity. More than 70% of the email messages are spam, and it has become a challenge to separate such messages from the legitimate ones. We have developed a spam identification engine which employs naïve Bayesian classifier to identify spam. This probabilistic classifier was trained on TREC 2006, a corpus of known spam/legitimate messages and it takes into account a comprehensive set of phrasal and domain specific features (non phrasal features viz. email containing attachments, emails sent from .edu domain etc) that are arrived at by using standard dimensionality reduction algorithms. The cost of classifying a legitimate message as spam (false positive) far outweighs the cost of classifying spam as legitimate (false negative). This cost sensitivity was incorporated into the spam engine and we have achieved high precision and recall, thereby reducing the false positive rates.

Keywords:
Naïve Bayesian Classifier, Support Vector Machines, Precision, Recall

1. Introduction

Spam is an unsolicited email that is sent indiscriminately to mailing lists, individuals and newsgroups. This misuse of the electronic message system is becoming rampant as spamming is economically feasible. A recent study says that more than 70% of the total messages that are sent over the internet are spam [1]. Spam brings down the productivity as users have to sift through their inbox to segregate legitimate email messages from spam. Hence the development of an effective and efficient spam filter is highly imperative.

We have developed a spam identification engine that identifies and segregates spam messages from legitimate ones. The classical naïve Bayesian approach was used to develop the spam filter. The use of naïve Bayesian classifier has become highly prevalent as the ensuing system will be less complex. Naïve Bayesian classifier is a probabilistic classifier based on Bayes' theorem. The theorem assumes that each feature is conditionally independent of each other. The TREC 2006 email corpus was used to train and test our filter. We made use of 70% (approx. 25,475) of the total messages from the corpus to train our filter. The remaining 30% (approx. 10918) of the messages were used to test the filter.

2. Literature Survey

Recently, varied techniques have been applied to identify spam. The technique proposed by Sahami et al was among the first studies that focused on this task. The naïve Bayesian approach was preferred because of its robustness and ease of implementation in cost sensitive decision framework. Jason Rennie's iFile program was the first anti spam filter developed using the Bayes' classifier. Few others have also implemented variations of the above technique. Paul Graham wrote an article "A Plan for Spam" which was intended for the general audiences and was well received.

Other techniques like RIPPER, Ensembles of Decision Trees, Boosting and Instance-based learning, SVM etc. were proposed subsequently. Experiments conducted by Drucker et al. verified the effectiveness of the SVM techniques. The study concluded that SVM and boosting are the top performing methods.

3. Project Description

The objective is to implement a Naïve Bayesian anti-spam filter to segregate spam from ham and measure its efficacy using various cost effective measures. The results are measured-up with a third party filter, LIBSVM based on another classification technique, called Support Vector Machine (SVM).

A supervised learning approach is used to enable the filter to differentiate between spam and ham. The filter is trained on 70% of spam & ham corpus that requires Feature Extraction and calculation of spam probability of the extracted feature, ***fi*** , using a naïve Bayesian expressed as:

$$P(SPAM \mid fi) = \frac{P(fi \mid SPAM)\ P(SPAM)}{P(fi)}$$

$$P(SPAM \mid fi) = \frac{P(fi \mid SPAM)\ P(SPAM)}{P(fi \mid Spam)\ P(SPAM) + \boldsymbol{k}.\ P(fi \mid HAM)\ P(HAM)}$$

We base our calculation on an assumption that a probability an email is either SPAM or NOT is 50%. That is, the prior probabilities: P(SPAM) = P(HAM) = 0.5. A ***k*** factor has been introduced that can be tuned to reduce the number of false positives – the number of HAMS misclassified as SPAMS.

Validation of each in coming email is attained by tokenizing the email and using the pre-calculated spam probability of each feature to classify the incoming email as SPAM or HAM using following naïve

Bayesian expression:

$$P(SPAM| f_1, f_2, f_3 \dots f_i) = \frac{P(f_1, f_2, f_3 \dots f_i \mid SPAM)\, P(SPAM)}{P(f_1, f_2, f_3 \dots f_i)}$$

$$P(SPAM| f_1, f_2, f_3 \dots f_i) = \frac{P(f_1, f_2, f_3 \dots f_i \mid SPAM)\, P(SPAM)}{P(f_1, f_2, f_3 \dots f_i \mid SPAM)\, P(SPAM) + P(f_1, f_2, f_3 \dots f_i \mid HAM)\, P(HAM)}$$

Since naïve Bayes classifier estimates the class-conditional probability by assuming that attributes are conditionally independent, the above equation can be re-written as:

$$P(SPAM| f_1, f_2, f_3 \dots f_i) = \frac{P(SPAM)\, \prod_{i=1}^{n} P(f_i \mid SPAM)}{P(SPAM)\, \prod_{i=1}^{n} P(f_i \mid SPAM) + P(HAM)\, \prod_{i=1}^{n} P(f_i \mid HAM)}$$

Since, P(SPAM) = P(HAM) = 0.5

$$P(SPAM| f_1, f_2, f_3 \dots f_i) = \frac{\prod_{i=1}^{n} P(f_i \mid SPAM)}{\prod_{i=1}^{n} P(f_i \mid SPAM) + \prod_{i=1}^{n} P(f_i \mid HAM)}$$

$$P(SPAM| f_1, f_2, f_3 \dots f_i) = \frac{\prod_{i=1}^{n} P(SPAM \mid f_i)\, P(f_i) / P(SPAM)}{\prod_{i=1}^{n} P(SPAM \mid f_i)\, P(f_i) / P(SPAM) + \prod_{i=1}^{n} P(HAM \mid f_i)\, P(f_i) / P(HAM)}$$

Since, P(SPAM) = P(HAM) = 0.5

$$P(SPAM| f_1, f_2, f_3 \ldots f_i) = \frac{\prod_{i=1}^{n} P(SPAM \mid f_i)\, P(f_i)}{\prod_{i=1}^{n} P(SPAM \mid f_i)\, P(f_i) + \prod_{i=1}^{n} P(HAM \mid f_i)\, P(f_i)}$$

Dividing nominator and denominator by $P(f_i)$ to get:

$$P(SPAM| f_1, f_2, f_3 \ldots f_i) = \frac{\prod_{i=1}^{n} P(SPAM \mid f_i)}{\prod_{i=1}^{n} P(SPAM \mid f_i) + \prod_{i=1}^{n} P(HAM \mid f_i)}$$

Since, $P(SPAM \mid f_i) = 1 - P(HAM \mid f_i)$

$$P(SPAM| f_1, f_2, f_3 \ldots f_i) = \frac{\prod_{i=1}^{n} P(SPAM \mid f_i)}{\prod_{i=1}^{n} P(SPAM \mid f_i) + \prod_{i=1}^{n} (1 - P(SPAM \mid f_i))}$$

Here n = 15.

That is, fifteen most interesting features are considered in the tokenized email to classify it either as SPAM or HAM and the interestingness of each feature is computed as follows:

$$I_f = | 0.5 - P_f |$$

where $P_f = P(SPAM \mid f)$ = Prior probability for SPAM given the feature.

Mistakenly blocking a legitimate (ham) message is more severe than letting a spam message pass the filter. Let,

H -> S denote HAM misclassified as SPAM

S -> H denote SPAM misclassified as HAM

Assuming that H->S is λ times more costly than S->H, we classified a message as spam only if:

$$\frac{P(SPAM \mid f_1, f_2, f_3 \dots f_i)}{P(HAM \mid f_1, f_2, f_3 \dots f_i)} > \lambda$$

Since,

$$P(HAM \mid f_1, f_2, f_3 \dots f_i) \quad = \quad 1 - P(SPAM \mid f_1, f_2, f_3 \dots f_i)$$

the classification criterion can be re-formulated as follows:

$P(SPAM \mid f_1, f_2, f_3 \dots f_i) > t$, with $t = \lambda / (1 + \lambda)$

Here λ determines the severity of penalty for misclassifying a legitimate email as SPAM. This cost sensitivity is incorporated into the system as threshold, given as $\lambda / (1 + \lambda)$. The model is re-configured and evaluated on different severity levels of λ. The table below details various levels of cost sensitivity of model that has been considered:

λ	**Threshold $t = \lambda / (1 + \lambda)$**	**What it means to have such cost sensitivity?**
999	0.999	Blocked messages are discarded without further processing.
9	0.9	Blocking a legitimate message is penalized mildly more than letting a spam message pass. To model the fact that re-sending a blocked message involves more work (by the sender) than manually deleting a spam message
1	0.5	If the recipient does not care much about losing a legitimate message.

4. Cost-sensitive evaluation measures

The classification model is usually evaluated on accuracy and error rate. Since the cost of classifying a legitimate message as spam (false positive) far outweighs the cost of classifying spam as legitimate (false negative), the cost sensitivity is considered in accuracy and error rate by treating each legitimate message as if it were λ messages. As a result, when a legitimate message is mis-classified, it will count as λ errors.

Thus,

$$\text{Wacc} = \frac{\lambda \, . \, n_{L\text{->}L} + n_{S\text{->}S}}{\lambda \, . \, N_L + N_s} \qquad \text{WErr} = \frac{\lambda \, .n_{L\text{->}S} + n_{S\text{->}L}}{\lambda \, . \, N_L + N_s}$$

A better measure of the filter is the relative comparison of the results of the model with a case when no filter is used. That is how the filter measure up with the baseline case when no filter is used. A new measure, called Total Cost Ratio (TCR) is considered for the same.

A TCR is defined as the ratio of "Baselined Weighted Error rate to Weighted Error rate". That is,

$$\text{TCR} = \text{WErr}^b \ / \ \text{WErr} = \frac{Ns}{\lambda \, . \, n_{L\text{->}S} + n_{S\text{->}L}}$$

where,

$$\text{WErr}^b = \text{Baselined Weighted Error rate} = \frac{Ns}{\lambda \, . \, N_L + N_S}$$

A higher TCR indicate a better performance. If the TCR is less than 1, than not using the filter is better. An effective spam filter should be able to achieve a TCR value greater than 1 to be useful in real world applications.

As shown in the ensuing experiments, we have run our filter on different values of λ for variety of test cases to evaluate the efficacy of the filter under different scenarios.

5. Experimental Results

We conducted a series of experiments and the results are tabulated as under. Each test case consisted of a collection of spam and non spam messages. All the tests were executed with a three different values of λ. The messages that were part of the test cases are:

Test Case 1:

- A total of 5000 messages consisting of,
  - 2500 non spam messages from the training set
  - 2500 spam messages from the training set

Test Case 2:

- A total of 5000 messages consisting of,
  - 1250 non spam messages from the test set
  - 1250 spam messages from the test set
  - 1250 non spam messages from the training set
  - 1250 spam messages from the training set

Test Case 3:

- A total of 5000 messages consisting of,
  - 2500 non spam messages from the test set
  - 2500 spam messages from the test set

Test Case 4:

- A total of 10917 messages consisting of,
  - 3778 non spam messages test set
  - 7139 spam messages from the test set

Test Case	λ	Spam Precision	Weighted Accuracy	TCR
Test Case 1	1	99.92%	99.92%	625
Test Case 2	1	98.04%	98.04%	25.51
Test Case 3	1	96.86%	96.86%	15.92
Test Case 4	1	96.46%	96.46%	18.49
Test Case 1	9	99.92%	99.98%	625
Test Case 2	9	97.92%	99.45%	18.38
Test Case 3	9	96.70%	99.02%	10.20
Test Case 4	9	96.38%	98.55%	11.99
Test Case 1	999	99.92%	99.99%	625
Test Case 2	999	97.72%	99.83%	0.6088
Test Case 3	999	96.36%	99.71%	0.3487
Test Case 4	999	95.84%	99.56%	0.434

Table 1. Results on TREC 2006 corpus.

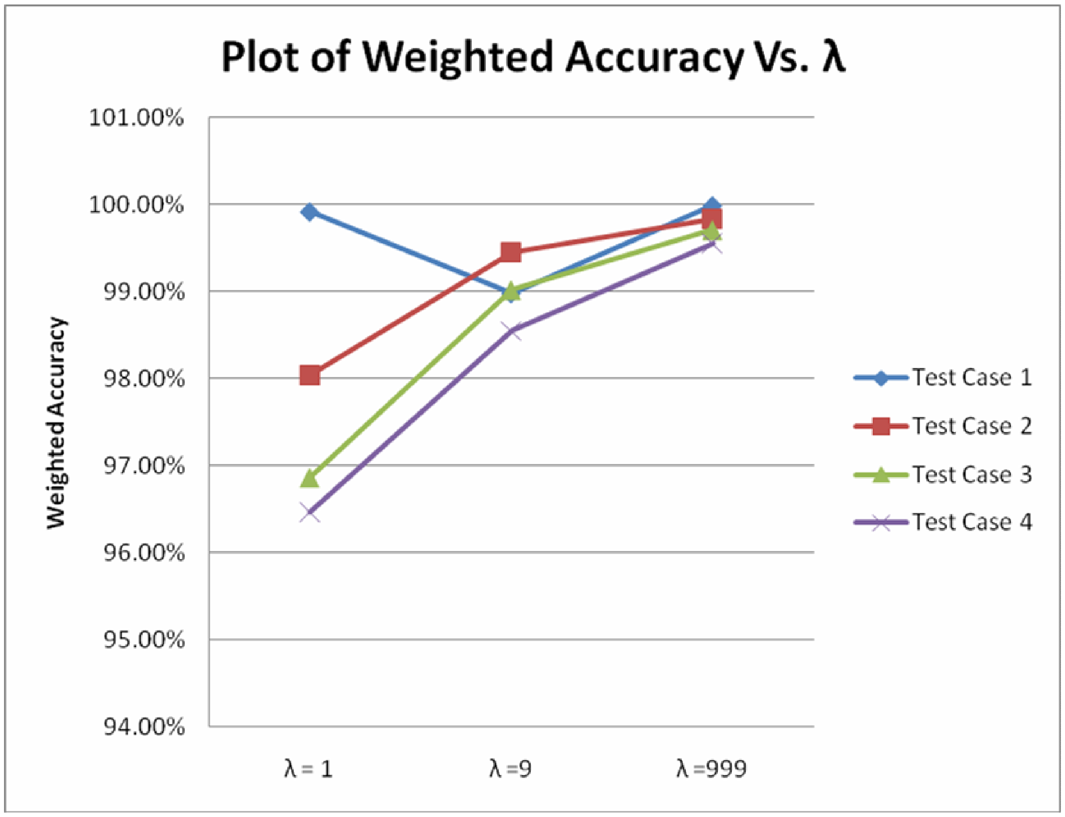
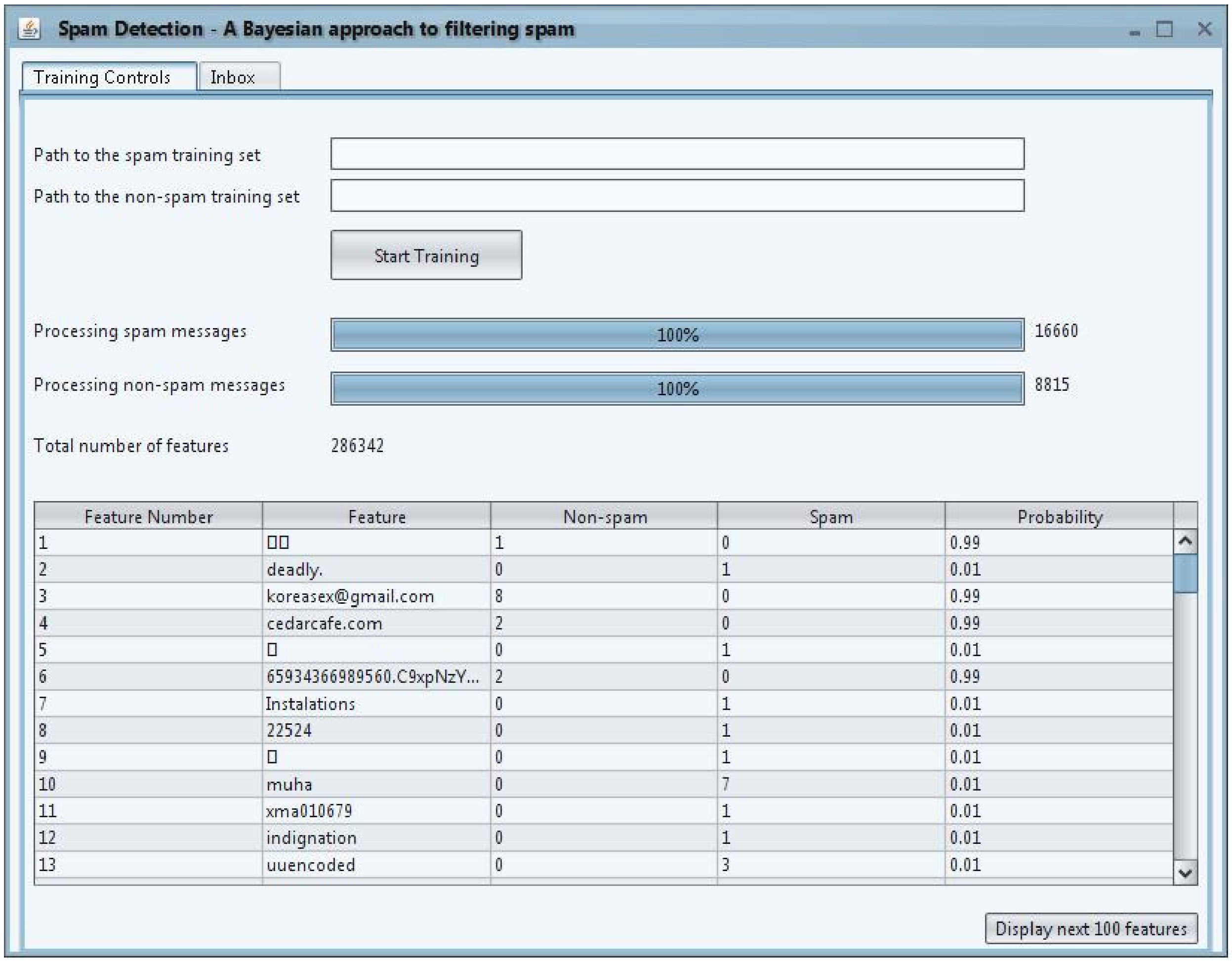


Figure 1: Weighted Accuracy vs. λ for different Test inputs

Figure 2: TRC vs. λ for different Test inputs

6. Screenshots

Screenshots of the Spam filter have been shown below to demonstrate the working of the application.


Spam Detection - A Bayesian approach to filtering spam
Training Controls
Inbox
Start Training
Display next 100 features

Path to the spam training set

Path to the non-spam training set

Processing spam messages 100% 16660

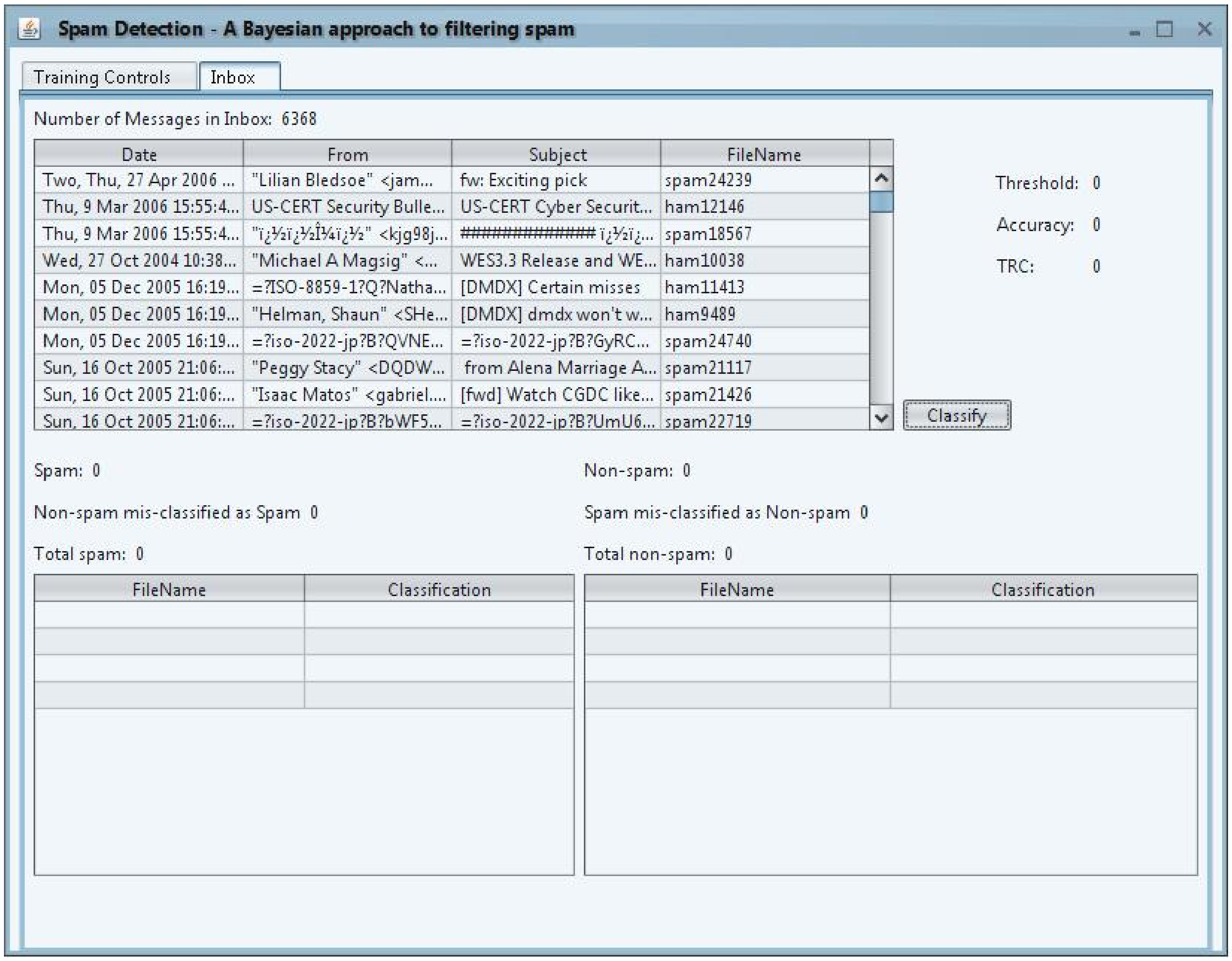
Processing non-spam messages 100% 8815

Total number of features 286342

Feature Number	Feature	Non-spam	Spam	Probability
1	□□	1	0	0.99
2	deadly.	0	1	0.01
3	koreasex@gmail.com	8	0	0.99
4	cedarcafe.com	2	0	0.99
5	□	0	1	0.01
6	65934366989560.C9xpNzY...	2	0	0.99
7	Instalations	0	1	0.01
8	22524	0	1	0.01
9	□	0	1	0.01
10	muha	0	7	0.01
11	xma010679	0	1	0.01
12	indignation	0	1	0.01
13	uuencoded	0	3	0.01

The above screenshot is the Training controls screen. An optional textbox is presented to provide the path to the spam and ham training sets. Once the training is completed the first 100 features are displayed in the table at the bottom of the screen.

In the next screenshot we load an inbox with sample messages as shown below.



Spam Detection - A Bayesian approach to filtering spam

Number of Messages in Inbox: 6368

Date	From	Subject	FileName
Two, Thu, 27 Apr 2006 ...	"Lilian Bledsoe" <jam...	fw: Exciting pick	spam24239
Thu, 9 Mar 2006 15:55:4...	US-CERT Security Bulle...	US-CERT Cyber Securit...	ham12146
Thu, 9 Mar 2006 15:55:4...	"ï¿½ï¿½Î¼ï¿½" <kjg98j...	################ ï¿½ï¿...	spam18567
Wed, 27 Oct 2004 10:38...	"Michael A Magsig" <...	WES3.3 Release and WE...	ham10038
Mon, 05 Dec 2005 16:19...	=?ISO-8859-1?Q?Natha...	[DMDX] Certain misses	ham11413
Mon, 05 Dec 2005 16:19...	"Helman, Shaun" <SHe...	[DMDX] dmdx won't w...	ham9489
Mon, 05 Dec 2005 16:19...	=?iso-2022-jp?B?QVNE...	=?iso-2022-jp?B?GyRC...	spam24740
Sun, 16 Oct 2005 21:06:...	"Peggy Stacy" <DQDW...	from Alena Marriage A...	spam21117
Sun, 16 Oct 2005 21:06:...	"Isaac Matos" <gabriel....	[fwd] Watch CGDC like...	spam21426
Sun, 16 Oct 2005 21:06:...	=?iso-2022-jp?B?bWF5...	=?iso-2022-jp?B?UmU6...	spam22719

Threshold: 0

Accuracy: 0

TRC: 0

Spam: 0

Non-spam mis-classified as Spam 0

Total spam: 0

FileName	Classification

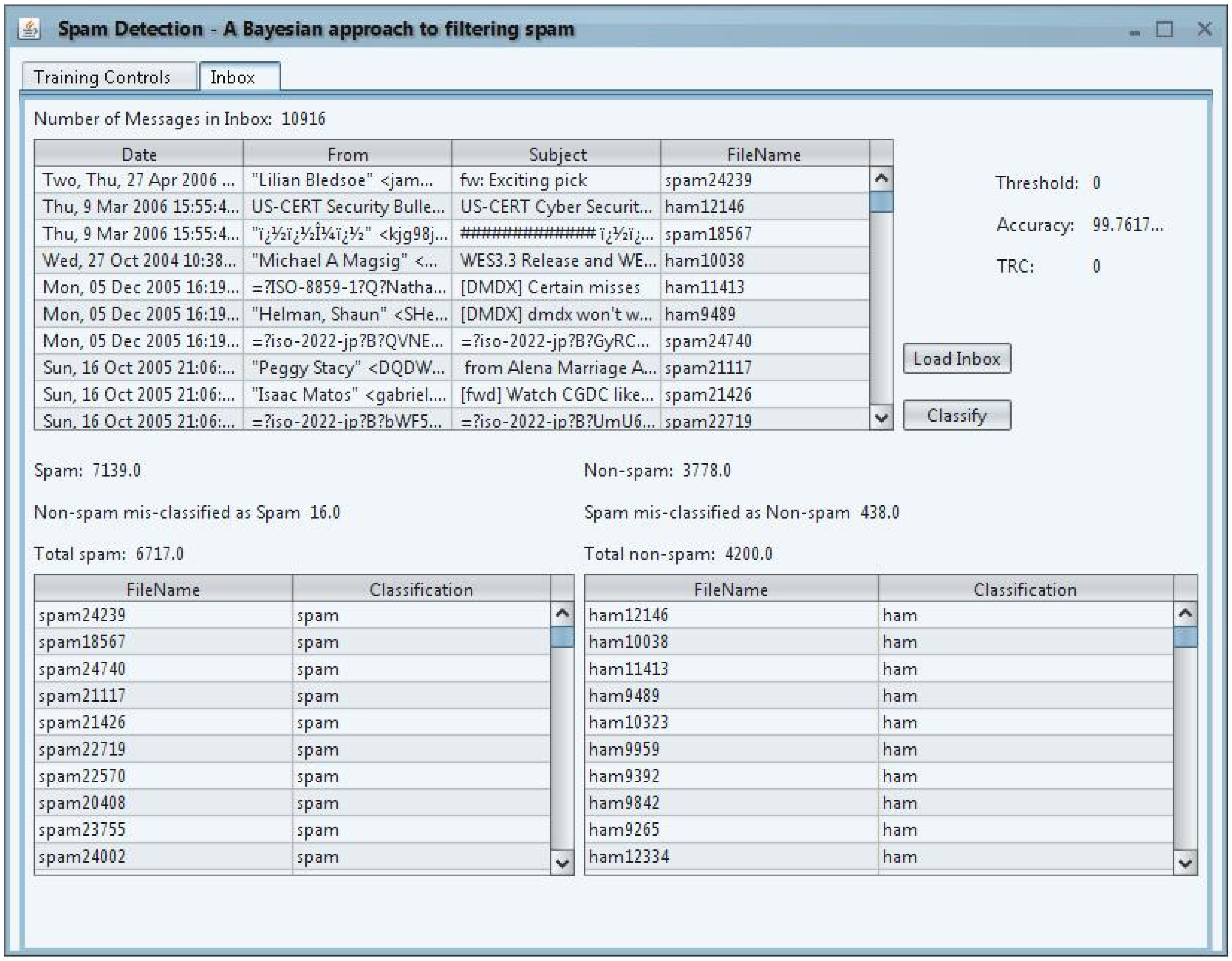
Non-spam: 0

Spam mis-classified as Non-spam 0

Total non-spam: 0

FileName	Classification

Now we test how the filter classifies the sample messages into spam and ham messages. The results are provided in the table as shown below.


Spam Detection - A Bayesian approach to filtering spam

Training Controls | Inbox

Number of Messages in Inbox: 10916

Date	From	Subject	FileName
Two, Thu, 27 Apr 2006 ...	"Lilian Bledsoe" <jam...	fw: Exciting pick	spam24239
Thu, 9 Mar 2006 15:55:4...	US-CERT Security Bulle...	US-CERT Cyber Securit...	ham12146
Thu, 9 Mar 2006 15:55:4...	"ï¿½ï¿½Î¼ï¿½" <kjg98j...	################ ï¿½ï¿...	spam18567
Wed, 27 Oct 2004 10:38...	"Michael A Magsig" <...	WES3.3 Release and WE...	ham10038
Mon, 05 Dec 2005 16:19...	=?ISO-8859-1?Q?Natha...	[DMDX] Certain misses	ham11413
Mon, 05 Dec 2005 16:19...	"Helman, Shaun" <SHe...	[DMDX] dmdx won't w...	ham9489
Mon, 05 Dec 2005 16:19...	=?iso-2022-jp?B?QVNE...	=?iso-2022-jp?B?GyRC...	spam24740
Sun, 16 Oct 2005 21:06:...	"Peggy Stacy" <DQDW...	from Alena Marriage A...	spam21117
Sun, 16 Oct 2005 21:06:...	"Isaac Matos" <gabriel....	[fwd] Watch CGDC like...	spam21426
Sun, 16 Oct 2005 21:06:...	=?iso-2022-jp?B?bWF5...	=?iso-2022-jp?B?UmU6...	spam22719

Threshold: 0

Accuracy: 99.7617...

TRC: 0

Load Inbox

Classify

Spam: 7139.0

Non-spam mis-classified as Spam 16.0

Total spam: 6717.0

Non-spam: 3778.0

Spam mis-classified as Non-spam 438.0

Total non-spam: 4200.0

FileName	Classification
spam24239	spam
spam18567	spam
spam24740	spam
spam21117	spam
spam21426	spam
spam22719	spam
spam22570	spam
spam20408	spam
spam23755	spam
spam24002	spam

FileName	Classification
ham12146	ham
ham10038	ham
ham11413	ham
ham9489	ham
ham10323	ham
ham9959	ham
ham9392	ham
ham9842	ham
ham9265	ham
ham12334	ham

7. Conclusion

The efficacy of our filter engine is evaluated against three levels of penalty ($\lambda = 1$, $\lambda = 9$, $\lambda = 999$). A high value (> 1) of the cost sensitive measure – Total Cost Ratio, on $\lambda = 9$ (threshold = 0.9) suggests that our filter is fit to be used in real world applications. However, the performance of the filter degrades to TRC < 1 when a threshold of 0.999 (for $\lambda = 0.999$) is enforced, thus making the model infeasible when blocked messages are straightaway deleted.

The comparison of Naïve Bayesian approach with SVM technique is still in works. We are in the process of fine tuning the penalty parameters C, k so as to achieve an improved accuracy. Some of our preliminary work around the same is as follows:

Testcases	λ 1:1 (Training set Optimized with Cross Validation Accuracy = 85.4132%)	λ 1:9	λ 1:999
Test 1 :	88.96 (4448/5000)	Accuracy = 79.72% (3986/5000) (classification)	Accuracy = 76.44% (3822/5000)
Test 2 :	Accuracy = 81.84% (4092/5000)	Accuracy = 50% (2500/5000)	Accuracy = 50% (2500/5000)
Test 3 :	Accuracy = 65.36% (3268/5000)	Accuracy = 50% (2500/5000)	Accuracy = 50% (2500/5000)
Test 4 :	Accuracy = 75.387% (8230/10917)	Accuracy = 34.6066% (3778/10917)	Accuracy = 34.6066% (3778/10917)

As can be seen that the accuracy with test-1 is 88.96% when 2500 SPAM 'training' and 2500 HAM 'training' messages are validated on SVM. This is quite low considering SVM filter classifies on a part of training set. As suggested by Chih-Jen Lin et al [7], a "grid-search" on C and □ using cross-validation is being performed. All the possible pairs of (C, □) are being tried and the one with the best cross-validation accuracy is picked. It is suggested to try exponential growing sequences of C and □ to identify good parameters (for example, C = 2^{-5}; 2^{-3};…, 2^{15}, □ = 2^{-15}; 2^{-13};…, 2^{3}). We are hopeful to configure libSVM to achieve satisfactory accuracy on the training set in the coming days.

References

[1] Androutsopoulos, J. Koutsias, K.V. Chandrinos, George Paliouras, and C.D. Spyropoulos (2000). An Evaluation of Naive Bayesian Anti-Spam Filtering.

[2] Cormack V. Gordon & Lynam R. Thomas (2006). Overview of the TREC 2006 Spam Track.

[3] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz (1998). A Bayesian approach to filtering junk e-mail.

[4] Mingjun Lan, Walei Zhou(2005). Spam Filtering based on Preference Ranking.

[5] Paul Graham(2002). A Plan for Spam, http://paulgraham.com/spam.html

[6] Ahmed Obied. Bayesian Spam Filtering.

[7] Chih-Chung Chang and Chih-Jen Lin (2001). LIBSVM : a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm