

# Email Classification

---

BEREND TOBER & MATTHEW BUHLER

# Email Classification

---

How do traditional spam filters work?

Our Basic Approach

Tools:

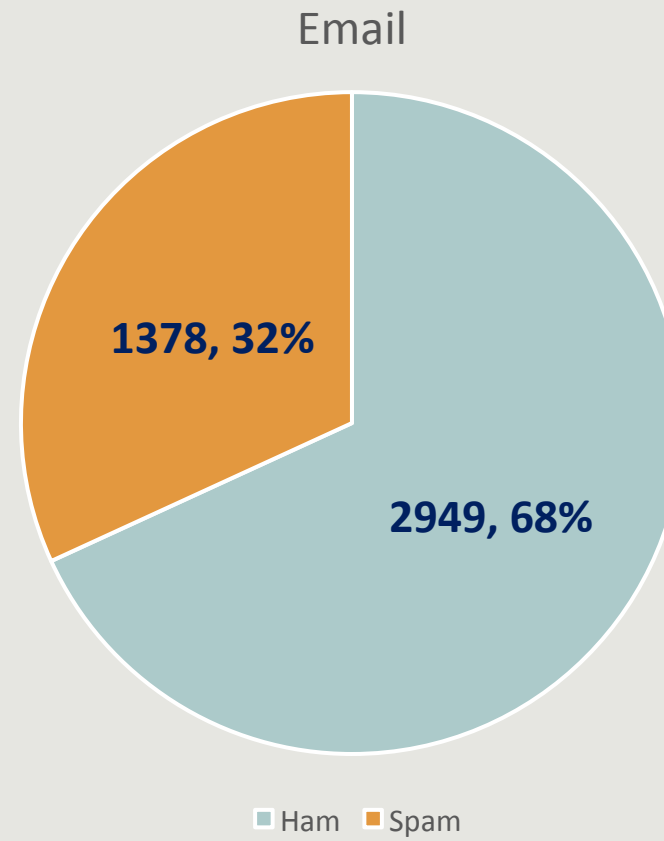
- Python
- bash/sed/awk
- Spam Assassin
- Postfix
- R

# Raw Data

---

4,327 Emails

- 1,378 Spam Messages
- 2,949 Ham Messages



# What does our data look like?

---

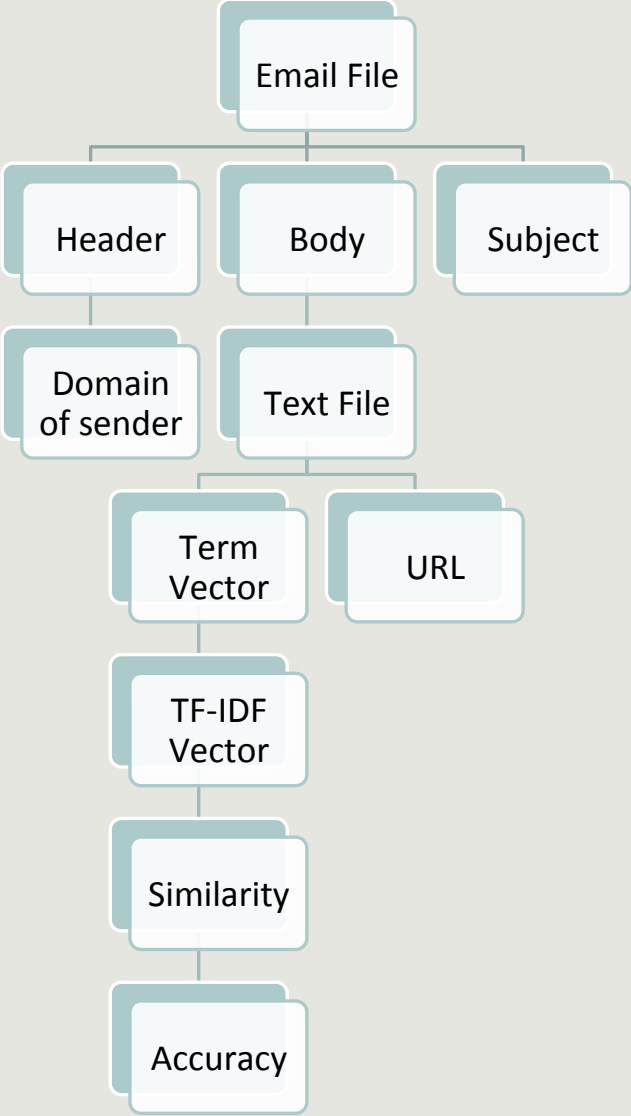
From - Fri Apr 16 10:29:33 2010  
Return-Path: [dapotor4203@mtnl.net.in](mailto:dapotor4203@mtnl.net.in)  
X-Original-To: [m0620212@mail.csmining.org](mailto:m0620212@mail.csmining.org)  
Delivered-To: [m0620212@mail.csmining.org](mailto:m0620212@mail.csmining.org)  
Received: from mail3.csmining.org (localhost [127.0.0.1])  
by mail3.csmining.org (csminingorg Mail) with ESMTP id C7C9B16B91  
for <m0620212@mail.csmining.org>; Thu, 15 Apr 2010 21:14:12 +0900 (JST)  
Received: from spamgw.csmining.org (spamgw.csmining.org [192.168.18.120])  
by mail3.csmining.org (csminingorg Mail) with ESMTP id C3FC816B43  
for <m0620212@mail.csmining.org>; Thu, 15 Apr 2010 21:14:12 +0900 (JST)  
Date: Thu, 15 Apr 2010 21:14:10 +0900 (JST)  
Message-Id: [201004151214.o3FCEAR7008043@gw1.csmining.org](mailto:201004151214.o3FCEAR7008043@gw1.csmining.org)  
Received: from mtnl.net.in (triband-mum-120.60.8.28.mtnl.net.in [120.60.8.28])  
by mx2.csmining.org (csminingorg MX Server2) with ESMTP id 421E96D  
for <hibody@csmining.org>; Thu, 15 Apr 2010 21:14:10 +0900 (JST)  
From: "Medicines from Pfizer" [dapotor4203@mtnl.net.in](mailto:dapotor4203@mtnl.net.in)  
To: [hibody@csmining.org](mailto:hibody@csmining.org)  
Subject: Enter now, hibody, 75% off

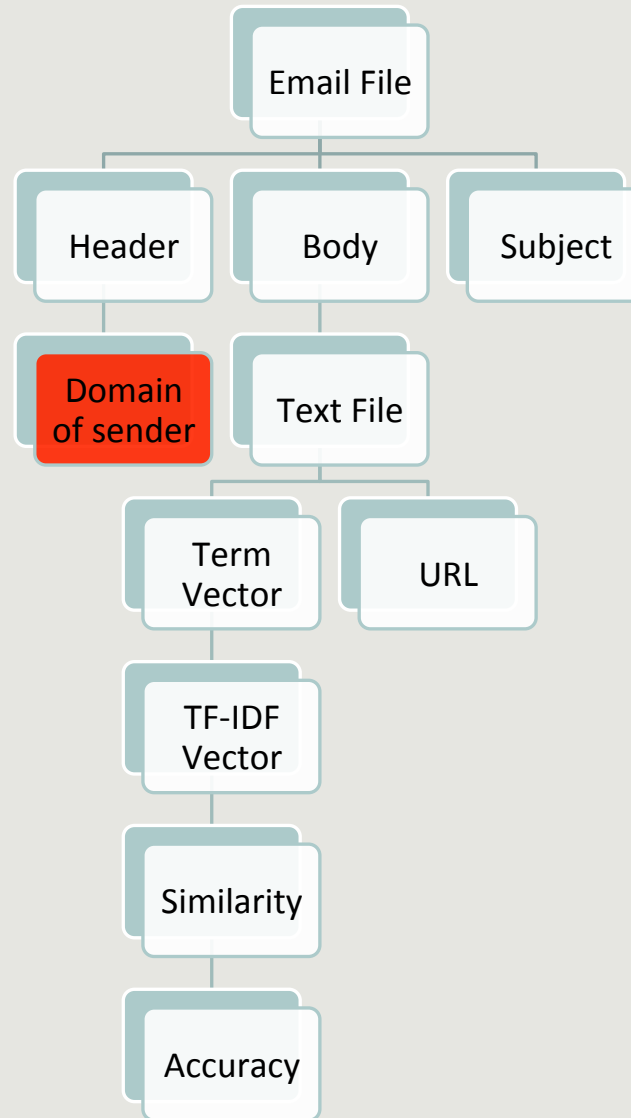
# What does our data look like?

---

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>
having National frequently would view</title>
</head>
<body link="#003366" alink="#003366" vlink="#003366">
<div align="center">
<table border="0" cellpadding="0" cellspacing="0" width="728">
<tbody>
<tr>
<td align="center" width="730">
<div style="padding: 5px; font-family: Arial,Helvetica,sans-serif; font-size: 11px; margin-top: 10px; background-color: rgb(239, 239, 239); color: rgb(102, 102, 102); margin-bottom: 3px;">
To view this email as a web page, <a href="http://f5.pharmlalo.ru/?seufyenow=d87b2b17dd">click here.</a>
</div>
<table bgcolor="#ffffff" border="0" cellpadding="0" width="730">
<tbody>
<tr>
<td align="left" valign="top" width="420">
<div style="font-size: 13px; font-family: arial,Helvetica,sans-serif;">
<div style="color: rgb(51, 51, 51); font-size: 16px; font-weight: 700;">
Thu, April 15, 2010</div>
</div>
</td>
<td width="10">&nbsp;</td>
</tr>
</tbody>
</table>
```

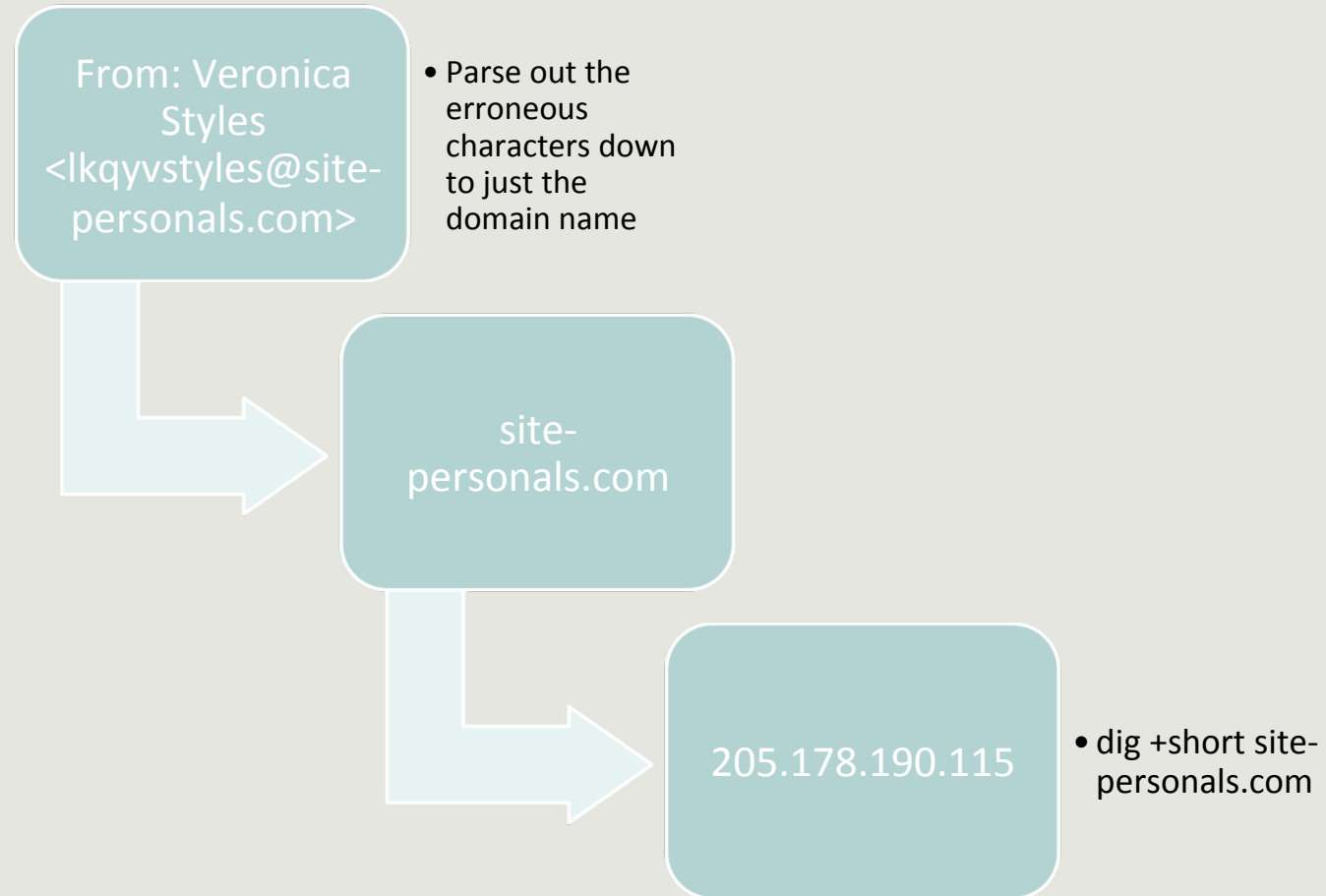
Preprocessing Data:





# Sender Reputation

---

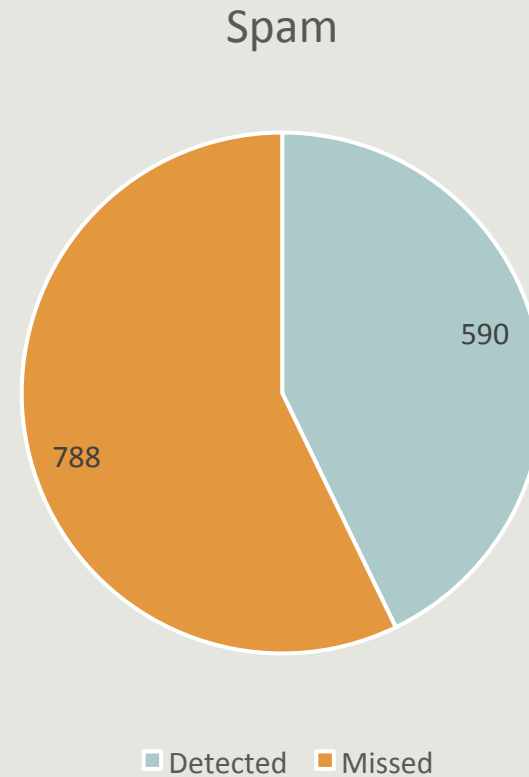


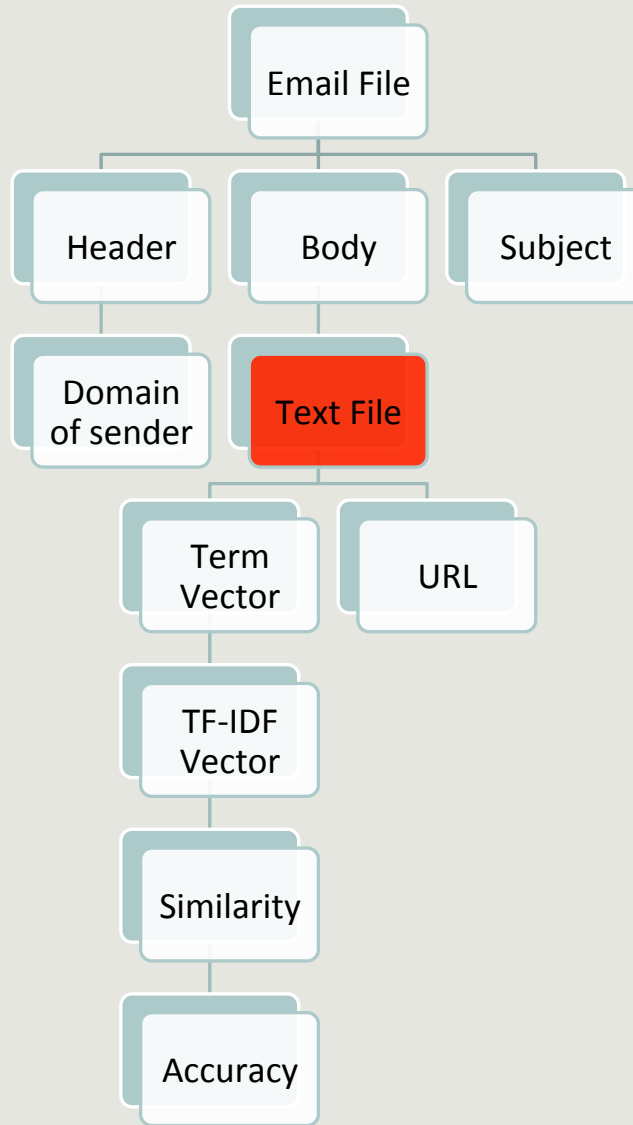


# Results

---

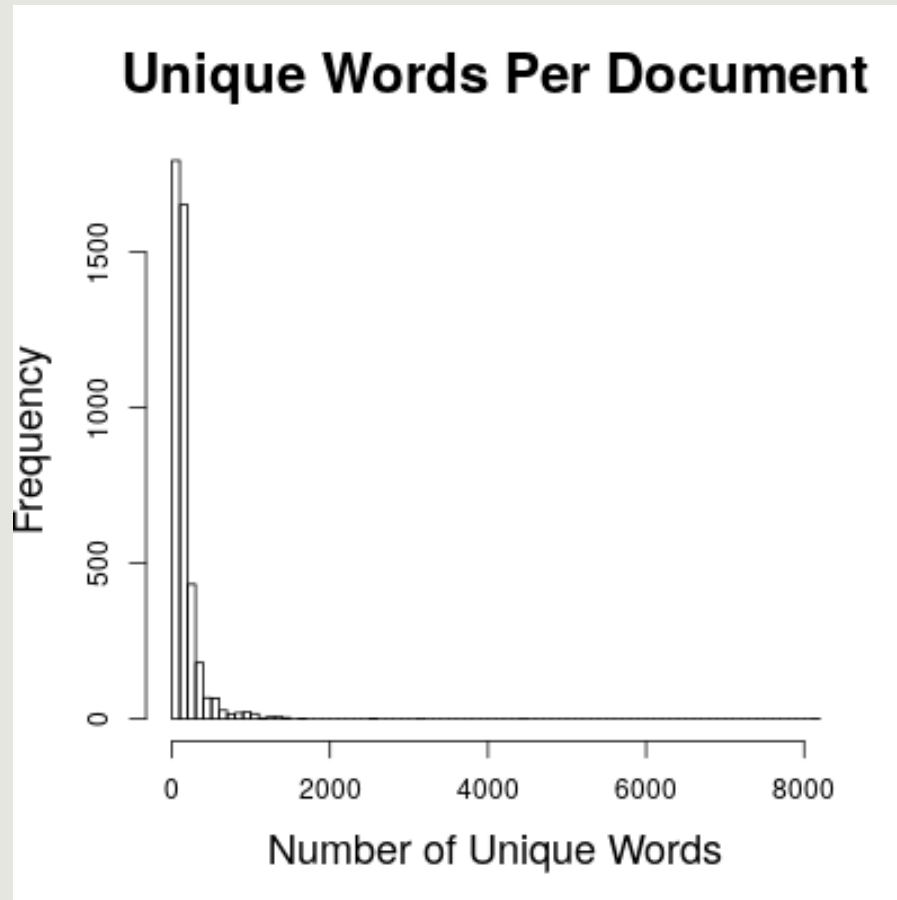
42.8% of Spam messages were successfully identified based solely on the networks.





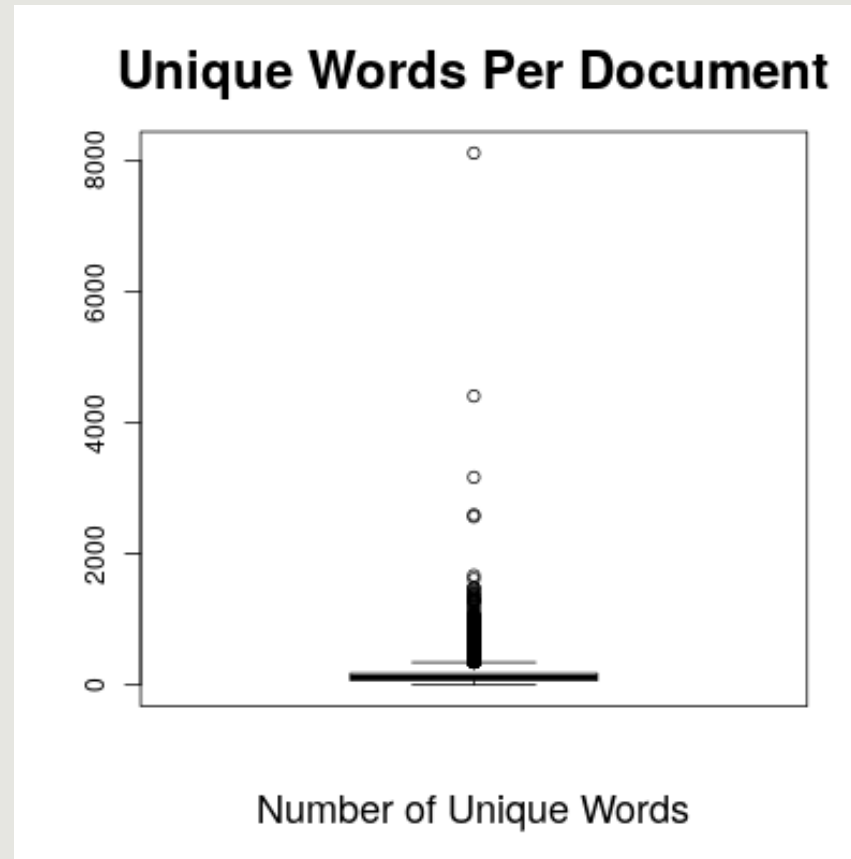
# Data

---



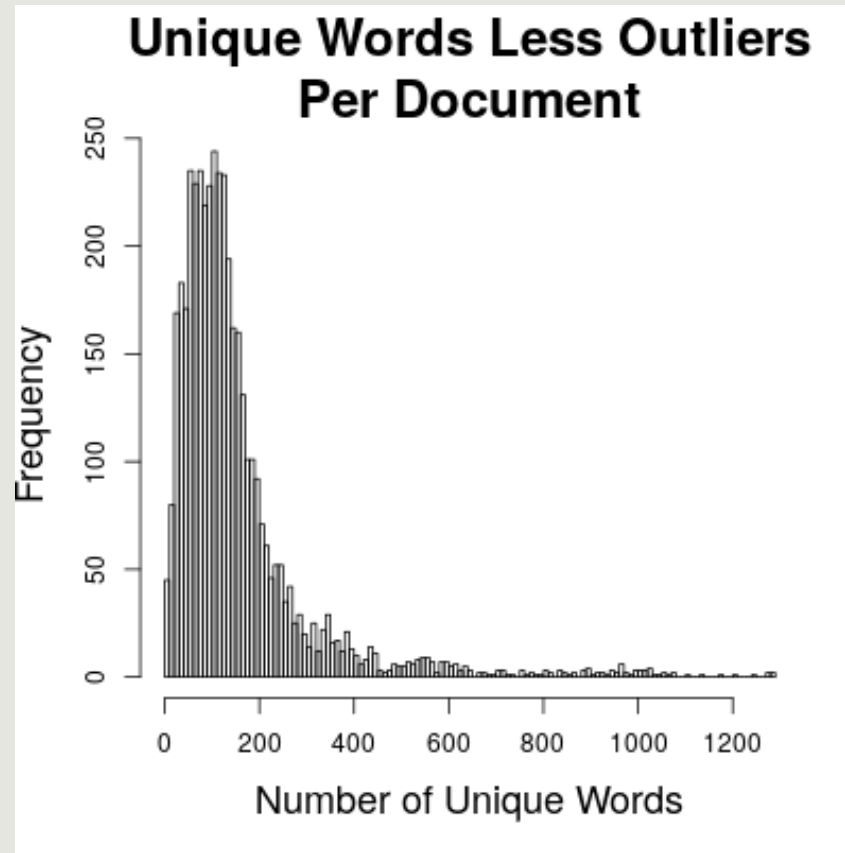
# Data

---



# Data with Outliers Removed

---



# Four Contrived Sample Documents

---

**==> examples/poem.eml <==**  
**From: noone@example.com**  
**Subject: A poem about foxes**

**The nimble brown and white fox jumped over the sleeping dog.**

**The brown and black dog awoke, looked up at the fox and said, "Woof!"**

# Four Contrived Sample Documents

---

==> examples/song.eml <==  
From: noone@example.com  
Subject: A song about foxes

What does the fox say?  
Ding ding ding da ding da ding ding.

# Four Contrived Sample Documents

---

==> examples/skit.eml <==

From: noone@example.com

Subject: A skit about spam

Man: You sit here, dear.

Wife: All right.

Man: Morning!

Waitress: Morning!

Man: Well, what've you got?

Waitress: Well, there's egg and bacon; egg sausage and bacon; egg and spam; egg bacon and spam; egg bacon sausage and spam; spam bacon sausage and spam; spam egg spam spam bacon and spam; spam sausage spam spam bacon spam tomato and spam;

Vikings: Spam spam spam spam...

Waitress: ...spam spam spam egg and spam; spam spam spam spam spam spam baked beans spam spam spam...

Vikings: Spam! Lovely spam! Lovely spam!

Waitress: ...or Lobster Thermidor a Crevette with a mornay sauce served in a Provencale manner with shallots and aubergines garnished with truffle pate, brandy and with a fried egg on top and spam.

Wife: Have you got anything without spam?

Waitress: Well, there's spam egg sausage and spam, that's not got much spam in it.

Wife: I don't want ANY spam!



# Four Contrived Sample Documents

---

Man: Why can't she have egg bacon spam and sausage?

Wife: THAT'S got spam in it!

Man: Hasn't got as much spam in it as spam egg sausage and spam, has it?

Vikings: Spam spam spam spam... (Crescendo through next few lines...)

Wife: Could you do the egg bacon spam and sausage without the spam then?

Waitress: Urrghh! Wife: What do you mean 'Urrghh'? I don't like spam!

Vikings: Lovely spam! Wonderful spam!

Waitress: Shut up! Vikings: Lovely spam! Wonderful spam!

Waitress: Shut up! (Vikings stop) Bloody Vikings! You can't have egg bacon spam and sausage without the spam.

Wife: I don't like spam!

Man: Sshh, dear, don't cause a fuss. I'll have your spam. I love it. I'm having spam spam spam spam spam spam spam beaked beans spam spam spam and spam!

Vikings: Spam spam spam spam. Lovely spam! Wonderful spam!

Waitress: Shut up!! Baked beans are off.

Man: Well could I have her spam instead of the baked beans then?

Waitress: You mean spam spam spam spam spam spam... (but it is too late and the Vikings drown her words)

Vikings: (Singing elaborately...) Spam spam spam spam. Lovely spam! Wonderful spam! Spam spa-a-a-a-am spam spa-a-a-a-am spam. Lovely spam! Lovely spam! Lovely spam!

# Four Contrived Sample Documents

---

==> examples/spam.eml <==  
From: noone@example.com  
Subject: A word about Spam

Spam

# Term Vectors

---

==>> examples/poem.term <<==

3 and

2 the

2 fox

2 dog

2 brown

2 The

1 white

1 up

1 sleeping

1 said

1 over

1 nimble

1 looked

1 jumped

1 black

1 awoke

1 at

1 Woof

# Term Vectors

---

==>> examples/song.term <<==

5 ding

2 da

1 the

1 say

1 fox

1 does

1 What

1 Ding

# Term Vectors

---

`==> spam.term <==`

1 Spam

# Term Vectors

---

==> skit.term <==

95 spam

19 and

13 egg

13 a

11 Lovely

10 bacon

10 Waitress

10 Vikings

9 sausage

7 t

7 Wife

7 Spam

7 Man

7 I

6 it

5 the

5 got

4 you

4 with

4 in

4 have

4 don

4 beans

4 Wonderful

4 Well

3 without

3 up

3 s

3 You

3 Shut

2 there . . .

# Term-Frequency/Inverse Document Frequency

---

$$tf-idf(t, d, D) = tf(t, d) * \log\left(\frac{M}{df(t, D)}\right)$$

where

$tf(t, d)$  is the number of times term  $t$  occurs in document  $d$

$df(t, D)$  is the number of documents term  $t$  occurs in across the entire corpus  $D$

$M$  is the number of documents.

# Document Frequency

---

==> examples/document\_frequency <==

3 the

2 up

2 fox

2 and

2 What

2 Spam

1 your

1 you

1 words

1 without

1 with

1 white

1 what

1 want

1 ve

1 truffle

1 top

1 too

1 tomato

1 through . . .



# Term Vectors vs TF-IDF Vectors

---

==>> examples/poem.term <<==

3 and  
2 the  
2 fox  
2 dog  
2 brown  
2 The  
1 white  
1 up  
1 sleeping  
1 said  
1 over  
1 nimble  
1 looked  
1 jumped  
1 black  
1 awoke  
1 at  
1 Woof

==>> examples/poem.tfidf <<==

2.772589 dog  
2.772589 brown  
2.772589 The  
2.079442 and  
1.386294 white  
1.386294 sleeping  
1.386294 said  
1.386294 over  
1.386294 nimble  
1.386294 looked  
1.386294 jumped  
1.386294 fox  
1.386294 black  
1.386294 awoke  
1.386294 at  
1.386294 Woof  
0.693147 up  
0.575364 the

# Term Vectors vs TF-IDF Vectors

---

==>> examples/song.term <<==

5 ding

2 da

1 the

1 say

1 fox

1 does

1 What

1 Ding

==>> examples/song.tfidf <<==

6.931472 ding

2.772589 da

1.386294 say

1.386294 does

1.386294 Ding

0.693147 fox

0.693147 What

0.287682 the

# Term Vectors vs TF-IDF Vectors

---

==>> examples/skit.term <<==

95 spam	4 have
19 and	4 don
13 egg	4 beans
13 a	4 Wonderful
11 Lovely	4 Well
10 bacon	3 without
10 Waitress	3 up
10 Vikings	3 s
9 sausage	3 You
7 t	3 Shut
7 Wife	2 there
7 Spam	2 then
7 Man	2 spa
7 I	2 much
6 it	2 mean
5 the	2 like
5 got	2 her
4 you	2 do
4 with	2 dear
4 in	2 can

==>> examples/skit.tfidf <<==

131.697964 spam	5.545177 beans
18.021827 egg	5.545177 Wonderful
18.021827 a	5.545177 Well
15.249238 Lovely	4.852030 Spam
13.862944 bacon	4.158883 without
13.862944 Waitress	4.158883 s
13.862944 Vikings	4.158883 You
13.169796 and	4.158883 Shut
12.476649 sausage	2.772589 there
9.704061 t	2.772589 then
9.704061 Wife	2.772589 spa
9.704061 Man	2.772589 much
9.704061 I	2.772589 mean
8.317766 it	2.772589 like
6.931472 got	2.772589 her
5.545177 you	2.772589 do
5.545177 with	2.772589 dear
5.545177 in	2.772589 can
5.545177 have	2.772589 baked
5.545177 don	2.772589 as

# Term Vectors vs TF-IDF Vectors

---

`==>> examples/spam.term <<==`  
1 Spam

`==>> examples/spam.tfidf <<==`  
0.693147 Spam

# Cosine Similarity

---

$$\cos(x, y) = \frac{x \cdot y}{|x| |y|}$$

Between vectors x and y

# Cosine Similarity

---

$$\cos(\text{song}, \text{poem}) = \frac{2 \times 1 + 2 \times 1}{\sqrt{41} \sqrt{35}} = 0.105593$$

==>> examples/poem.term <<==

3 and  
2 the  
2 fox  
2 dog  
2 brown  
2 The  
1 white  
1 up  
1 sleeping  
1 said  
1 over  
1 nimble  
1 looked  
1 jumped  
1 black  
1 awoke  
1 at  
1 Woof

==>> examples/song.term <<==

5 ding  
2 da  
1 the  
1 say  
1 fox  
1 does  
1 What  
1 Ding

# Document Similarity

---

$$s_h = \cos(v_{tfidf}, e_h) = \text{"hamminess"}$$

$$s_s = \cos(v_{tfidf}, e_s) = \text{"spamminess"}$$

# Threshold

---

$$s_h / s_s \geq t \Rightarrow \text{"ham"}$$

Where

$$s_h = \cos(v_{tfidf}, e_h) \quad = \text{"hamminess"}$$

$$s_s = \cos(v_{tfidf}, e_s) \quad = \text{"spamminess"}$$



# Scenarios

---

## **Document Frequency**

tf-idf vectors compared to the document frequency vector

## **Average Term Frequency**

tf-idf vectors compared average of term vectors

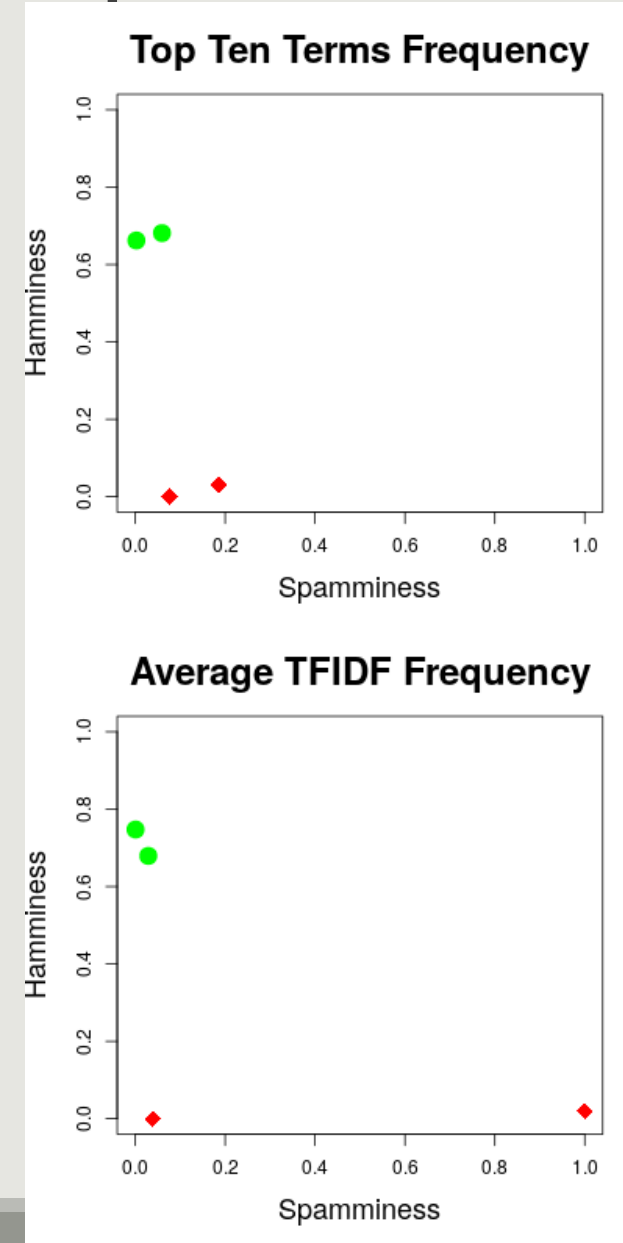
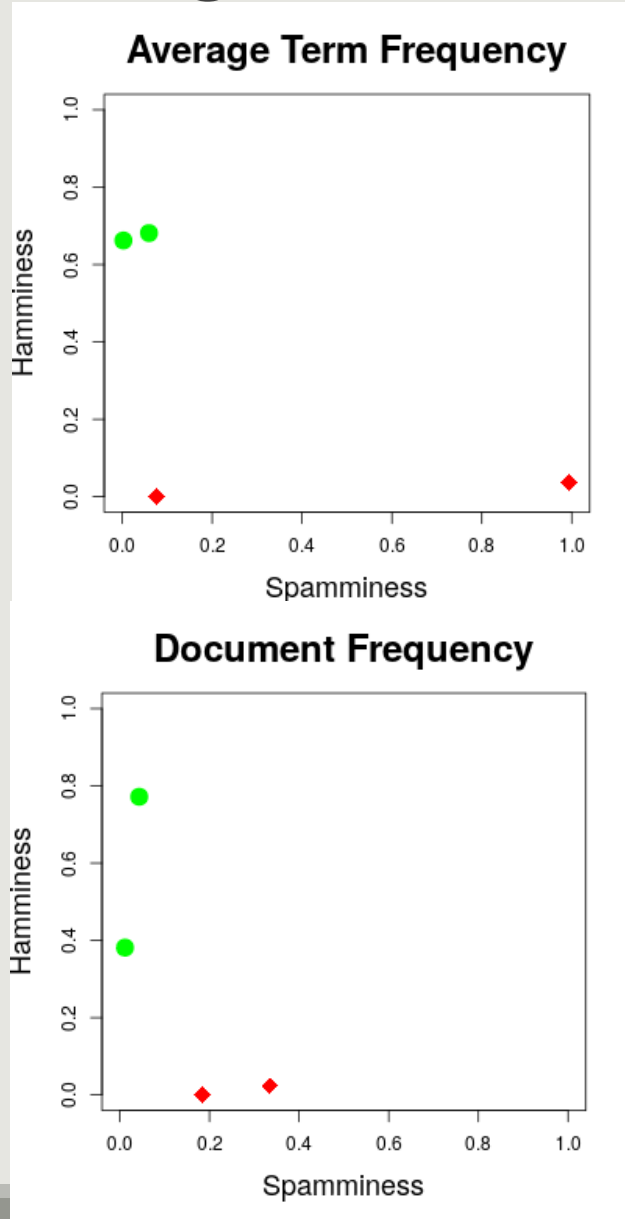
## **Top-10**

tf-idf vectors truncated to the top-ten scoring words in the document

## **Average TF-IDF**

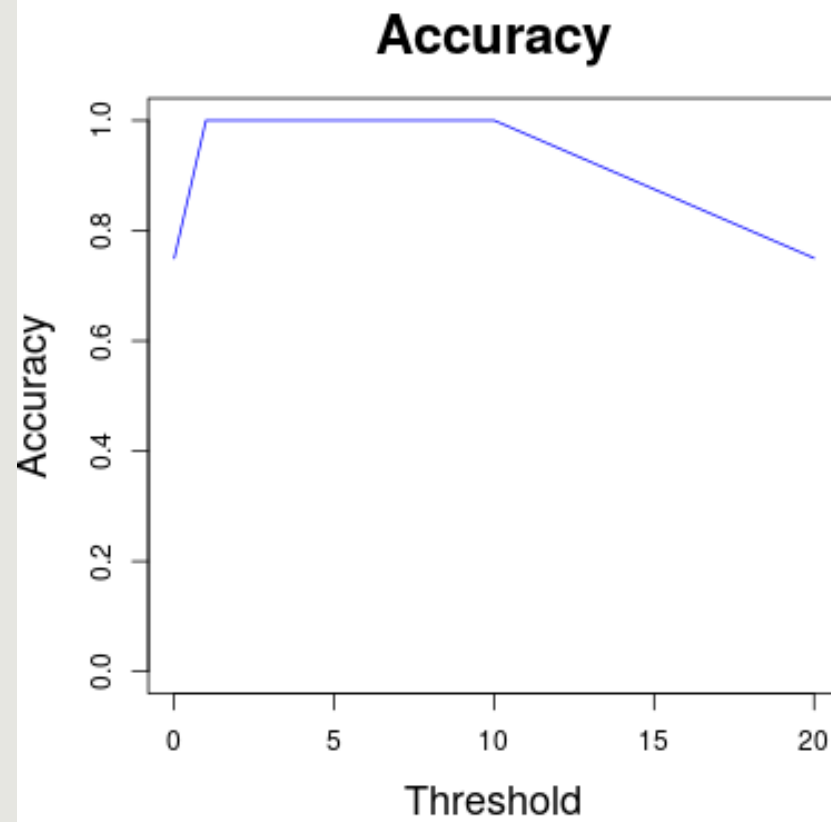
tf-idf vectors compared to the average of tf-idf vectors

# Discriminating Between Ham & Spam



# Accuracy

---



# Accuracy

$$A(t) = \frac{m_{00}(t) + m_{11}(t)}{m_{00}(t) + m_{01}(t) + m_{10}(t) + m_{11}(t)}$$

where  
 $m_{ij}(t)$  is the number of messages actually in class  $i$  and identified as class  $j$ .

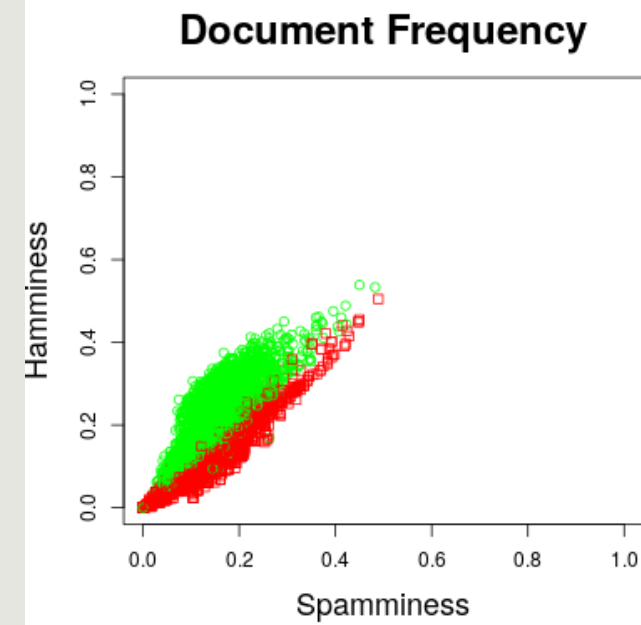
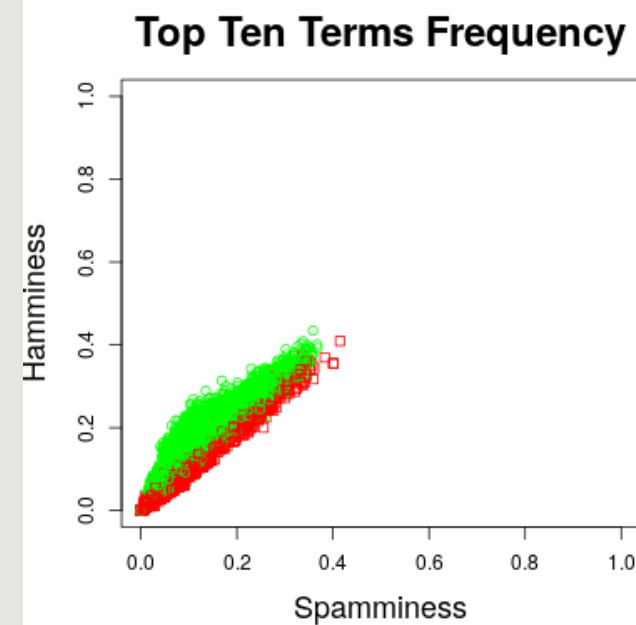
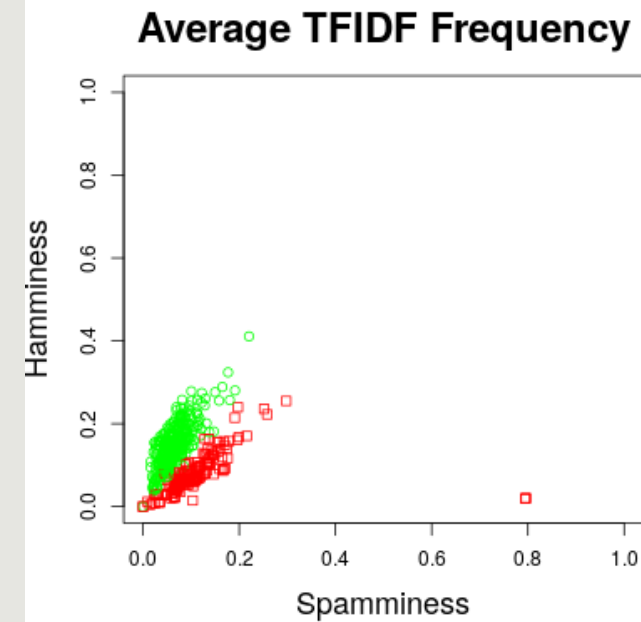
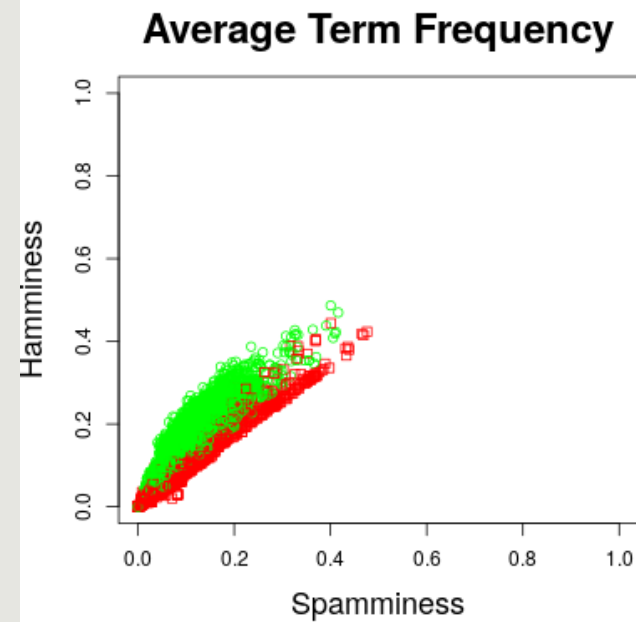
		Predicted Class	
		Spam	Ham
Actual Class	Spam	$m_{00}(t)$	$m_{01}(t)$
	Ham	$m_{10}(t)$	$m_{11}(t)$

## Note

$i=j$  implies correct classification

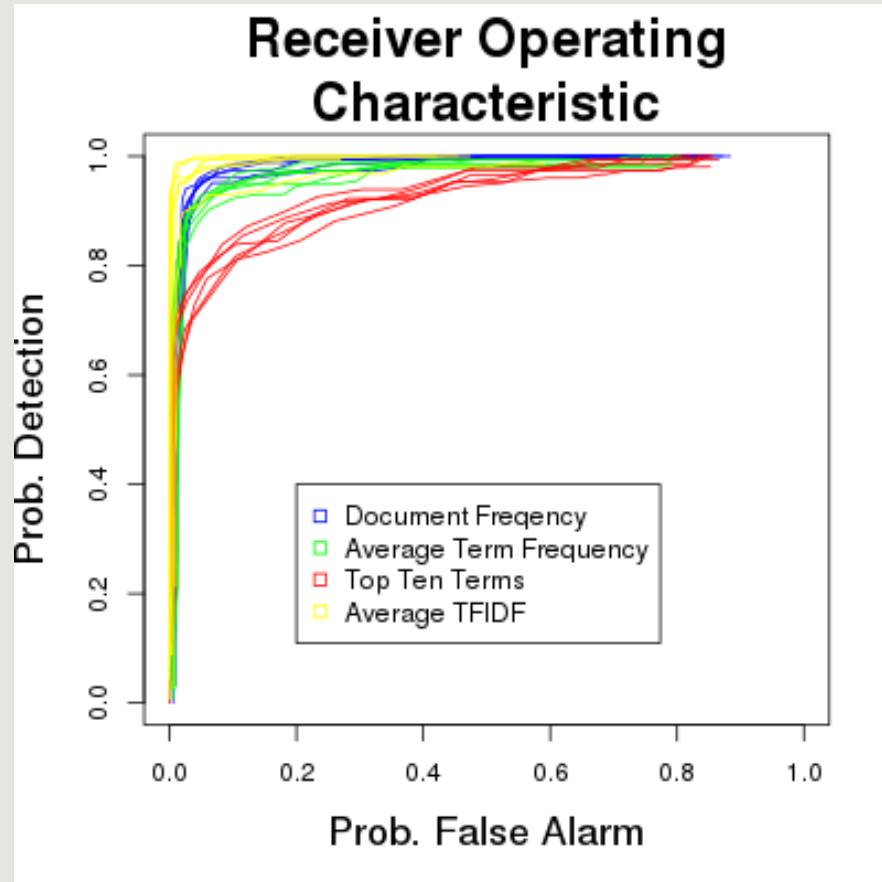
$i \neq j$  implies incorrect classification

# Results



# Receiver Operating Characteristics

---



# Receiver Operating Characteristics

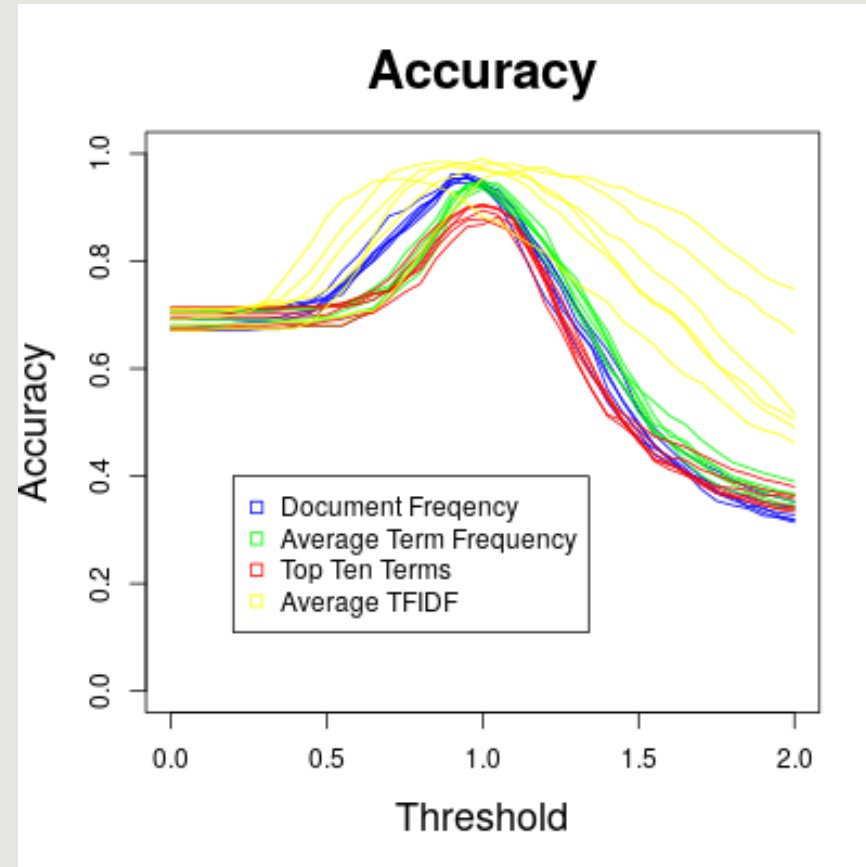
---

$$P_D(t) = \frac{m_{00}(t)}{m_{00}(t) + m_{01}(t)}$$

$$P_{FA}(t) = \frac{m_{10}(t)}{m_{10}(t) + m_{11}(t)}$$

# Accuracy

---





# Amateurs vs Pros

---

How does a commercial spam filter stack up using the same dataset?

- Postfix Server
- Dovecot as mail delivery agent
- Spam Assassin as the filter

Accuracy: 93.8%

Average Maximum Accuracy:

Document Similarity – 95.53%

Average Term Similarity – 94.23%

Top-Ten Term Similarity – 89.2%

Average TFIDF Similarity – 95.8%

