



articles Q&A forums lounge

Search for articles, questions, tips




Naive Bayes Theorem



saharkiz, 2 Mar 2009

★★★★★ 4.73 (18 votes) Rate:

Anti Spam Filter using Naive Bayes Theorem



Is your email address OK? You are signed up for our newsletters but your email address is either unconfirmed, or has not been reconfirmed in a long time. Please **click here to have a confirmation email sent** so we can confirm your email address and start sending you newsletters again. Alternatively, you can **update your subscriptions**.

Download tutorial - 3.64 MB
Download source - 3.64 KB

expiration viagra

Verify

Analysis

MailType	Total Count	P(MailType)
NonSpam	40080293	0.25317654492356
Spam	118229368	0.74682345507644
	158309661	100%

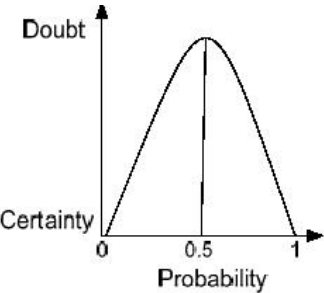
Probability

NonSpam Percent	1.63118058324232E-11
Spam Percent	1.91218061618023E-11

Spam Mail

Introduction

Probability is defined as a quantitative measure of uncertainty state of information or event. It has an index which ranges from 0 to 1. It is also approximated through proportion of number of events over the total experiment. If the probability of a state is 0 (zero), we are certain the state will not happen. However if the probability is 1, the event will surely happen. A probability of 0.5 means we have maximum doubt about the state that will happen.



The following section describes some of the basic probability formulas that will be used:

Conditional probability: The probability of an event may depend on the occurrence or non-occurrence of another event. This dependency is written in terms of conditional probability:

$P(A|B)$

"the probability that A will happen given that B already has" or "the probability to select A among B"

Notice that B is given first, and we find the proportion of A among B:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$$P(B|A) = P(A \cap B) / P(A)$$

$$P(A|B) = P(A \cap B) / P(B)$$

$$P(A \cap B) = P(B|A) P(A) = P(A|B) P(B)$$

Given the above formula: An event A is INDEPENDENT from event B if the conditional probability is the same as the marginal probability.

Hide Copy Code

$P(B|A) = P(B)$

Hide Copy Code

$P(A|B) = P(A)$

From the formulas the Bayes Theorem States the Prior probability: Unconditional probabilities of our hypothesis before we get any data or any NEW evidence. Simply speaking, it is the state of our knowledge before the data is observed.

Also stated is the posterior probability: A conditional probability about our hypothesis (our state of knowledge) after we revised based on the new data.

Likelihood is the conditional probability based on our observation data given that our hypothesis holds.

Bayes Theorem:

The diagram shows two versions of Bayes' Theorem. The first is $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ and the second is $P(B|A) = \frac{P(A|B)P(B)}{P(A)}$. In both, $P(A|B)$ and $P(B|A)$ are boxed in blue and labeled 'Posterior probability'. $P(B|A)$ and $P(A|B)$ are boxed in green and labeled 'Likelihood'. $P(A)$ and $P(B)$ are circled in red and labeled 'Prior probability'.

Thomas Bayes (c. 1702 – 17 April 1761) was a British mathematician and Presbyterian minister, known for having formulated a specific case of the theorem that bears his name: Bayes' theorem, which was published posthumously.

The following are the mathematical formalisms, and the example on a spam filter, but keep in mind the basic idea.

The Bayesian classifier uses the Bayes theorem, which says:

$$p(c_j | d) = \frac{p(d | c_j) p(c_j)}{p(d)}$$

Considering each attribute and class label as a random variable and given a record with attributes (A_1, A_2, \dots, A_n) , the goal is to predict class C. Specifically, we want to find the value of C that maximizes $P(C | A_1, A_2, \dots, A_n)$.

The approach taken is to compute the posterior probability $P(C | A_1, A_2, \dots, A_n)$ for all values of C using the Bayes theorem.

$$P(C | A_1 A_2 \dots A_n) = \frac{P(A_1 A_2 \dots A_n | C) P(C)}{P(A_1 A_2 \dots A_n)}$$

So you choose the value of C that maximizes $P(C | A_1, A_2, \dots, A_n)$. This is equivalent to choosing the value of C that maximizes $P(A_1, A_2, \dots, A_n | C) P(C)$.

To simplify the task of Naïve Bayesian Classifiers, we assume attributes have independent distributions.

The Naïve Bayes theorem has the following characteristics as advantages and disadvantages:

Advantages:

1. Handles quantitative and discrete data
2. Robust to isolated noise points
3. Handles missing values by ignoring the instance
4. During probability estimate calculations
5. Fast and space efficient
6. Not sensitive to irrelevant features
7. Quadratic decision boundary

Disadvantages:

- If conditional probability is zero
- Assumes independence of features

Naïve Bayesian prediction requires each conditional probability be non zero. Otherwise, the predicted probability will be zero.

$$P(X | C_i) = \prod_{k=1}^n P(x_k | C_i)$$

In order to overcome this, we use probability estimation from one of the following:

Original : $P(A_i | C) = \frac{N_{ic}}{N_c}$

Laplace : $P(A_i | C) = \frac{N_{ic} + 1}{N_c + c}$

m - estimate : $P(A_i | C) = \frac{N_{ic} + mp}{N_c + m}$

c: number of classes
p: prior probability
m: parameter

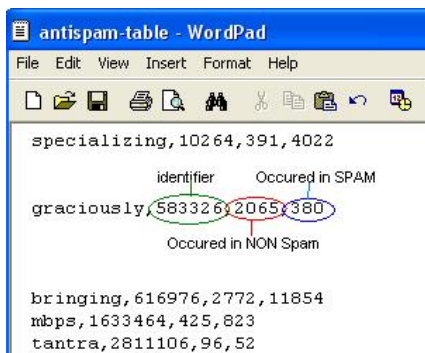
The explanation of these is out of the scope of this tutorial.

Program

In order to classify and predict a spam email from a non spam one, I will be using the following techniques and assumptions:

1. I'll be sorting according to language (spam or non spam), then words, then count
2. If a word does not exist, consider to approximate P(word|class) using Laplacian
3. I'll be using the following table for my analysis

The *antispam-table.txt* file is the file that contains each word that content filtering uses to determine if a message is spam. Beside each word, there are three numbers. The first number is an identifier assigned by the anti-spam engine. The second number is the number of times that the word has occurred in non-spam e-mail messages. The third number is the number of times that the word has occurred in spam e-mail messages.



<http://support.ipswitch.com/kb/IM-20030513-DM01.htm>

Anti-spam table for Statistical Filtering and a new phrase list for Phrase Filtering.

I've create a database from the CSV file. Please see the [tutorial](#). I've also included the Excel computation as a reference. The code goes as follows:

1. Remove or replace any known or unknown special characters so only the words can be used for the probability computation. It's best to stick with lower case or convert everything to lower case. (I've not done that.)

[Hide](#) [Copy Code](#)

```
String inputContent = TextBox1.Text;
inputContent = inputContent.Replace("\t", "");
inputContent = inputContent.Replace("\n", "");
inputContent = inputContent.Replace(", ", "");
inputContent = inputContent.Replace(".", "");
inputContent = inputContent.Replace(";", "");
inputContent = inputContent.Replace(":", "");
inputContent = inputContent.Replace("?", "");
inputContent = inputContent.Replace("!", "");
inputContent = inputContent.Replace("&", "");
```

2. Separate each word and find the corresponding value from the database. the value shows how many times the word has been found in a spam or non spam mail. If the word is not found, use laplace or just make it equal to 1 (reason stated above).

[Hide](#) [Shrink](#) [Copy Code](#)

```
char seperator = ' ';
String[] words = inputContent.Split(seperator);
int CountWords = words.Length;
//keep P(score) and multiply
double SpamPercent = 1.0;
double NonSpamPercent = 1.0;

//look for percentage of word in nonspam
for (int i = 0; i < CountWords; i++)
{
    String thisword = words[i];
    String DBaseValue = (BayesTheorem.FindNonSpamWord(thisword));
    if (DBaseValue == "")
    {
        //Perform Laplace or just equal to 1
        DBaseValue = "1";
    }
    NonSpamPercent = NonSpamPercent * (Convert.ToDouble(DBaseValue) /
        Convert.ToInt32(BayesTheorem.TotalNonSpam()));
}

//Look for percentage of word in spam
for (int i = 0; i < CountWords; i++)
{
    String thisword = words[i];
    String DBaseValue = (BayesTheorem.FindSpamWord(thisword));
    if (DBaseValue == "")
    {
        //Perform Laplacin or just equal to 1
        DBaseValue = "1";
    }
    SpamPercent = SpamPercent * (Convert.ToDouble(DBaseValue) /
        Convert.ToInt32(BayesTheorem.TotalSpam()));
}
```

3. Get the final probability by multiplying each word to the total probability:

[Hide](#) [Copy Code](#)

```
//get P(mailType).P(Word | MailType) for Spam
SpamPercent = SpamPercent * Convert.ToDouble(Label4.Text) * 100;
//get P(mailType).P(Word | MailType) for NonSpam
NonSpamPercent = NonSpamPercent * Convert.ToDouble(Label3.Text) * 100;
```

4. Show the results in a table:

[Hide](#) [Copy Code](#)

```
//Show results in table
Label16.Text = NonSpamPercent.ToString();
Label17.Text = SpamPercent.ToString();
```

Compare the results to one another and determine spam or nonspam based on the highest probability.

[Hide](#) [Copy Code](#)

```
if (SpamPercent > NonSpamPercent)
    Label18.Text = "Spam Mail";
else Label18.Text = "NonSpam Mail";
```

5. Finally if you have determined an email to be spam, update the values in the database for each word. Same thing as with nonSpam. If the value does not exist in the database, just insert for future training. The more training, the better the classification.

[Hide](#) [Copy Code](#)

```
//insert words or add new word and count for future training
//TO DO: .....
```

History

- February 27, 2009: Draft of tutorial without source code
- March 1, 2009: Program download and output screenshot

License

This article, along with any associated source code and files, is licensed under [The Code Project Open License \(CPOL\)](#)

Share

[TWITTER](#)
[FACEBOOK](#)

About the Author



saharkiz
Web Developer
Philippines

My name : Aresh Saharkhiz.
Origin: Unknown

Education : BS Computer Science
MS Computer Science
Interests : Genetic Programming
Neural Networks
Game AI
Programming: (language == statementEnd(semicolon))

<http://sites.google.com/site/docaresh>

Skill:
Flash
Carrara 3D
PHP,ASP,ASP.NET
J2SE

You may also be interested in...

- [Naive Bayes Classifier](#)
- [Window Tabs \(WndTabs\) Add-In for DevStudio](#)
- [A spelling corrector based on Bayes Theorem \(PHP, C#\)](#)
- [OLE DB - First steps](#)
- [SAPrefs - Netscape-like Preferences Dialog](#)
- [Introduction to D3DImage](#)

Comments and Discussions

Add a Comment or Question

Search Comments

First Prev Next

CODE run

Member 11163981 7-Mar-16 1:54

Re: CODE run

saharkiz 21-Jun-16 16:09

Useful project

kinhkong3004 26-Mar-14 17:41

Re: Useful project

Member 12928685 31-Dec-16 1:48

navie baysian classifier

asma_cool14 17-Jun-13 20:32

My vote of 5

Filip D'haene 26-May-11 18:38

Good Detail! Vote 5

TheArchitect_{mc}[∞] 25-Mar-10 19:19

Good but....

Diego Galeano 10-Mar-09 21:11

Re: Good but....

saharkiz 11-Mar-09 16:13

Re: Good but....

Diego Galeano 17-Mar-09 0:55

