# 4th work assignment

Create a classifier of SPAM using Bayes theorem. Investigate a dependency between classifier parameters and efficiency of the classifier output – dependency of number *false positive* and *true negative* number on   
i) number *N* of analyzed lexemes, ii) spamicity value of unseen lexeme,.

Recommendations

Use hash tables for lexeme recordings. Lexeme could be a string containing symbols of digits, alphabet, and (! @ $ &).

# Work description (one of the possible)

1. Let us we have 2 folders with SPAM and HAM. Files in those folders are training data. Let overall number of lexemes of files in SPAM folder be 300, and number of files lexemes in HAM folder files is 250.
2. A data structure for recording of lexemes should be created (e.g. *hash* table). It should contain numbers of occurrences of lexemes in SPAM and HAM folders respectively, e.g. lexeme „m0ney“ has been encountered 50 times in SPAM folder files and 15 times in HAM folder files, while lexeme „earn“ – 200 and 33 times respectively.

|  |  |  |
| --- | --- | --- |
|  | SPAM | HAM |
| m0ney | 50 | 15 |
| earn | 200 | 33 |

1. A spamicity probability is calculated for all lexemes, i.e. a probability that file containing that lexeme is SPAM. The following formula is used:



where P(W|S) is a probability that lexeme W belongs to SPAM, P(W|H) is a probability that lexeme W is in HAM.

In our case:

|  |  |  |  |
| --- | --- | --- | --- |
| W | P(W|S) | P(W|H) | P(S|W) |
| m0ney |  |  |  |
| earn |  |  |  |

It is advisable to store P(S|W) spamicity probabilities of training data in external file rather than calculate them each time when you need to calculate spamicity of new file.

1. New file presented for classification should be split into lexemes. A spamicity of each lexeme is calculated (see previous instruction). Spamicity of a new lexeme is set equal to 0,4.  
   Let us we have new file: Earn m0ney zzz.
2. Only certain number *N* of lexemes is chosen from an analyzed file (e.g. 16), which spamicity probabilities are as far as possible from a mean value of all probabilities of an analyzed file.

*l*1  *l*2  *l*3 ... *l*M -lexemes of analyzed file

0.1 0.2 ... 0.99 –spamicity probabilities

N/2 maximally remoted lexemes N/2 maximally remoted lexemes  
from mean value from mean value

... 0.99 –spamicity probabilities

1. A probability of spamicity of the analyzed file with respect to the chosen lexemes is evaluated. The following formula could be used

,

where *pi* – a spamicity probability of a chosen lexeme. For our example we choose 2 lexemes – „m0ney“ and „Earn“:



**Note:** if we have the following situation

|  |  |  |
| --- | --- | --- |
|  | SPAM | HAM |
| top | 11 | 0 |
| bottom | 0 | 44 |

Then a spamicity of those lexemes is evaluated in the following way

|  |  |  |  |
| --- | --- | --- | --- |
| W | P(W|S) | P(W|H) | P(S|W) |
| top |  |  | 0.99, if P(W|H)=0 |
| bottom |  |  | 0.01, if P(W|S)=0 |

1. A threshold of spamicity value should be defined in order to distinguish SPAM and HAM files based on training data. When testing a performance of the developed SPAM analyzer the following metrics should be evaluated:
   1. Number of SPAM files classified as HAM (*false positive*);
   2. Number of HAM files classified as SPAM (*true negative*);
   3. Ratio (%) between correctly classified files and overall number of files.

# Report should contain:

1. Title page, work assignment, program code (use single space interval, 6 pt Courier New)
2. Metrics values as required in #7
3. Dependency of number *false positive* and *true negative* number (separate charts) on
4. number *N* of analyzed lexemes (e.g. 8, 16, 32) ,
5. spamicity value of unseen lexeme,

**Defense deadline: End of semester**

# Additional readings:

1. <http://www.paulgraham.com/spam.html>
2. <http://en.wikipedia.org/wiki/Bayesian_spam_filtering>