TotalCostRatio

Total Cost Ratio

A scheme for MeasuringAccuracy developed by Ion Androutsopoulos et al, and described in *An Evaluation of Naive Bayesian Anti-Spam Filtering* (Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras and Constantine D. Spyropoulos, published in the Proceedings of the workshop on Machine Learning in the New Information Age, G. Potamias, V. Moustakis and M. van Someren (eds.), 11th European Conference on Machine Learning, Barcelona, Spain, pp. 9-17, 2000.)

Quoting that paper:

To compare easily with the baseline, we introduce the total cost ratio (TCR): [...]
Greater TCR indicates better performance. For TCR < 1, not using the filter
is better. If cost is proportional to wasted time, TCR measures how much time
is wasted to delete manually all spam messages when no filter is present,
compared to the time wasted to delete manually any spam messages that passed
the filter plus the time needed to recover from mistakenly blocked legitimate
messages.

TCR uses weighted costs – it introduces a lambda value, indicating how much more important (costly) a non-spam mail is, when compared to a spam mail, based on how much effort a user would have to expend to recover from a misclassification.
The lambda value is intended to be set to a value based on how the filter deals with classified messages; for example, if the filter simply displays spam messages in a different colour in the same inbox, the lambda is 1; if the filter blocks the message and asks the sender to re-send, the lambda is 9; if the filter deletes the message silently, the lambda is 999.

We used to use a lambda of 5 in SpamAssassin, to represent its typical use, where tagged messages are simply moved to a separate folder in the user's mail client, where the user can easily recover from a misclassification. More recently, we've been using a lambda of 50, because SpamAssassin is now used in more aggressive use cases at some sites.

The TCR is computed as follows. First, take the usual set of 4 inputs:

```
nspam   = number of known-to-be-spam messages in the corpus
nham    = number of known-to-be-ham (nonspam) messages in the corpus
fp      = number of ham messages incorrectly marked as spam
fn      = number of spam messages incorrectly marked as ham
```

Now, this perl code will generate the TCR:

```
$werr = ($lambda * $fp + $fn) / ($lambda * $nham + $nspam);
$werr ||= 0.000001;     # avoid / by 0

$werr_base = $nspam / ($lambda * $nham + $nspam);

$tcr = $werr_base / $werr;
```

The 'STATISTICS.txt' files distributed with SpamAssassin versions since about 2.30 include this data, measuring the ruleset's accuracy against a validation ruleset:

```
# SUMMARY for threshold 5.0:
# Correctly non-spam:  29443  99.97%
# Correctly spam:      27220  97.53%
# False positives:         9  0.03%
# False negatives:       688  2.47%
# TCR(l=50): 24.523726  SpamRecall: 97.535%  SpamPrec: 99.967%
```

TCR has a major benefit, in that it's a single-figure measurement, whereas most others require at least two figures. However, it can be misleading, since a good TCR may represent either a low false-positive rate, or an extremely high hit-rate with a moderately-high FP rate, and you cannot tell from the TCR alone which it is.

Also, TCR seems to be sensitive to the balancing of the corpora – if the amounts of nonspam and spam in the corpora differ, TCR figures are not portable.

Finally, TCR figures can only be compared when all compared TCRs were computed using the same lambda.

See also MeasuringAccuracy for other methods.