Precision and recall

John Goldsmith

June 26, 2015
Document retrieval

<table>
<thead>
<tr>
<th>Precision</th>
<th>How well do the documents that your system gives you actually satisfy what you are looking for?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>How sure are you that you got back all of the documents you really wanted?</td>
</tr>
</tbody>
</table>
Document retrieval

**Precision**

\[
\frac{\#(\text{appropriate documents returned})}{\#(\text{documents returned})}
\]

**Recall**

\[
\frac{\#(\text{appropriate documents returned})}{\#(\text{appropriate documents})}
\]
These terms have become the standard expectation of how a method is evaluated.

Precision and recall trade-off
Precision and recall trade-off

You can always get 100% precision, and you can always get 100% recall, but the cost is almost always too great, in both cases.

One solution: use the F-score: the reciprocal of the average of the reciprocals. $2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$.

$$\frac{1}{\frac{1}{2}\left(\frac{1}{a} + \frac{1}{b}\right)} = \frac{1}{\frac{1}{2}\left(\frac{a+b}{ab}\right)} = \frac{2ab}{a + b}$$
Precision and recall trade-off

Or you can give a chart of various precision/recall trade-offs produced by adjusting parameters of the algorithm.
### Precision and recall

<table>
<thead>
<tr>
<th>Gold standard:</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test says yes:</td>
<td>True positives</td>
<td>False positives</td>
</tr>
<tr>
<td>Test says no:</td>
<td>False negatives</td>
<td>True negatives</td>
</tr>
</tbody>
</table>

**Precision**

**Recall**
More than one possible test: 1

Task: Find morphemes

Your algorithm wants to find morphemes (=word parts): anti-alias-ing

Measurement: find breaks

One way to measure this is by predicting which positions mark breaks: Gold standard truth is 0,4,9,14. Then antialias-ing is 0,9,14. Precision is $\frac{3}{4}$ and recall is $\frac{3}{4}$. 
What is the precision and recall of a clever but useless algorithm: e.g., mark morphemes boundaries before the first and after the last letter?

A clever but useless algorithm defines our baseline. Hopefully we have nowhere to go than up from there (though that is not guaranteed!).
Possible test 2:

Discover a list of morphemes

Suppose our goal is to “pullout” the morphemes of the language. Then if *ed* or *ing* is found in *any* word, that counts as 1 true positive.

If the algorithm cuts: *jump-ed walk-ed mov-e-d lov-ed raise-d* and the gold standard says *jump walk move love raise ed*, then there are 4 true positives (*jump, walk, raise, ed*) and 2 false negatives (*move, love*) (because they were *not found* by the algorithm), and 3 false positives (*e,d, lov*) (because they were found but they should not have been found).

Precision: 4 out of $(4 + 3) = 0.571$; recall is 4 out of $(4 + 2) = 0.667$. 