The document ranking problem

- We have a collection of documents
- User issues a query
- A list of documents needs to be returned
- Ranking method is core of an IR system:
 - In what order do we present documents to the user?
 - We want the "best" document to be first, second best second, etc....
- Idea: Rank by probability of relevance of the document w.r.t. information need
 - P(relevant|document_i, query)

Recall a few probability basics

Bayes' Rule: For events *a* and *b*, $p(a,b) = p(a \cap b) = p(a | b) p(b) = p(b | a) p(a)$ $p(\overline{a} | b) p(b) = p(b | \overline{a}) p(\overline{a})$ $p(a | b) = \frac{p(b | a) p(a)}{p(b)} = \frac{p(b | a) p(a)}{\sum_{x=a,\overline{a}} p(b | x) p(x)} \xrightarrow{\text{Prior}}$ Posterior Odds: $O(a) = \frac{p(a)}{p(\overline{a})} = \frac{p(a)}{1-p(a)}$

The Probability Ranking Principle

"If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data."

> [1960s/1970s] S. Robertson, W.S. Cooper, M.E. Maron; van Rijsbergen (1979:113); Manning & Schütze (1999:538)

Probability Ranking Principle

Let x be a document in the collection. Let R represent **relevance** of a document w.r.t. given (fixed) query and let NR represent **non-relevance**. **R={0,1} vs. NR/R**

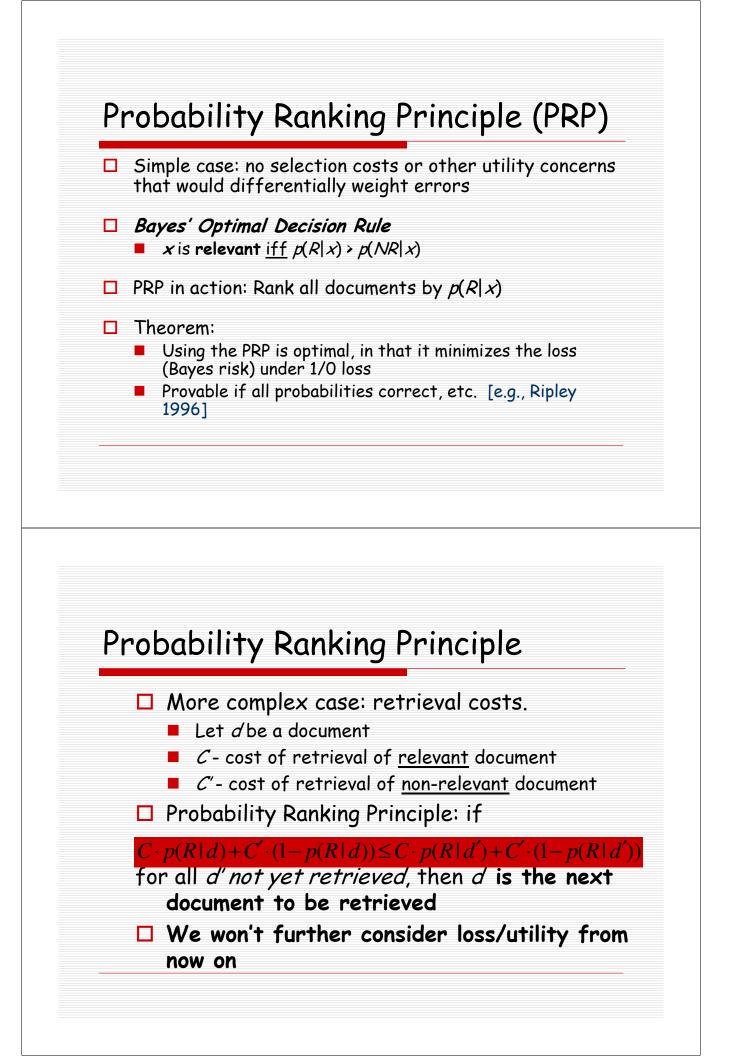
Need to find p(R/x) - probability that a document x is **relevant**. $p(R \mid x) = \frac{p(x \mid R) p(R)}{p(x)}$ p(R),p(NR) - prior probabilityof retrieving a (non) relevant

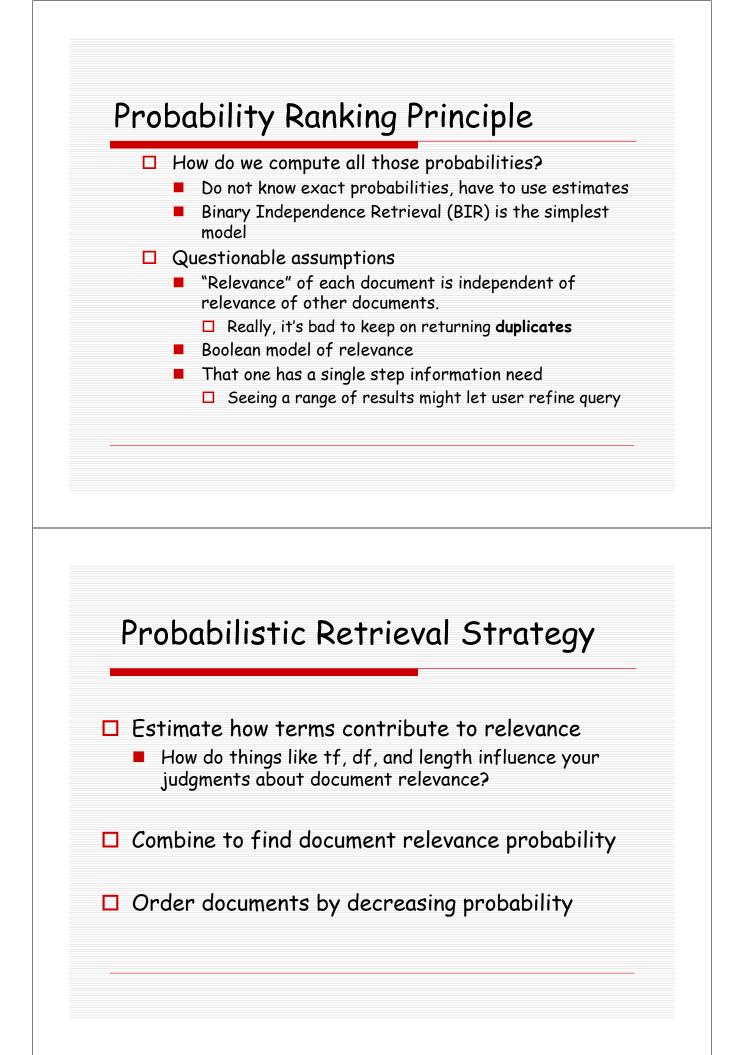
document

 $p(NR \mid x) = \frac{p(x \mid NR)p(NR)}{p(x)}$

 $p(R \mid x) + p(NR \mid x) = 1$

p(x|R), p(x|NR) - probability that if a relevant (non-relevant) document is retrieved, it is x.





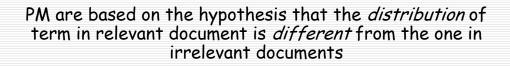
Probabilistic Ranking

Basic concept:

"For a given query, if we know some documents that are relevant, terms that occur in those documents should be given greater weighting in searching for other relevant documents.

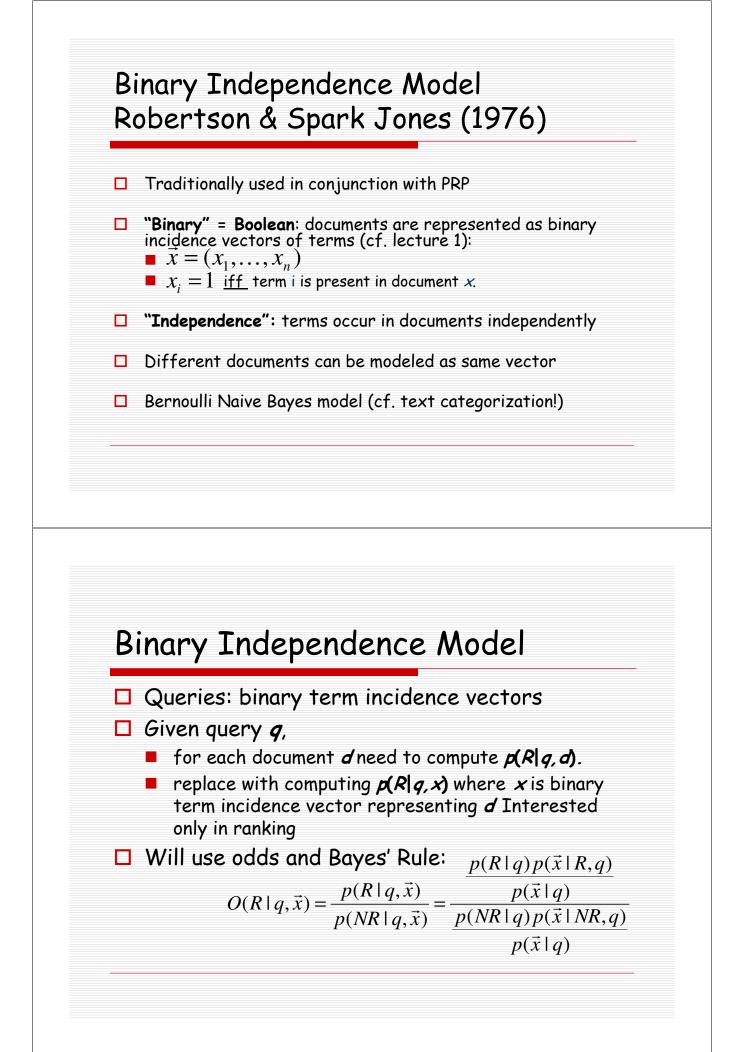
By making assumptions about the distribution of terms and applying Bayes Theorem, it is possible to derive weights theoretically."

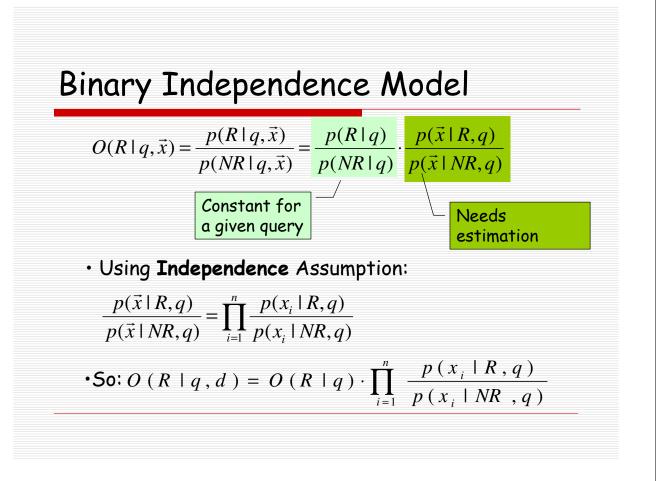
Van Rijsbergen



Then,

- A greater importance should be given to terms that occur in many relevant documents and are absent in many irrelevant documents
- A smaller importance should be given to terms that occur in many irrelevant documents and are absent in many relevant documents



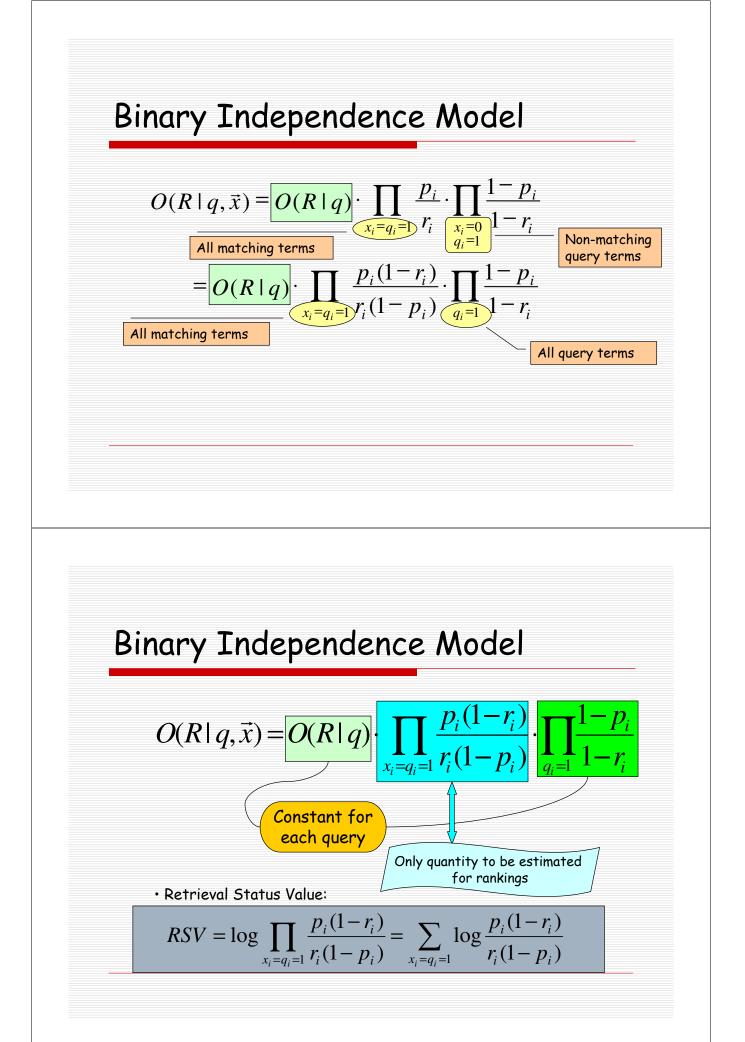




$$O(R | q, d) = O(R | q) \cdot \prod_{i=1}^{n} \frac{p(x_i | R, q)}{p(x_i | NR, q)}$$

• Since
$$x_i$$
 is either 0 or 1:
 $O(R|q,d) = O(R|q) \cdot \prod_{x_i=1} \frac{p(x_i=1|R,q)}{p(x_i=1|NR,q)} \cdot \prod_{x_i=0} \frac{p(x_i=0|R,q)}{p(x_i=0|NR,q)}$
• Let $p_i = p(x_i=1|R,q)$; $r_i = p(x_i=1|NR,q)$;

• Assume, for all terms not occurring in the query $(q_i=0)$ $p_i = r_i$ Then... Then...



Binary Independence Model

· All boils down to computing RSV.

$$RSV = \log \prod_{x_i=q_i=1}^{n} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i=q_i=1}^{n} \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$
$$RSV = \sum_{x_i=q_i=1}^{n} c_i; \quad c_i = \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

So, how do we compute *ci's* from our data ?

Binary Independence Model

- Estimating RSV coefficients.
- For each term / look at this table of document counts:

Documents	Relevant	Non-Relevant	Total
Xi=1	S	n-s	n
Xi=0	S-s	N-n-S+s	N-n
Total	S	N-S	N

• Estimates: $p_i \approx \frac{s}{S}$ $r_i \approx \frac{(n-s)}{(N-S)}$ $c_i \approx K(N, n, S, s) = \log \frac{s/(S-s)}{(n-s)/(N-n-S+s)}$ For now, assume no zero terms

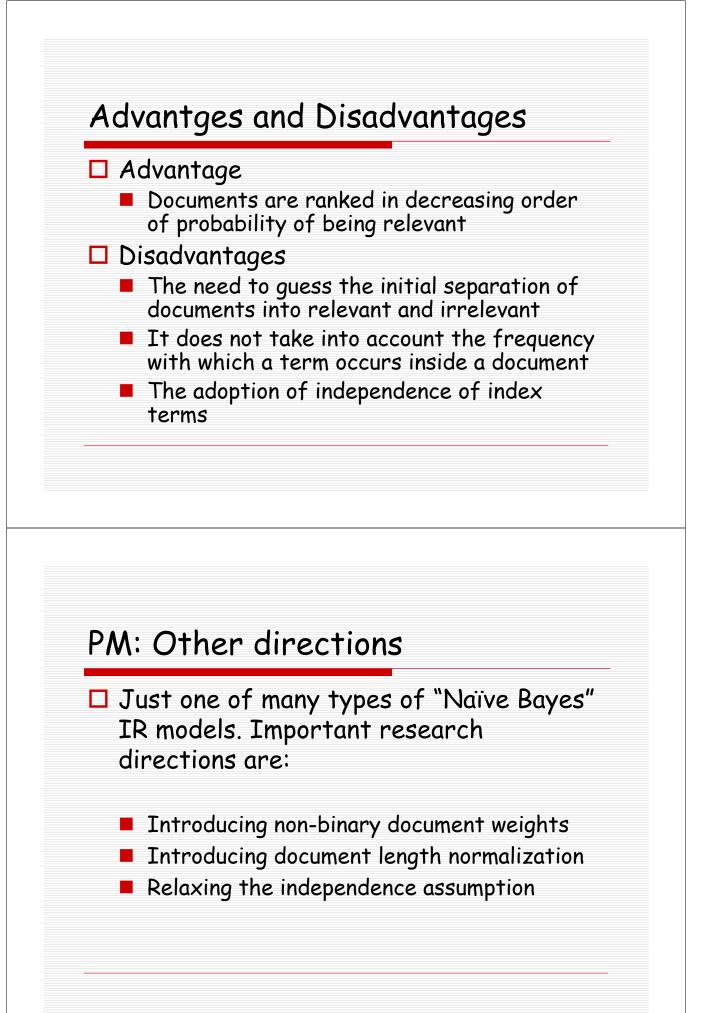
Estimation - key challenge

If non-relevant documents are approximated by the whole collection, then r_i (prob. of occurrence in nonrelevant documents for query) is n/N and

- $\log (1-r_i)/r_i = \log (N-n)/n \approx \log N/n = IDF!$
- p_i (probability of occurrence in relevant documents) can be estimated in various ways:
 - from relevant documents if know some
 - Relevance weighting can be used in feedback loop
 - constant (Croft and Harper combination match) then just get idf weighting of terms
 - proportional to prob. of occurrence in collection
 more accurately, to log of this (Greiff, SIGIR 1998)

Iteratively estimating p_i

- 1. Assume that p_i constant over all x_i in query
 - *p_i* = 0.5 (even odds) for any given doc
- 2. Determine guess of relevant document set:
 - V is fixed size set of highest ranked documents on this model (note: now a bit like tf.idf!)
- 3. We need to improve our guesses for p_i and r_i , so
 - Use distribution of x_i in docs in V. Let V_i be set of documents containing x_i
 - $\square p_i = |V_i| / |V|$
 - Assume if not retrieved then not relevant $r_i = (n_i - |V_i|) / (N - |V|)$
- 4. Go to 2. until converges then return ranking



Probabilistic Graphical Model for information retrieval
Use of directed graph to describe dependencies between variables (e.g. terms)
Algorithms for propagating probabilities (infering) by using the Bayes rule