CS4051 Information Retrieval Week 09

Muhammad Rafi

March 27, 2024

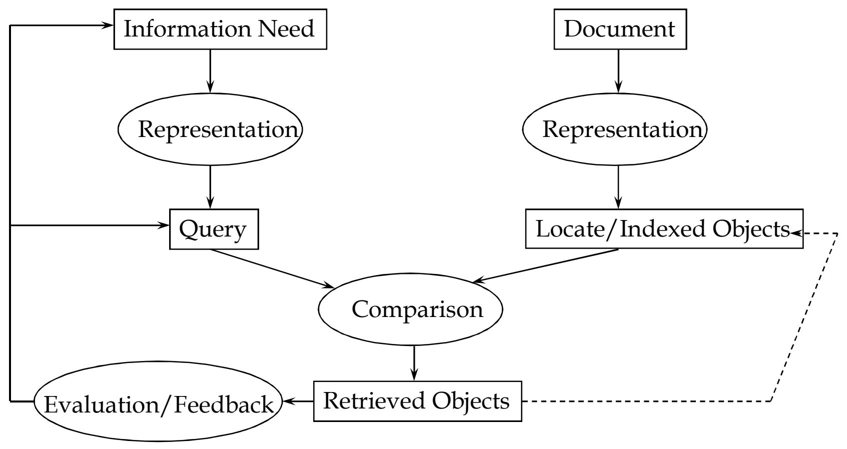
Probabilistic Information Retrieval

Chapter No. 11

Agenda

- Why use Probabilities in IR
- Basic Probability
- Probability for IR
- Probability Ranking Principle (PRP)
- Binary Independence Model (BIM)
- Relevance feedback in PIR
- Conclusion

Basic Information Retrieval Process



Why Probability in IR

- Information Retrieval deals with uncertain information
  - Probability is a measure of uncertainty
- Probabilistic Ranking Principle
  - provable
  - minimization of risk
- Probabilistic Inference
  - To justify your decision

Basic Probability

- *Probability* is the study of randomness and uncertainty.
- It can be define as ratio of favorable events over the total event space in consideration.
- A Probability is a number assigned to each subset (events) of a sample space Ω*.*

Axioms of Probability

- For any event A, $0 \leq P(A) \leq 1$.
- $P(\Omega) = 1$.
- If $A_1, A_2, \ldots A_n$ is a partition of A, then

  $$P(A) = P(A_1) + P(A_2) + \ldots + P(A_n)$$

  ($A_1, A_2, \ldots A_n$ is called a partition of A if $A_1 \cup A_2 \cup \ldots \cup A_n = A$ and $A_1, A_2, \ldots A_n$ are mutually exclusive.)

Properties of Probability

- For any event A, $P(A^c) = 1 - P(A)$.
- If $A \subset B$, then $P(A) \leq P(B)$.
- For any two events A and B,

  $$P(A \cup B) = P(A) + P(B) - P(A \cap B).$$

  For three events, A, B, and C,

  $P(A \cup B \cup C) = P(A) + P(B) + P(C) -$

  $P(A \cap B) - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C)$.

More Formal:

- Ω is the Sample Space:
  - Contains all possible outcomes of an experiment
- ω in Ω is a single outcome
- A in Ω is a set of outcomes of interest

1. $P(A) \geq 0 \forall A \in \Omega$
2. $P(\Omega) = 1$
3. $A_i \cap A_j = \emptyset \forall i, j \Rightarrow P(\cup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$
4. $P(\emptyset) = 0$

Independence

- The probability of independent events A, B and C is given by:

  P(A,B,C) = P(A)P(B)P(C)

A and B are independent, if knowing that A has happened does not say anything about B happening

Bayes Theorem

- Provides a way to convert *a-priori* probabilities to *a-posteriori* probabilities:

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

Basic Probability Theory

- For events A and B
  - Joint probability P(A, B) of both events occurring
  - Conditional probability P($A|B$) of event A occurring given that event B has occurred
- Chain rule gives fundamental relationship between joint and conditional probabilities:

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

- Similarly for the complement of an event $P(\overline{A})$:

$$P(\overline{A},B) = P(B|\overline{A})P(\overline{A})$$

- Partition rule: if B can be divided into an exhaustive set of disjoint subcases, then $P(B)$ is the sum of the probabilities of the subcases. A special case of this rule gives:

$$P(B) = P(A,B) + P(\overline{A},B)$$

Basic Probability Theory

Bayes' Rule for inverting conditional probabilities:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \left[\frac{P(B|A)}{\sum_{X\in\{A,\overline{A}\}} P(B|X)P(X)}\right] P(A)$$

Can be thought of as a way of updating probabilities:

- Start off with prior probability $P(A)$ (initial estimate of how likely event A is in the absence of any other information)
- Derive a posterior probability $P(A|B)$ after having seen the evidence B, based on the likelihood of B occurring in the two cases that A does or does not hold

Odds of an event provide a kind of multiplier for how probabilities change:

$$\text{Odds:} \quad O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

Probability Ranking Principle

- Collection of Documents
- User issues a query
- A Set of documents needs to be returned
- **Question: In what order to present documents to user ?**

Probability Ranking Principle

- **Question: In what order to present documents to user ?**
- Intuitively, want the “best” document to be first, second best - second, etc…
- Need a formal way to judge the “goodness” of documents w.r.t. queries.
- **Idea: Probability of relevance of the document w.r.t. query**

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.”

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.**

Need to find p(*R*|*x)* - probability that a retrieved document *x* is **relevant.**

$$p(R \mid x) = \frac{p(x \mid R)\,p(R)}{p(x)}$$

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

p(*R)*,p(*NR*) - prior probability of retrieving a (non) relevant document

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

Probability Ranking Principle

$$p(R \mid x) = \frac{p(x \mid R)\,p(R)}{p(x)}$$

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

Ranking Principle (Bayes' Decision Rule):

If p(*R*|*x)* > p(*NR*|*x*) then *x* <u>is relevant</u>, otherwise *x* <u>is not relevant</u>

Assumptions

- A1: One Random variable for each term(word).
- A2: d(w) are mutually independent given R.
- A3: P(0/R=1)=P(0/R=0)=0
- A4: If the word is not in the query, it is equally likely to occur in relevant and non-relevant populations(documents). practically: We only need to calculate probabilities of common words in query and documents.
- A5: On average, a query word will occur in half the relevant documents. Practical: pw and (1-pw) will cancel out.
- A6: Non-relevant set approximated by collection as a whole. (Most documents are non-relevant).

Probability Ranking Principle

- Let we have a document collection D, with |D| number of documents in it.
- Consider a fixed query "Q"
- Let N_1 is the document set from D which is relevant to Q. Let N_0 is the document set from D which is non-relevant to Q. If we know the prior for R and NR we can use this for calculation. { P(R) and P(NR)}
- Let x is a RV for a term appearing in a document.

Probability Ranking Principle

- Let x be the document vector for a document d, its relevance to query Q can be given as

  P(R/ d->*x,* q) = {P(x/R) * P(R) } / P(*x*)

  P(NR/ d->*x,* q) = {P(x/NR) * P(NR) } / P(*x*)
- In order to estimate *x->(*w_1*,*w_2*,..*w_n*)* we need to define two probabilities
  - p_w = {N_1(w) + 0.5} /{ N_1+1.0} and
  - $q_{w =}$ {N_0(w) + 0.5} /{ N_0+1.0}

Probability Ranking Principle

- Now, P(R/ d->*x,* Q) = {P(x/R) * P(R) } / P(*x*)

  P(R/ d_i,Q) = {P(d_i->x/R) * P(R)} / P(d_i->x)
- The retrieval value status (RSV) is given by Odd of R/NR probabilities (rank not probabilities- practically we have)

$$RSV(d_i,Q) = \frac{P(R/ d_i,Q) = \{P(d_i\text{->}x/R) * P(R)\} / P(d_i\text{->}x)}{P(NR/ d_i,Q) = \{P(d_i\text{->}x/NR) * P(NR)\} / P(d_i\text{->}x)}$$

Probability Ranking Principle

$$RSV(d_i,Q) = \frac{P(R/\ d_i,Q) = \{P(d_i\text{->}x/R) * P(R)\} / P(d_i\text{->}x)}{P(NR/\ d_i,Q) = \{P(d_i\text{->}x/NR) * P(NR)\} / P(d_i\text{->}x)}$$

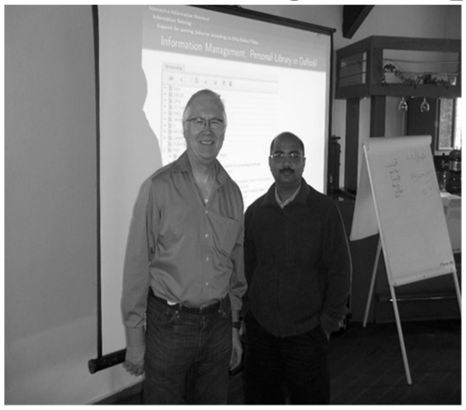
RSV(R/di,Q) = π (for all w ε di) { p_w* (1-q_w) / q_w * (1-p_w) }

Cost in PRP

- Lets associate cost to PRP

$$C_0 \cdot P(R=0|d) - C_1 \cdot P(R=1|d) \leq C_0 \cdot P(R=0|d') - C_1 \cdot P(R=1|d')$$

Probability Ranking Principle



The Gerard Salton Award is presented by the Association for Computing Machinery (ACM) Special Interest Group on Information Retrieval (SIGIR) 2012 - Norbert Fuhr, University of Duisburg-Essen: "Information Retrieval as Engineering Science."

Binary Independence Model

Given a query q, we wish to order returned documents by descending $P(R = 1|d, q)$. Under the BIM, this is modeled as ordering by $P(R = 1|\vec{x}, \vec{q})$. Rather than estimating this probability directly, because we are interested only in the ranking of documents, we work with some other quantities which are easier to compute and which give the same ordering of documents. In particular, we can rank documents by their odds of relevance (as the odds of relevance is monotonic with the probability of relevance). This makes things easier, because we can ignore the common denominator in (11.8), giving:

$$O(R|\vec{x},\vec{q}) = \frac{P(R=1|\vec{x},\vec{q})}{P(R=0|\vec{x},\vec{q})} = \frac{\frac{P(R=1|\vec{q})P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|\vec{q})}}{\frac{P(R=0|\vec{q})P(\vec{x}|R=0,\vec{q})}{P(\vec{x}|\vec{q})}} = \frac{P(R=1|\vec{q})}{P(R=0|\vec{q})} \cdot \frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})}$$

Binary Independence Model

The left term in the rightmost expression of Equation (11.10) is a constant for a given query. Since we are only ranking documents, there is thus no need for us to estimate it. The right-hand term does, however, require estimation, and this initially appears to be difficult: How can we accurately estimate the probability of an entire term incidence vector occurring? It is at this point that we make the *Naive Bayes conditional independence assumption* that the presence or absence of a word in a document is independent of the presence or absence of any other word (given the query):

$$\frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})} = \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

So:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

Binary Independence Model

Since each x_t is either 0 or 1, we can separate the terms to give:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t:x_t=1} \frac{P(x_t=1|R=1,\vec{q})}{P(x_t=1|R=0,\vec{q})} \cdot \prod_{t:x_t=0} \frac{P(x_t=0|R=1,\vec{q})}{P(x_t=0|R=0,\vec{q})}$$

Henceforth, let $p_t = P(x_t = 1|R = 1,\vec{q})$ be the probability of a term appearing in a document relevant to the query, and $u_t = P(x_t = 1|R = 0,\vec{q})$ be the probability of a term appearing in a nonrelevant document. These quantities can be visualized in the following contingency table where the columns add to 1:

	document	relevant ($R=1$)	nonrelevant ($R=0$)
Term present	$x_t = 1$	p_t	u_t
Term absent	$x_t = 0$	$1 - p_t$	$1 - u_t$

Binary Independence Model

Let us make an additional simplifying assumption that terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents: that is, if $q_t = 0$ then $p_t = u_t$. (This assumption can be changed, as when doing relevance feedback in Section 11.3.4.) Then we need only consider terms in the products that appear in the query, and so,

$$O(R|\vec{q},\vec{x}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0,q_t=1} \frac{1-p_t}{1-u_t}$$

The left product is over query terms found in the document and the right product is over query terms not found in the document.

Binary Independence Model

We can manipulate this expression by including the query terms found in the document into the right product, but simultaneously dividing through by them in the left product, so the value is unchanged. Then we have:

$$O(R|\vec{q},\vec{x}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \cdot \prod_{t:q_t=1} \frac{1-p_t}{1-u_t}$$

The left product is still over query terms found in the document, but the right product is now over all query terms. That means that this right product is a constant for a particular query, just like the odds $O(R|\vec{q})$. So the only quantity that needs to be estimated to rank documents for relevance to a query is the left product. We can equally rank documents by the logarithm of this term, since log is a monotonic function. The resulting quantity used for ranking is called the *Retrieval Status Value* (RSV) in this model:

$$RSV_d = \log \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} = \sum_{t:x_t=q_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)}$$

Binary Independence Model

So everything comes down to computing the RSV. Define c_t:

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t}$$

The c_t terms are log odds ratios for the terms in the query. We have the odds of the term appearing if the document is relevant ($p_t/(1-p_t)$) and the odds of the term appearing if the document is nonrelevant ($u_t/(1-u_t)$). The *odds ratio* is the ratio of two such odds, and then we finally take the log of that quantity. The value will be 0 if a term has equal odds of appearing in relevant and nonrelevant documents, and positive if it is more likely to appear in relevant documents. The c_t quantities function as term weights in the model, and the document score for a query is $RSV_d = \sum_{x_t=q_t=1} c_t$. Operationally, we sum them in accumulators for query terms appearing in documents, just as for the vector space model calculations discussed in Section 7.1 (page 135). We now turn to how we estimate these c_t quantities for a particular collection and query.

Binary Independence Model

Probability estimates in theory

For each term t, what would these c_t numbers look like for the whole collection? (11.19) gives a contingency table of counts of documents in the collection, where df_t is the number of documents that contain term t:

	documents	relevant	nonrelevant	Total
Term present	$x_t = 1$	s	$\mathrm{df}_t - s$	df_t
Term absent	$x_t = 0$	$S - s$	$(N - \mathrm{df}_t) - (S - s)$	$N - \mathrm{df}_t$
	Total	S	$N - S$	N

Using this, $p_t = s/S$ and $u_t = (\mathrm{df}_t - s)/(N - S)$ and

$$c_t = K(N, \mathrm{df}_t, S, s) = \log \frac{s/(S-s)}{(\mathrm{df}_t - s)/((N - \mathrm{df}_t) - (S - s))}$$

Binary Independence Model

	documents	relevant	nonrelevant	Total
Term present	$x_t = 1$	s	$\mathrm{df}_t - s$	df_t
Term absent	$x_t = 0$	$S - s$	$(N - \mathrm{df}_t) - (S - s)$	$N - \mathrm{df}_t$
	Total	S	$N - S$	N

Using this, $p_t = s/S$ and $u_t = (\mathrm{df}_t - s)/(N - S)$ and

$$c_t = K(N, \mathrm{df}_t, S, s) = \log \frac{s/(S-s)}{(\mathrm{df}_t - s)/((N - \mathrm{df}_t) - (S - s))}$$

To avoid the possibility of zeroes (such as if every or no relevant document has a particular term) it is fairly standard to add $\frac{1}{2}$ to each of the quantities in the center 4 terms of (11.19), and then to adjust the marginal counts (the totals) accordingly (so, the bottom right cell totals $N + 2$). Then we have:

$$\hat{c}_t = K(N, \mathrm{df}_t, S, s) = \log \frac{(s + \frac{1}{2})/(S - s + \frac{1}{2})}{(\mathrm{df}_t - s + \frac{1}{2})/(N - \mathrm{df}_t - S + s + \frac{1}{2})}$$

Pseudo Relevance Feedback

- Retrieval Status Value (RSV)

	documents	relevant	nonrelevant	total
term present	$x_t = 1$	s	$\mathrm{df}_t - s$	df_t
term absent	$x_t = 0$	$S - s$	$(N - \mathrm{df}_t) - (S - s)$	$N - \mathrm{df}_t$
	total	S	$N - S$	N

Using this, $p_t = s/S$ and $u_t = (\mathrm{df}_t - s)/(N - S)$ and

$$RSV_d = \log \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} = \sum_{t:x_t=q_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)}.$$

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t}.$$

$$c_t = K(N, \mathrm{df}_t, S, s) = \log \frac{s/(S-s)}{(\mathrm{df}_t - s)/((N - \mathrm{df}_t) - (S - s))}.$$

Relevance Feedback in Prob. IR

1. Guess initial estimates of p_t and u_t. This can be done using the probability estimates of the previous section. For instance, we can assume that p_t is constant over all x_t in the query, in particular, perhaps taking $p_t = \frac{1}{2}$.
2. Use the current estimates of p_t and u_t to determine a best guess at the set of relevant documents $R = \{d : R_{d,q} = 1\}$. Use this model to retrieve a set of candidate relevant documents, which we present to the user.
3. We interact with the user to refine the model of R. We do this by learning from the user relevance judgments for some subset of documents V. Based on relevance judgments, V is partitioned into two subsets: $VR = \{d \in V, R_{d,q} = 1\} \subset R$ and $VNR = \{d \in V, R_{d,q} = 0\}$, which is disjoint from R.
4. We reestimate p_t and u_t on the basis of known relevant and nonrelevant documents. If the sets VR and VNR are large enough, we may be able to estimate these quantities directly from these documents as maximum likelihood estimates:

$$p_t = |VR_t|/|VR|$$

Relevance Feedback in Prob. IR

where VR_t is the set of documents in VR containing x_t. In practice, we usually need to smooth these estimates. We can do this by adding $\frac{1}{2}$ to both the count $|VR_t|$ and to the number of relevant documents not containing the term, giving:

$$p_t = \frac{|VR_t| + \frac{1}{2}}{|VR| + 1}.$$

However, the set of documents judged by the user (V) is usually very small, and so the resulting statistical estimate is quite unreliable (noisy), even if the estimate is smoothed. So it is often better to combine the new information with the original guess in a process of Bayesian updating. In this case we have:

$$p_t^{(k+1)} = \frac{|VR_t| + \kappa p_t^{(k)}}{|VR| + \kappa}.$$

Relevance Feedback in Prob. IR

Here $p_t^{(k)}$ is the k^{th} estimate for p_t in an iterative updating process and is used as a Bayesian prior in the next iteration with a weighting of κ. Relating this equation back to Equation (11.4) requires a bit more probability theory than we have presented here (we need to use a beta distribution prior, conjugate to the Bernoulli random variable X_t). But the form of the resulting equation is quite straightforward: Rather than uniformly distributing pseudocounts, we now distribute a total of κ pseudocounts according to the previous estimate, which acts as the prior distribution. In the absence of other evidence (and assuming that the user is perhaps indicating roughly five relevant or nonrelevant documents) then a value of around $\kappa = 5$ is perhaps appropriate. That is, the prior is strongly weighted so that the estimate does not change too much from the evidence provided by a very small number of documents.

Relevance Feedback in Prob. IR

5. Repeat the above process from Step 2, generating a succession of approximations to R and hence p_t, until the user is satisfied.

Pseudo Relevance Feedback

1. Assume initial estimates for p_t and u_t as above.
2. Determine a guess for the size of the relevant document set. If unsure, a conservative (too small) guess is likely to be best. This motivates use of a fixed size set V of highest ranked documents.
3. Improve our guesses for p_t and u_t. We choose from the methods of Equations (11.23) and (11.25) for reestimating p_t, except now based on the set V instead of VR. If we let V_t be the subset of documents in V containing x_t and use add $\frac{1}{2}$ smoothing, we get:

$$p_t = \frac{|V_t| + \frac{1}{2}}{|V| + 1}$$

and if we assume that documents that are non retrieved are nonrelevant then we can update our u_t estimates as:

$$u_t = \frac{\mathrm{df}_t - |V_t| + \frac{1}{2}}{N - |V| + 1}.$$

Pseudo Relevance Feedback

…

4. Go to Step 2 until the ranking of the returned results converges.

Once we have a real estimate for p_t, then the c_t weights used in the RSV value look almost like a tf–idf value. For instance, using Equation (11.18), Equation (11.22), and Equation (11.26), we have:

$$c_t = \log\left[\frac{p_t}{1-p_t} \cdot \frac{1-u_t}{u_t}\right] \approx \log\left[\frac{|V_t| + \frac{1}{2}}{|V| - |V_t| + 1} \cdot \frac{N}{\mathrm{df}_t}\right].$$

But things aren't quite the same: $p_t/(1-p_t)$ measures the (estimated) proportion of relevant documents that the term t occurs in, not term frequency. Moreover, if we apply log identities:

$$c_t = \log \frac{|V_t| + \frac{1}{2}}{|V| - |V_t| + 1} + \log \frac{N}{\mathrm{df}_t}$$

we see that we are now *adding* the two log-scaled components rather than multiplying them.