

Parametric Search and Machine Learning Based Scoring

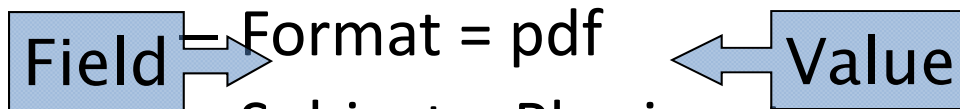
Reference: Introduction to Information Retrieval
by C. Manning, P. Raghavan, H. Schutze

Parametric Search

Parametric search

- Most documents have, in addition to text, some “meta-data” in fields e.g.,

- Language = French



- Format = pdf

- Subject = Physics etc.

- Date = Feb 2000

- A parametric search interface allows the user to combine a full-text query with selections on these field values e.g.,
 - language, date range, etc.

Parametric search example

CarFinder.com 

Over one million fictional vehicles to choose from!

Choose your search criteria from the drop down menus:

Number of results to display: 50

Make Model Category Year
City Color Price

Search 

Reset Filters

Reset Sorts

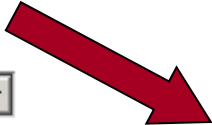
Notice that the output is a (large) table. Various parameters in the table (column headings) may be clicked on to effect a sort.

Make	Model	Year	City	Mileage	Price	Category	Description	Color
BMW	5-Series	1995	San Francisco	16100	11100	Luxury	Never driven in winter conditions. Body work makes it look like new. Keyless entry and security features. This is a bargain.	Silver
BMW	5-Series	1995	San Francisco	16600	11100	Luxury	Great first car for your teen-aged kid. Solid, dependable, affordable with 0% down and owner financing.	Blue
BMW	5-Series	1995	San Francisco	16800	11200	Luxury	Upgraded sound system really rocks. Customized interior features wood grain dash and beige leather seats. Power locks, windows, steering. Price firm.	White
BMW	5-Series	1995	San Francisco	16100	11300	Luxury	Safe choice for a young family: ABS, driver and passenger air bags. Roomy interior with power everything. Low mileage driving kids back and forth to soccer.	Maroon
BMW	5-Series	1995	San Francisco	16300	11400	Luxury	This baby's got it all: power steering, cruise, power locks, power windows, remote entry, leather interior, security alarm, AM/FM/CD/Cassette. Priced to sell!	Brown

Parametric search example

CarFinder.com 
Over one million fictional vehicles to choose from!

We can add text search.



Choose your search criteria from the drop down menus:

Make

Model

Category

Year

City

Color

Price

Description



Make	Model	Year	City	Mileage 	Price 	Category	Description	Color
BMW	5-Series	1997	San Francisco	14300	13100	Luxury	5-speed, heavy-duty suspension, extra wide tires. Well-maintained by mechanic-owner. Cloth seats and upgraded stereo system.	White
BMW	5-Series	1997	San Francisco	14600	13100	Luxury	Is that price for real? You bet it is. Fully loaded with all factory options. Former floor model.	Beige
BMW	5-Series	1997	San Francisco	14900	13100	Luxury	Fun to drive. Manual 5-speed transmission, turbo charger. Garaged all winter and pampered the rest of the year. This is a steal!	Orange
BMW	5-Series	1997	San Francisco	14800	13200	Luxury	Fully loaded, automatic transmission. Power everything. Anti-lock brakes and full safety features. Must test drive. Price firm.	Green
BMW	5-Series	1997	San Francisco	14300	13200	Luxury	Formerly an executive's vehicle. Interior has been professionally maintained, engine factory serviced every 3000 miles. Great gas mileage. Price negotiable.	Maroon
BMW	5-Series	1997	San Francisco	15000	13200	Luxury	Sun roof, air, CD player, driver side air bag. 10% deposit required. Owner financing available. Best offer by end of weekend buys it.	Red

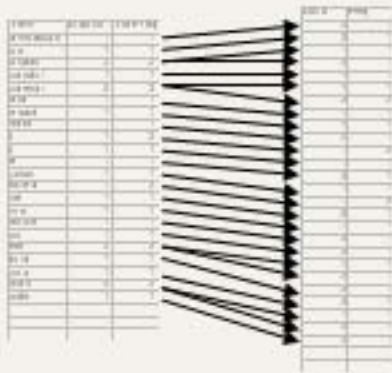
Number of results to display:

5

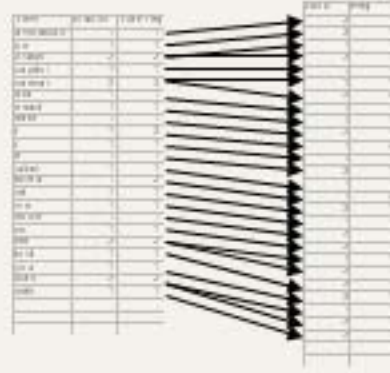
Zones

- A zone is an identified region within a doc
 - E.g., Title, Abstract, Bibliography
 - Generally culled from marked-up input or document metadata (e.g., powerpoint)
- Contents of a zone are free text
 - Not a “finite” vocabulary
- Indexes for each zone - allow queries like
 - ***sorting*** in Title AND ***smith*** in Bibliography AND ***recur**** in Body

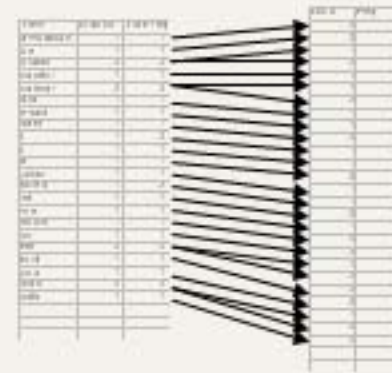
Zone indexes – simple view



Title



Author



Body

etc.

Scoring and Ranking

Scoring

- Thus far, our queries have all been Boolean
 - Docs either match or not
- OK for expert users with precise understanding of their needs and the corpus
- Not good for (the majority of) users with poor Boolean formulation of their needs
- Most users don't want to wade through 1000's of results – cf. use of web search engines

Scoring

- *We wish to return in order the documents most likely to be useful to the searcher*
- How can we rank order the docs in the corpus with respect to a query?
- Assign a score – say in $[0,1]$ for each doc d on each query q

Linear zone combinations

- First generation of scoring methods: use a linear combination of Booleans:

– e.g.,

$$\text{Score} = 0.6 * \langle \textit{sorting} \text{ in } \underline{\text{Title}} \rangle + 0.3 * \langle \textit{sorting} \text{ in } \underline{\text{Abstract}} \rangle + 0.05 * \langle \textit{sorting} \text{ in } \underline{\text{Body}} \rangle + 0.05 * \langle \textit{sorting} \text{ in } \text{Boldface} \rangle$$

- Each expression such as $\langle \textit{sorting} \text{ in } \underline{\text{Title}} \rangle$ takes on a value in $\{0,1\}$.

Linear zone combinations

- In fact, the expressions between $\langle \rangle$ on the last slide could be *any* Boolean query
- We are given a weight vector whose components sum up to 1.
 - There is a weight for each zone/field.
- Then the overall score is in $[0,1]$.
 - Remark: For this example, the scores can only take on a finite set of values.

Linear zone combinations

- Given a Boolean query, we assign a score to each doc by adding up the weighted contributions of the zones/fields.
- The retrieval model for document d and query q is:

$$score(d, q) = w_1 s_1(d, q) + \cdots + w_m s_m(d, q)$$

where $s_i(d, q)$ denotes the Boolean result for zone i
and $\sum_i w_i = 1$

Linear zone combinations

- Most commonly, a query parser that takes the user's query and runs it on the indexes for each zone
- Typically - users want to see the K highest-scoring docs.

Machine Learning Based Scoring

Where do these weights come from?

- Machine learned scoring
- Given
 - *A test corpus*
 - *A suite of test queries*
 - *A set of relevance judgments*
- Learn a set of weights such that relevance judgments matched

Simple example

- Each doc has two zones, Title and Body
- For a chosen $w \in [0,1]$, score for doc d on query q

$$\text{score}(d, q) = w \cdot s_T(d, q) + (1 - w)s_B(d, q)$$

where:

$s_T(d, q) \in \{0,1\}$ is a Boolean denoting whether q matches the Title and

$s_B(d, q) \in \{0,1\}$ is a Boolean denoting whether q matches the Body

Learning w from training examples

Example	DocID	Query	s_T	s_B	Judgment
Φ_1	37	linux	1	1	Relevant
Φ_2	37	penguin	0	1	Non-relevant
Φ_3	238	system	0	1	Relevant
Φ_4	238	penguin	0	0	Non-relevant
Φ_5	1741	kernel	1	1	Relevant
Φ_6	2094	driver	0	1	Relevant
Φ_7	3191	driver	1	0	Non-relevant

We are given *training examples*, each of which is a triple:

DocID d , Query q , and Judgment *Relevant/Non-relevant*.

From these, we will learn the best value of w .

How?

- For each example Φ_t we can compute the score based on

$$\text{score}(d_t, q_t) = w \cdot s_T(d_t, q_t) + (1 - w)s_B(d_t, q_t).$$

- We quantify Relevant as 1 and Non-relevant as 0
 - Would like the choice of w to be such that the computed scores are as close to these 1/0 judgments as possible
 - Denote by $r(d_t, q_t)$ the judgment for Φ_t
- Then minimize total squared error

$$\sum_{\Phi_t} (r(d_t, q_t) - \text{score}(d_t, q_t))^2$$

Optimizing w

- There are 4 kinds of training examples

s_T	s_B	Score
0	0	0
0	1	$1 - w$
1	0	w
1	1	1

Optimizing w

- There are 8 possible values for errors

s_T	s_B	Score
0	0	0
0	1	$1 - w$
1	0	w
1	1	1

Judgment=1 $\Rightarrow$ Error= w

Judgment=0 $\Rightarrow$ Error= $1 - w$

- Let n_{01r} be the number of training examples for which $s_T(d, q)=0$, $s_B(d, q)=1$, judgment = *Relevant*.
- Similarly define n_{00r} , n_{10r} , n_{11r} , n_{00i} , n_{01i} , n_{10i} , n_{11i}

Total error – then calculus

- Add up from various cases to get the total error
- The total error E is:

$$E = (n_{01r} + n_{10i})w^2 + (n_{10r} + n_{01i})(1 - w)^2 + n_{00r} + n_{11i}$$

- Now differentiate with respect to w to get optimal value
- Set $\frac{d(E)}{d(w)} = 0$, we get
$$2w(n_{01r} + n_{10i}) + (n_{10r} + n_{01i})2(1 - w)(-1) = 0$$

Optimal weight parameter

$$2w(n_{01r} + n_{10i}) + (n_{10r} + n_{01i})2(1 - w)(-1) = 0$$

$$w(n_{01r} + n_{10i}) + (w - 1)(n_{10r} + n_{01i}) = 0$$

$$w = \frac{n_{10r} + n_{01i}}{n_{10r} + n_{10i} + n_{01r} + n_{01i}}$$

Full text queries

- Most users more likely to type ***bill rights*** or ***bill of rights***
 - How do we interpret these *full text* queries?
 - No Boolean connectives
 - Of several query terms, some may be missing in a doc
 - Only some query terms may occur in the title, etc.

Scoring: density-based

- Thus far: position and overlap of terms in a doc – title, author etc.
- Obvious next idea: if a document talks about a topic *more*, then it is a better match
- This applies even when we only have a single query term.
- Document relevant if it has many occurrences of the term(s)
- This leads to the idea of term weighting.