Ranking Tutorial

IRE Minor Project

Why Ranking?

- Rather than returning a set of documents that satisfy a query expression, you're expected to return the "top K" documents that satisfy the query. How to figure out what the top k is?
- Issue with boolean retrieval:
  - Query 1: "standard user dlink 650" → 200,000 hits
  - Query 2: "standard user dlink 650 no card found": 0 hits

Term Frequency Count

- Store the term frequencies in the posting list.
- The more frequent the query term in the document, the higher the ranking of that document should be.
- If the query term does not occur in the document: score should be 0.
- Since you’re storing the fields in which the term occurs in that document, have different weightages for the term occurring in title, infobox, body etc.
    - Document with the query term occurring is expected to more relevant than a document with the term in the body.

Term Document Count Matrices

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Why Term Frequency- Inverse Document Frequency (Tf-IDF)?

- Term weights have two components:
  - Local = How important is this term in the document?
  - Global = How important is the term in the collection?
- Intuition:
  - Terms that appear often in a document should have high weights.
  - Terms that appear in many documents should have low weights.
- Term Frequency to capture local
- Inverse document frequency to capture global

Term Frequency

- The term frequency $tf_{t,d}$ of term *t* in document *d* is defined as the number of times that *t* occurs in *d*
- Relevance does not increase proportionally with term frequency so use log frequency weighting.
- The frequency weight of term *t* in document *d* is $w_{t,d} = 1 + \log(tf_{t,d})$, if $tf_{t,d} > 0$, else 0.

Inverse Document Frequency IDF

– – –

- Frequent terms are less informative than rare terms
- df_t is the document frequency of t : the number of documents containing t.
- $idf_t = \log (N / df_t)$
- IDF has no effect on ranking one term queries
  - IDF affects the ranking of documents for queries with at least two terms
  - For the query capricious person, IDF weighting makes occurrences of capricious count for much more in the final document ranking than occurrences of person.

TF-IDF Weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight.
  - $W_{t,d} = \log(1 + tf_{t,d}) * \log(N / df_t)$
- Variants to consider :

Term frequency		Document frequency		Normalization	
n (natural)	$tf_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1 + \log(tf_{t,d})$	t (idf)	$\log \frac{N}{df_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5 + \frac{0.5 \times tf_{t,d}}{\max_t(tf_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-df_t}{df_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases} 1 & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha < 1$
L (log ave)	$\frac{1+\log(tf_{t,d})}{1+\log(ave_{t \in d}(tf_{t,d}))}$				

Tf - IDF Weighted Matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Ranking for Minor Project

- Calculate tf-idf weights for each field separately
  - Eg. Calculate the frequencies and tf-idf scores for title differently than body
- Use different weights for different fields.
  - Eg. Title,infobox having a higher weightage than body

References

- Ranking slide from sep 1 on Moodle
- Information Retrieval: tf-idf and Vector Ranking Models （youtube lecture in Materials)
- Chapters 6,7,11 of Intro to IRE textbook