**Term frequency and weighting**

Thus far, scoring has hinged on whether or not a query term is present in a zone within a document. We take the next logical step: a document or zone that mentions a query term more often has more to do with that query and therefore should receive a higher score. To motivate this, we recall the notion of a *free text query* introduced in Section [1.4](https://nlp.stanford.edu/IR-book/html/htmledition/the-extended-boolean-model-versus-ranked-retrieval-1.html#sec:boolean-querying) : a query in which the terms of the query are typed freeform into the search interface, without any connecting search operators (such as Boolean operators). This query style, which is extremely popular on the web, views the query as simply a set of words. A plausible scoring mechanism then is to compute a score that is the sum, over the query terms, of the match scores between each query term and the document.

Towards this end, we assign to each term in a document a *weight* for that term, that depends on the number of occurrences of the term in the document. We would like to compute a score between a query term $t$ and a document $d$, based on the weight of $t$ in $d$. The simplest approach is to assign the weight to be equal to the number of occurrences of term $t$ in document $d$. This weighting scheme is referred to as *term frequency* and is denoted $\mbox{tf}_{t,d}$, with the subscripts denoting the term and the document in order.

For a document $d$, the set of weights determined by the $\mbox{tf}$ weights above (or indeed any weighting function that maps the number of occurrences of $t$ in $d$ to a positive real value) may be viewed as a quantitative digest of that document. In this view of a document, known in the literature as the *bag of words model* , the exact ordering of the terms in a document is ignored but the number of occurrences of each term is material (in contrast to Boolean retrieval). We only retain information on the number of occurrences of each term. Thus, the document ``Mary is quicker than John'' is, in this view, identical to the document ``John is quicker than Mary''. Nevertheless, it seems intuitive that two documents with similar bag of words representations are similar in content. We will develop this intuition further in Section [6.3](https://nlp.stanford.edu/IR-book/html/htmledition/the-vector-space-model-for-scoring-1.html#sec:docvectors) .

Before doing so we first study the question: are all words in a document equally important? Clearly not; in Section [2.2.2](https://nlp.stanford.edu/IR-book/html/htmledition/dropping-common-terms-stop-words-1.html#sec:stopwords) (page [[*]](https://nlp.stanford.edu/IR-book/html/htmledition/dropping-common-terms-stop-words-1.html#p:stopwords)) we looked at the idea of *stop words* - words that we decide not to index at all, and therefore do not contribute in any way to retrieval and scoring

## Inverse document frequency

Raw term frequency as above suffers from a critical problem: all terms are considered equally important when it comes to assessing relevancy on a query. In fact certain terms have little or no discriminating power in determining relevance. For instance, a collection of documents on the auto industry is likely to have the term auto in almost every document. To this end, we introduce a mechanism for attenuating the effect of terms that occur too often in the collection to be meaningful for relevance determination. An immediate idea is to scale down the term weights of terms with high collection frequency, defined to be the total number of occurrences of a term in the collection. The idea would be to reduce the $\mbox{tf}$ weight of a term by a factor that grows with its collection frequency.

Instead, it is more commonplace to use for this purpose the *document frequency* $\mbox{df}_t$, defined to be the number of documents in the collection that contain a term $t$. This is because in trying to discriminate between documents for the purpose of scoring it is better to use a document-level statistic (such as the number of documents containing a term) than to use a collection-wide statistic for the term.

|  |
| --- |
| \begin{figure}\begin{tabular}{\vert l\vert l\vert l\vert} \hline % after \\ : ... ...10422 & 8760\\ insurance & 10440 & 3997 \\ \hline \end{tabular} \end{figure} |
| **Figure 6.7:** Collection frequency (cf) and document frequency (df) behave differently, as in this example from the Reuters collection. |

The reason to prefer df to cf is illustrated in Figure [6.7](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#fig:cfdf) , where a simple example shows that collection frequency (cf) and document frequency (df) can behave rather differently. In particular, the cf values for both try and insurance are roughly equal, but their df values differ significantly. Intuitively, we want the few documents that contain insurance to get a higher boost for a query on insurance than the many documents containing try get from a query on try.

How is the document frequency df of a term used to scale its weight? Denoting as usual the total number of documents in a collection by $N$, we define the *inverse document frequency* of a term $t$ as follows:

|  |  |
| --- | --- |
| \begin{displaymath} \mbox{idf}_t = \log {N\over \mbox{df}_t}. \end{displaymath} | (21) |

Thus the idf of a rare term is high, whereas the idf of a frequent term is likely to be low. Figure [6.8](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#fig:figureidf) gives an example of idf's in the Reuters collection of 806,791 documents; in this example logarithms are to the base 10. In fact, as we will see in Exercise [6.2.2](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#ex:logbase) , the precise base of the logarithm is not material to ranking. We will give on page [11.3.3](https://nlp.stanford.edu/IR-book/html/htmledition/probability-estimates-in-practice-1.html#p:justificationofidf) a justification of the particular form in Equation [21](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#eqn:idf).

\begin{figure}
% latex2html id marker 7963
\par
\begin{tabular}{\vert\vert l\ver...
...quencies in the Reuters collection of 806,791 documents.}
\par
\par
\end{figure}

## Tf-idf weighting

We now combine the definitions of term frequency and inverse document frequency, to produce a composite weight for each term in each document. The *tf-idf* weighting scheme assigns to term $t$ a weight in document $d$ given by

|  |  |
| --- | --- |
| \begin{displaymath} \mbox{tf-idf}_{t,d} = \mbox{tf}_{t,d} \times \mbox{idf}_t. \end{displaymath} | (22) |

In other words, $\mbox{tf-idf}_{t,d}$ assigns to term $t$ a weight in document $d$ that is

1. highest when $t$ occurs many times within a small number of documents (thus lending high discriminating power to those documents);
2. lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);
3. lowest when the term occurs in virtually all documents.

At this point, we may view each document as a *vector* with one component corresponding to each term in the dictionary, together with a weight for each component that is given by ([22](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#eqn:tfidf)). For dictionary terms that do not occur in a document, this weight is zero. This vector form will prove to be crucial to scoring and ranking; we will develop these ideas in Section [6.3](https://nlp.stanford.edu/IR-book/html/htmledition/the-vector-space-model-for-scoring-1.html#sec:docvectors) . As a first step, we introduce the *overlap score measure*: the score of a document $d$ is the sum, over all query terms, of the number of times each of the query terms occurs in $d$. We can refine this idea so that we add up not the number of occurrences of each query term $t$ in $d$, but instead the tf-idf weight of each term in $d$.

|  |  |
| --- | --- |
| \begin{displaymath} \mbox{Score}(q,d)=\sum_{t\in q} \mbox{tf-idf}_{t,d}. \end{displaymath} | (23) |

In Section [6.3](https://nlp.stanford.edu/IR-book/html/htmledition/the-vector-space-model-for-scoring-1.html#sec:docvectors) we will develop a more rigorous form of Equation [23](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#eqn:docscore).

**Exercises.**

* Why is the idf of a term always finite?
* What is the idf of a term that occurs in every document? Compare this with the use of stop word lists.
* Consider the table of term frequencies for 3 documents denoted Doc1, Doc2, Doc3 in Figure [6.9](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#fig:tfgraph) .

|  |
| --- |
| \begin{figure}\begin{tabular}{\vert\vert l\vert r\vert r\vert r\vert\vert} \hlin... ...rance & 0 & 33 & 29 \\ best & 14 & 0 & 17 \\ \hline \end{tabular} \end{figure} |
| **Figure 6.9:** Table of tf values for Exercise [6.2.2](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#ex:tfidf). |

* Compute the tf-idf weights for the terms car, auto, insurance, best, for each document, using the idf values from Figure [6.8](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#fig:figureidf) .
* Can the tf-idf weight of a term in a document exceed 1?
* How does the base of the logarithm in ([21](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#eqn:idf)) affect the score calculation in ([23](https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html#eqn:docscore))? How does the base of the logarithm affect the relative scores of two documents on a given query?
* If the logarithm in ([21](https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html#eqn:idf)) is computed base 2, suggest a simple approximation to the idf of a term.