Information Retrivel: Assignment 2

Shifei Chen

Exercise 6.1

No it's not necessary for all zones to use the same boolean match function. The weighted zone score is defined as $\sum_{i=1}^{l} g_i s_i$, it doesn't really matter what boolean match function it uses as long as the boolean match function correctly matches s_i to 0 or 1.

Exercise 6.2

Let's start with the case where the term appeared in the title zone but not in the body or author zone. The score of this document would be

$$\sum_{i=1}^{3} g_i s_i = 0 * 0.2 + 1 * 0.31 + 0 * 0.49 = 0.31$$

The whole table of all possible values a document may get is listed below, where 1 means the term appeared in that zone and 0 means not.

Author	**Title**	**Body**	**Score**
0	0	0	0
0	0	1	0.49
0	1	0	0.31
0	1	1	0.8
1	0	0	0.2
1	0	1	0.69
1	1	0	0.51
1	1	1	1

Exercise 6.3

```
ZoneScore({t1, ..., tn})
    float scores[N] = [0]
    constant g[1]
    terms = SortByIncreasingFrequency({t1, ..., tn})
    q1 = first(terms)
    terms = rest(terms)
    while terms != NIL and scores != NIL
    do scores = ZoneScore(q1, first(terms))
        terms = rest(terms)
    return scores
```

Exercise 6.4

```
Weighted(p1, p2, g)
    float sum = 0
    // z1 and z2 are initialized to point at the beginning of p1 and p2
    z1 = zones(p1)
    z2 = zones(p2)
    while z1 != NIL
    do sum = weight(g[z1])
        z1 = next(z1)
    while z2 != NIL
    do sum = weight(g[z2])
        z2 = next(z2)
    return sum
```

Exercise 6.5

From Figure 6.5 we could see that there is only one case where $S_T = 1$ and $S_B = 0$ and its judgement is non-relevant, hence $n_{10n} = 1$. The full list is like the one below

$$n_{10r} = 0$$
$$n_{01n} = 1$$
$$n_{10n} = 1$$
$$n_{01r} = 2$$

Therefore the optimal value of g is

$$\frac{n_{10r} + n_{01n}}{n_{10r} + n_{10n} + n_{01r} + n_{01n}} = \frac{0 + 1}{0 + 1 + 2 + 1} = 0.25$$

Exercise 6.6

Take searching term "linux" in document 37 as an example to see how to calculate the weighted zone score.

$$score(d_{37}, q_{linux}) = 0.25 * 1 + (1 - 0.25) * 1 = 1$$

The full list of the scores are listed in the table.

$score(d_j, q_j)$	**Value**	**Relevance**	**Error**
$score(d_{37}, q_{linux})$	1	1	0
$score(d_{37}, q_{penguin})$	0.75	0	0.5625
$score(d_{238}, q_{system})$	0.75	1	0.0625
$score(d_{238}, q_{penguin})$	0	0	0
$score(d_{1741}, q_{kernel})$	1	1	0
$score(d_{2094}, q_{driver})$	0.75	1	0.0625
$score(d_{3191}, q_{driver})$	0.25	0	0.0625

If the relevance is quantized to 0/1, then we can use calculate the error value to see how the weighted zone score is realted to the real relevance.

Exercise 6.8

$idf_t = \log \frac{N}{df_t}$, since $0 \leq df \leq N$ and $df_t \in \mathbb{N}$, it's easy to see $0 \leq idf_t \leq \log N$. Its lower limit happens when $df_t = N$ and is never going to be infinite. Also $df_t = 0$ is not possible since all of the terms by definition came from the docment collection and should at least appeared once. The denominator df_t is always larger than 1 hence the upper limit is $\log N$.

Exercise 6.9

In this case, $idf_t = \log \frac{N}{df_t} = \log \frac{N}{N} = 0$. Hence it's a good idea to put the k most frequent words into the stop words list. Otherwise imagine we have a term appears in every document, like "the", and a document that only contains "the", we can't calculate the cosine similarity of that document to anyone else and it's hard to search "the the" as well.

Exercise 6.10

$$
\begin{aligned}
tf\text{-}idf_{1,car} &= 27 * 1.65 = 44.55 \\
tf\text{-}idf_{2,car} &= 4 * 1.65 = 6.6 \\
tf\text{-}idf_{3,car} &= 24 * 1.65 = 39.6 \\
tf\text{-}idf_{1,auto} &= 3 * 2.08 = 6.24 \\
tf\text{-}idf_{2,auto} &= 33 * 2.08 = 68.64 \\
tf\text{-}idf_{3,auto} &= 0 * 2.08 = 0 \\
tf\text{-}idf_{1,insurance} &= 0 * 1.62 = 0 \\
tf\text{-}idf_{2,insurance} &= 33 * 1.62 = 53.46 \\
tf\text{-}idf_{3,insurance} &= 29 * 1.62 = 46.98 \\
tf\text{-}idf_{1,best} &= 14 * 1.5 = 21 \\
tf\text{-}idf_{2,best} &= 0 * 1.5 = 0 \\
tf\text{-}idf_{3,best} &= 17 * 1.5 = 25.5
\end{aligned}
$$

Exercise 6.11

It is possible. See the exerice above as an example.

Exercise 6.15

Take $\vec{v_1}$ as an example

$$
\begin{aligned}
\vec{V_1} &= [44.55, 6.24, 0, 21] \\
|\vec{V_1}| &= \sqrt{44.55^2 + 6.24^2 + 0^2 + 21^2} = 49.645 \\
\vec{v_1} &= \frac{\vec{V_1}}{|\vec{V_1}|} = [0.897, 0.126, 0, 0.423]
\end{aligned}
$$

And all the other Euclidean normalized document vectors are

$$
\begin{aligned}
\vec{v_2} &= [0.076, 0.787, 0.613, 0] \\
\vec{v_3} &= [0.595, 0, 0.706, 0.383]
\end{aligned}
$$

Exercise 6.16

$\vec{v} = \frac{\vec{V}}{|\vec{V}|}$ creates a unit vector $\vec{v}$ that points the same direction as $\vec{V}$ and has a unit of length (which equals 1).

Exercise 6.17

1

$$
\begin{aligned}
\vec{v_q} &= [1, 0, 1, 0] \\
score(q, 1) &= \frac{\vec{v_1}\vec{v_q}}{|\vec{v_1}||\vec{v_q}|} = \frac{0.897}{1.414} = 0.635 \\
score(q, 2) &= 0.487 \\
score(q, 3) &= 0.920
\end{aligned}
$$

Hence Doc 3 $>$ Doc 1 $>$ Doc 2

2

For a full query (car, auto, insurance, best) its idf vector is $[1.65, 2.08, 1.62, 1.5]$, so the Euclidean normalized one should be $[0.478, 0, 602, 0.469, 0.434]$.

$$
\begin{aligned}
\vec{v_q} &= [0.478, 0, 0.469, 0] \\
score(q, 1) &= 0.478 * 0.897 + 0.469 * 0 = 0.429 \\
score(q, 2) &= 0.324 \\
score(q, 3) &= 0.615
\end{aligned}
$$

Hence Doc 3 $>$ Doc 1 $>$ Doc 2