Information Retrieval & Text Analytics

Exercises week 3

Exercise 1

Are the following statements true or false?

a) In a Boolean retrieval system, stemming never lowers precision.

b) In a Boolean retrieval system, stemming never lowers recall.

c) Stemming increases the size of the vocabulary.

d) Stemming should be invoked at indexing time but not while processing a query.

Solution Exercise 1

a) False. Stemming can increase the retrieved set without increasing the number of relevant documents.

b) True. Stemming can only increase the retrieved set, which means increased or unchanged recall.

c) False. Stemming decreases the size of the vocabulary.

d) False. The same processing should be applied to documents and queries to ensure matching terms

Exercise 2

Given a biased coin with p(H)=0.25 and p(T)=0.75.

- Suppose we generate a series of symbols $s \in \{H, T\}$
- What is the theoretical minimum # bits per symbol required for a lossless compression?

Solution Exercise 2

Theoretical minimum of bits per symbol is defined by the entropy

$$H(P) = -\sum_{x \in X} P(x) \log P(x)$$

$$= -(0.25 \log_2 0.25 + 0.75 \log_2 0.75)$$

$$= 0.811278\ldots \; \texttt{bits}$$

Exercise 3

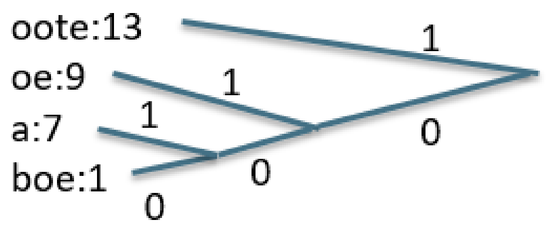
- Compress a fragment of Jan Hanlo’s poem “Oote” (1952)
- Use a word based version of Huffman coding
- Assume a lowercase version of the poem, ignoring whitespace
- Provide code table and compute compression ratio (one ASCII character is 7 bits)

OOTE

Oote oote oote
Boe
Oote oote
Oote oote oote boe
Oe oe
Oe oe oote oote oote
A
A a a
Oote a a a
Oote oe oe
Oe oe oe
<…>

Solution Exercise 3

- Frequency of occurrence in ascending order: boe:1, a:7, oe:9 oote:13
- Construct Huffman tree:



- Space for original version: (13*4+9*2+7*1+3*1)*7 = 581 bits
- Space for compressed version (excluding dictionary): 13*1+9*2+7*3+1*3= 55 bits
- Compression ratio (excluding dictionary): 55/581

Exercise 4

Consider the postings list (4, 10, 11, 12, 15, 62, 63, 265, 268, 270, 400, 444) with a corresponding list of gaps (4, 6, 1, 1, 3, 47, 1, 202, 3, 2, 130, 44). Assume that the length of the postings list is stored separately, so the system knows when a postings list is complete. Using variable byte encoding:

(i) What is the largest gap you can encode in 1 byte?

(ii) What is the largest gap you can encode in 4 bytes?

(iii) How many bytes will the above postings list require under this encoding? (Count only space for encoding the sequence of numbers.)

Solution Exercise 4

i. Variable byte encoding reserves one bit for determining continuation of final byte. With seven bits, the maximum gap we can encode is $2^7 - 1$ =127. Since gaps with zero length will never occur, we could theoretically encode 128 different gap lengths, but 7 bit representation equaling integer gap length is much more convenient.

ii. Similarly, four bytes yields $2^{28} - 1$

iii. 10*1+2*2 = 14 bytes for the gaps, one byte for the first docid. Total 15 bytes