Course Exam

Enrico Buratto

09/03/2022

Exercise 1

Task a

After normalizing by applying case folding and removing punctuation we get the following document collection (note: I only applied these two techniques, not stemming as it's not required by the exam text):

doc 1: the smallest planet is mercury

doc 2: jupiter the largest planet

doc 3: neptune is smaller than uranus but is heavier

doc 4: uranus is smaller than saturn

doc 5 neptune is heavier than uranus

doc 6: jupiter saturn uranus neptune

The term-dcument incidence matrix for the normalized collection is the following:

term	**doc 1**	**doc 2**	**doc 3**	**doc 4**	**doc 5**	**doc 6**
but	0	0	1	0	0	0
heavier	0	0	1	0	1	0
is	1	0	1	1	1	0
jupiter	0	1	0	0	0	1
largest	0	1	0	0	0	0
mercury	1	0	0	0	0	0
neptune	0	0	1	0	1	1
planet	1	1	0	0	0	0
saturn	0	0	0	1	0	1
smaller	0	0	1	1	0	0
smallest	1	0	0	0	0	0
than	0	0	1	1	1	0
the	1	1	0	0	0	0
uranus	0	0	1	1	1	1

Task b

The inverted indices I drew have the following notation:

`lexicon[doc frequency] -> [document ids] , [term frequencies]`

Where:

- `doc frequency` is the document frequency;

- `document ids` are the documents that contain the word;
- `term frequencies` are the term frequencies. The first term frequency refers to the first document id, the second to the second and so on.

```
but[1] -> [3], [1]
heavier[2] -> [3][5], [1][1]
is[4] -> [1][3][4][5], [1][2][1][1]
jupiter[2] -> [2][6], [1][1]
largest[1] -> [2], [1]
mercury[1] -> [1], [1]
neptune[3] -> [3][5][6], [1][1][1]
planet[2] -> [1][2], [1][1]
saturn[2] -> [4][6], [1][1]
smaller[2] -> [3][4], [1][1]
smallest[1] -> [1], [1]
than[3] -> [3][4][5], [1][1][1]
the[2] -> [1][2], [1][1]
uranus[4] -> [3][4][5][6], [1][1][1][1]
```

Task c

Both the queries will return an empty result:

- The first query is a conjunctive query, so a query like this would return the intersection between the sets of documents containing, respectively, the first and the second word. However, both sets are empty, so the intersection is an empty set;
- The second query should be executed in this order:
  - drug **OR** hope: disjunctive query, so it should be the union of the sets of document containing, respectively, the first and the second word;
  - **NOT** (drug **OR** hope): the result is the difference between all the sets of documents and the union set previously computed;
  - new **AND NOT** (drug **OR** hope): conjunctive query again. However, also in this query the terms are not present in the document collection, so the result is an empty set.

Exercise 2

Task a

- **9**: the binary representation for 9 is 1001; the offset is then 001 (the binary number without the leading bit). The selector size is 3 (the length of the offset); in unary code, this is 0001. The Elias-γ code for 9 is then **0001001**;
- **29**: the binary representation for 29 is 11101; the offset is then 1101. The selector size is 4; in unary code, this is 00001. The Elias-γ code for 9 is then **000011101**;
- **229**: the binary representation for 229 is 11100101; the offset is then 1100101. The selector size is 7; in unary code, this is 00000001. The Elias-γ code for 9 is then **000000011100101**.

Task b

- **9**: the binary representation for 9 is 1001. This fits 7 bit, so we can put the continuation bit $c = 0$. The VByte code for 9 is then **00001001**;
- **29**: the binary representation for 29 is 11101. This fits 7 bit, so we can put the continuation bit $c = 0$. The VByte code for 29 is then **00011101**;
- **229**: the binary representation for 229 is 11100101. This does not fit 7 bit, so we put the lower 8 bits in the first byte, we put the continuation bit $c = 1$ and we encode the remaining bit in the second byte (this time with $c = 0$). The VByte code for 229 is then **11100101 00000001**.

Exercise 3

Task a

We can compute the idf values for each term using the following formula:

$$idf_t = log_{10}\left(\frac{N}{df_t}\right)$$

where df_t is the document frequency of t, *i.e.* the number of documents that contain t. The idf values are, then:

- slow: $idf_t = log_{10}(906799/11816) = \mathbf{1.8850}$
- this: $idf_t = log_{10}(906799/606881) = \mathbf{0.1744}$
- bird: $idf_t = log_{10}(906799/1969) = \mathbf{2.6633}$
- down: $idf_t = log_{10}(906799/12523) = \mathbf{1.8598}$

The tf.idf weight of a term can be defined in different ways. For instance, in the slides by Manning and Raghavan it is defined as follows:

$$W_{t,d} = log_{10}(1 + tf_{t,d}) * log_{10}\left(\frac{N}{df_t}\right)$$

where $tf_{t,d}$ is the term frequency of t in document d. The **tf.idf weights** using this formula are, then:

- **Doc 1**
  - slow: $W_{t,d} = log_{10}(1+5) * 1.8850 = 1.4668$
  - this: $W_{t,d} = log_{10}(1+30) * 0.1744 = 0.2601$
  - bird: $W_{t,d} = log_{10}(1+0) * 2.6633 = 0$
  - down: $W_{t,d} = log_{10}(1+15) * 1.8598 = 2.2394$
- **Doc 2**
  - slow: $W_{t,d} = log_{10}(1+17) * 1.8850 = 2.3662$
  - this: $W_{t,d} = log_{10}(1+33) * 0.1744 = 0.2671$
  - bird: $W_{t,d} = log_{10}(1+6) * 2.6633 = 2.2507$
  - down: $W_{t,d} = log_{10}(1+33) * 1.8598 = 2.8482$
- **Doc 3**
  - slow: $W_{t,d} = log_{10}(1+19) * 1.8850 = 2.4524$
  - this: $W_{t,d} = log_{10}(1+67) * 0.1744 = 0.3196$
  - bird: $W_{t,d} = log_{10}(1+9) * 2.6633 = 2.6633$
  - down: $W_{t,d} = log_{10}(1+24) * 1.8598 = 2.5999$

However, we can also consider the term frequency not logaritmically scaled, as also the book does; in this case, the td.idf weights are:

- **Doc 1**
  - slow: $W_{t,d} = 5 * 1.8850 = 9.425$
  - this: $W_{t,d} = 30 * 0.1744 = 5.232$
  - bird: $W_{t,d} = 0 * 2.6633 = 0$
  - down: $W_{t,d} = 15 * 1.8598 = 27.897$
- **Doc 2**
  - slow: $W_{t,d} = 17 * 1.8850 = 32.045$
  - this: $W_{t,d} = 33 * 0.1744 = 5.7552$
  - bird: $W_{t,d} = 6 * 2.6633 = 15.9798$
  - down: $W_{t,d} = 33 * 1.8598 = 61.3734$
- **Doc 3**
  - slow: $W_{t,d} = 19 * 1.8850 = 35.815$
  - this: $W_{t,d} = 68 * 0.1744 = 11.8592$
  - bird: $W_{t,d} = 10 * 2.6633 = 26.633$
  - down: $W_{t,d} = 24 * 1.8598 = 44.6352$

Task b

In order to calculate the normalized euclidean document vectors we just need to calculate the L_2 norm, which is defined as:

$$||\bar{x}||_2 = \sqrt{\sum_i x_i^2}$$

and divide every tf.idf score by it. The vectors are different if we use the logarithmic scaling; for this reason, here are calculated only the normal ones (not scaled). However, the procedure is the same.

The L_2 norms are:

- Doc 1: $\sqrt{9.425^2 + 5.232^2 + 0^2 + 27.897^2} = 29.9073$
- Doc 2: $\sqrt{32.045^2 + 5.7552^2 + 15.9798^2 + 61.3734^2} = 71.2885$
- Doc 3: $\sqrt{35.815^2 + 11.8592^2 + 26.633^2 + 44.6352^2} = 64.2259$

And so the euclidean normalized vectors are:

- Doc 1: $(\mathbf{0.3151}, \mathbf{0.1749}, \mathbf{0}, \mathbf{0.9328})$
- Doc 2: $(\mathbf{0.4495}, \mathbf{0.0807}, \mathbf{0.2242}, \mathbf{0.8609})$
- Doc 3: $(\mathbf{0.5576}, \mathbf{0.1846}, \mathbf{0.4147}, \mathbf{0.695})$

Exercise 4

Task a

A/B testing is a research methodology that is usually used for testing and evaluating the experience that users have with a system. The underlying idea of this methodology is that two versions of the system, A and B precisely, are compared using a randomized experiment. These two systems are usually almost identical, with the exception of one variation (*e.g.* one extra feature, the same feature with different implementation, different font types, etc...) that might affect a user's behaviour. *Randomized experiment*, moreover, means that two different groups are made from a population; these two groups should be the same size, and one group is evaluated on the first system and the other is evaluated on the second system.

I would not entirely suggest to use this evaluation method, unless some requirements are respected: in fact, using A/B testing to evaluate a complete different system would most probably lead to not-interpretable results due to the high amount of changes, since this methodology is design to study atomic changes to the system. However, A/B testing could be used in an "incremental" fashion: the evaluation could be made passing progressively from the old to the new system adding or modifying only a feature at a time and studying the behaviour of the users on that particular modification.

Task b

MAP, acronym for Mean Average Precision, is an evaluation measure used to compare two or more systems. Instead of taking into consideration precision and recall only as stand-alone measures, one can calculate MAP to have a single value to compare: this value is an average precision value across all recall levels, which means that we calculate the average of the precisions of queries for every recall value. We can then plot these results into a graph or we can display this information in a table; the main advantage of using MAP is then that we have easier to intepret results.

NDCG, acronym for Normalized Discounted Cumulative Gain (Normalized DCG) is another evaluation methods that uses graded relevance as a measure of usefulness from examining a document (retrieved by a query). This usefulness is the gain, and it is accumulated starting at the top of the ranking and may be reduced at lower ranks; however, since comparing two query may lead to misleading conclusion, the cumulative gain is normalized, and this is useful for contrasting queries with different numbers of relevant results.

For this system in particular, I would suggest to use NDCG. This is, in fact, a powerful measure that should be used in order to compare the effectiveness of the different systems that will be developed and tested given the requirement of finding precise information in technical manuals quickly. NDCG brings, however, some issues: it does not penalize a system for bad documents in the result list and for missing documents. Despite this, I would say that if information should be found quickly some sort of ranking should be taken into consideration; therefore, NDCG could be a suitable measure.

Task c

I would say that the system specialist is almost wrong about the population and almost right about the statistical testing. The best chosen population, in fact, should be the affected users: since the system will be mostly used by the engineers, the company secretaries are not a suitable population (given that the secretaries are not also the engineers which will use the system). However, he/she is right about statistical testing: the standard procedure for this is, in fact, exactly to collect information with suitable evaluation methods and measures, interpret them and then make a "launch/no-launch" decision based on the results, which is exactly what he/she described.

Exercise 5

Several strategies could be implemented in order to recude the sample size, while maintaining statistcal relevance.

A first strategy could be to increase the statistical significance threshold: a higher threshold leads to a smaller sample size, therefore increasing the relevance threshold to, e.g., 0.1 dramatically decrease the sample size. However, doing this means that the statistical relevance naturally drops, so this should be an *in extremis* strategy.

Another strategy could be to reduce the statistical power, *i.e.* the "power" of rejection of the null hypothesis when is should be rejected. Also in this case, though, statistcal relevance drops.

The best strategy in my opinion is to use paried tests instead of independent tests; if the ecommerce company manages to do this, the sample size would be exactly cut in half, achieving the goal.