Intelligent Systems – Natural Language Processing

Topic Modelling

Teresa Gómez-Carpintero García (t.gomez-carpinterog@alumnos.upm.es)

1. Problem to solve

The objective of this project is to create a basic system that predicts the topic of a given document. To do this, we have a set of files classified in the following categories: computer science, history, mathematics (maths), biology, geography, accounts and physics. So, the goal is to classify the given text in one of those categories.

Also, some metrics will be computed in order to evaluate the system.

2. Methodology

To develop this project, it has been used R and Rstudio. In order to aim the objective described before, the following steps were done:

1. Explore each corpus (there are 7 corpora, one for each category)
2. Analyze transformations
3. Slip in train and test
4. Train the system
5. Evaluate the system
6. Analyze the results

2.1. Explore each corpus

	Number of documents	Number of terms	Number of terms with frequency 1
Computer science	642	23,060	13,264
History	500	40,200	22,521
Mathematics	214	4,914	2,685
Accounts	284	6,621	3,478
Biology	635	24,798	13,115
Geography	98	6,967	4,149
Physics	769	28,793	16,830

Table 1. First summary of each corpus

The first observation of this corpora is that the number of documents of each category is different, and the difference goes from 70 documents (mathematics-accounts) to 671 (geography-physics). We will see later that these differences will be taken into account when developing the system.

One key factor of this project is based on the Term Document Matrix (tdm), where we can observe the relationship between terms in a corpus and the documents of that corpus. In this project, there are seven tdm as we have seven corpora. These matrices will be used to predict the category of a document by looking at the most frequent words that appear in a corpus and the most frequent words in the document. But in order to obtain which terms characterize a corpus, some transformations must be applied to the corpora. These transformations could be removing numbers, removing punctuation, removing some words, stemming document, stripping whitespace.

It is needed to do an exploration of each corpus in order to know which transformation should be applied. By looking at the documents of each corpus we can see that maybe it is relevant to keep numbers in the corpus of computer science or keeping some punctuation symbols like (+) in the corpus of mathematics.

2.2. Analyze transformations

To analyze if numbers should be kept, we look at the hundred most frequent terms of each corpus and if there are no numbers in that rank, we can consider that they are not so relevant (they do not characterize the corpus), so we can remove them. The same logic is applied to know if punctuation symbols should be kept.

Now, we will analyze the problems that were found in some transformations.

- Remove punctuation: when looking at the hundred most frequent terms of each corpus, the symbol (-) was found in mathematics, biology, geography and physics. As this symbol appears in four of the seven corpora, it does not characterize the corpus, so it can be removed.

- Remove words: stop words are deleted in every corpus. The only corpus that has another word to be removed is mathematics, because "www.ebook3000.com" appears in the hundred most frequent terms. The reason to delete this term is that since it appears always at the end of documents and it can be the source of the files, it can appear also in documents from other categories, not only maths.

- Remove numbers: the only corpus that has number in his hundred most frequent term is accounts. As we will see when analyzing results, the accuracy of this category is very high when numbers are deleted.

In conclusion, all the transformations will be applied to all the corpora.

2.3. Slip in train and test

To develop the system, it has been necessary to split each corpus in a training part, with the 70% of the documents, and a test part, with the 30% of the documents.

2.4. Train the system

The goal of this part is to obtain a weigh matrix where the terms that high frequency only in one corpus are highlighted. This means that a term that only appears in one corpus will have more weight than one that appears in four corpora.

This idea is based on TF-IDF weighting, but the formulas are a little bit different.

$$w_{i,j} = tf_{i,j} \cdot idf_j$$

$tf_{i,j}$: number of occurrences of the term *i* in the corpus *j*.

Notice that it will not matter if term *i* appears 1 or 15 times in a document. As the size of each corpus differs, every column (term frequency of a corpus) is scaled by taking into account its size.

$$idf_j = \log\left(\frac{number\ of\ corpora\ (7)}{number\ of\ corpora\ where\ appears\ term\ i}\right)$$

Once we have calculated the $tf_{i,j}$ matrix, it is reduced by deleting terms that has a frequency lower than 0.25, because those terms are not relevant.

	Number of terms after reducing the tdm
Computer science	55
History	152
Maths	31
Accounts	42
Biology	33
Geography	49
Physics	65

Table 2. Number of terms per category after the reduction of tdm

After obtaining $w_{i,j}$, we can start the prediction of the category of a given document. The logic behind the prediction consists on:

1. Calculate the term document matrix of the given document. This matrix has only one column because we only have one document, the one that we want to predict its category.

2. Which category fits better that document? Remember that we have a column in $w_{i,j}$ per category, so we have the weights of each category.

3. Compute the sum of the category weighted column times the corresponding element of the tdm of step 1. If *m = tdm of step 1,* then for each category *j*:

$$cs_j = \sum_{i\ is\ term\ of\ m} i \cdot w_{ij}$$

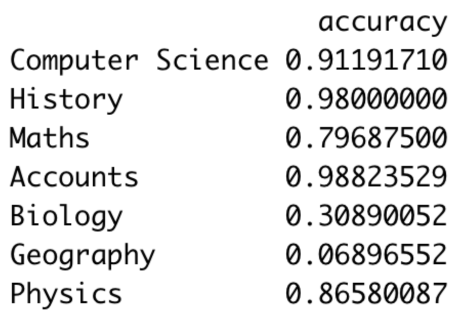
4. The category of the given document will be the one that has the maximum cs_j obtained in step 3.

2.5. Evaluate the system

Each category has been evaluated individually by calculating its accuracy.

$$accuracy = \frac{number\ of\ corrected\ predictions}{number\ of\ total\ predictions}$$

3. Analysis of results



```
                     accuracy
Computer Science 0.91191710
History          0.98000000
Maths            0.79687500
Accounts         0.98823529
Biology          0.30890052
Geography        0.06896552
Physics          0.86580087
```

Figure 1. Accuracy of each category. From better to worse: accounts, history, computer science, physics, maths, biology and geography

As we can see in figure 1, the system predicts very good documents from computer science, history, mathematics, accounts and physics corpora. But, if we look at biology and geography, we have a very low accuracy.

In order to detect the problem with these categories, we can see what the system is predicting instead of the correct category. Notice that the following explanations are referred only to the documents that have been not predicted correctly.

- Geography: the system always predicts these documents as history except one that is classified as computer science. If we look at table 2 (section 2.4) we can see that history is the one with most terms (152) while geography has 49 terms. Also, if we take a look at the hundred most frequent terms of each category (analysis done in section 2.1 and 2.2), we can see that the terms of geography appear also other categories like biology. So as $w_{i,j}$ penalize those terms that are not unique (terms that are frequent in more than one corpus), the terms of geography are being penalized. And that is why the system cannot predict correctly this category. Its performance could be better with more documents of geography.

- Biology: most of the times the system predicts the documents as history, but also physics, geography, and computer science. The reason of the bad accuracy in biology is the same as in geography.

4. Link to the Github Repository

https://github.com/TeresaGCG/IntelligentSystems-NLP

5. Bibliography

- García-Castro, Raúl. Intelligent Systems. Hands-on 1.RPubs. Retrieved from https://rpubs.com/rgcmme/IS-HO1
- García-Castro, Raúl. Intelligent Systems. Hands-on 1.RPubs. Retrieved from https://rpubs.com/rgcmme/IS-HO3
- Deepak (2018-2020). 7 topic data for text classification. Kaggle. Retrieved from https://www.kaggle.com/deepak711/4-subject-data-text-classification
- termFreq function | R Documentation. Retrieved from https://www.rdocumentation.org/packages/tm/versions/0.7-8/topics/termFreq