UNIVERSIDAD
POLITÉCNICA
DE MADRID

Adaptive Systems

Practical Assignment 1: TF-IDF vs LDA - Text Analysis Comparison

Fabbri Lucia, Marza Mathilde

October 2023

Given this data set including news, the assignment consists of the following tasks:

1. Implement the following pseudocode to calculate the variable *ratio_quality* using the TF IDF vectors:

```
total_goods = 0
For every article (a) on topic "Food and Drink":
    Obtain the top-10 most similar articles (top-10) in Corpus to a
    Count how many articles in top-10 are related to topic "Food and Drink" (goods)ê
    total_goods = total_goods + goods
ratio_quality = total_goods/(num_articles_food_and_drink*10)
```

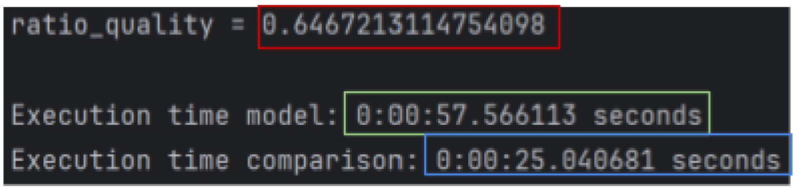
And measure the execution times separately for the following two subprocesses:

- **Creating the model (from the program begin to the call** `similarities.MatrixSimilarity(tfidf_vectors)`**)**
- **Implementation of the pseudocode above.**

This first part shows the results obtained performing both task1 and task2, using respectively TF-IDF and LDA vectors for finding the top-10 most similar articles related to the topic “Food & Drinks”, adhering to all the constraints imposed by the exercise.
Before applying the models, we have performed some preprocessing operations, such as removing stopwords, in order to have a cleaner corpus as much as possible. Despite all the preprocessing operations, we have noticed that some punctuations, marks and emojis are still in our descriptions corpus.

TF-IDF results for ‘Food & Drink’ category

The results obtained while using TF-IDF for the ‘Food & Drink’ category are quite satisfying since the ratio_quality value we have obtained is 0,647 as shown in Figure 1. This means that 65% of the 10 most similar articles about Food and Drink are also part of this topic and because we have decided to not customize our TF-IDF model, we think that those ones are the best results we could obtain while performing this task with the provided data.

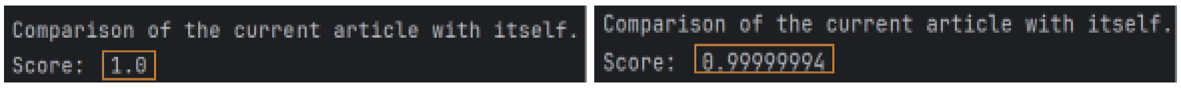


```
ratio_quality = 0.6467213114754098

Execution time model: 0:00:57.566113 seconds
Execution time comparison: 0:00:25.040681 seconds
```

Figure 1: TF-IDF results for ‘Food & Drink’ category

As far as it concerns the execution times of the task still shown in Figure 1, the process to build the model takes less than 1 minute while the comparison process takes only 25 seconds. This means that the whole process is performed quite quickly and so we are delighted with the results.

During the process of comparison, we have chosen to leave the current element in the list of elements to which it is compared. Another option could have been to remove this element since it wouldn't provide any additional information, but we thought it would be interesting to see how TF-IDF compared an item to itself. The score we obtained was 1 or sometimes 0,99 as shown in Figure 2: these results are very satisfying since they match all our expectations.



```
Comparison of the current article with itself.
Score:  1.0
```

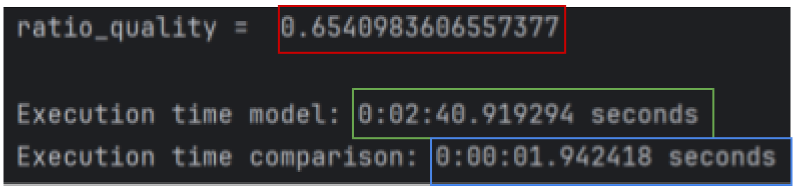
```
Comparison of the current article with itself.
Score:  0.99999994
```

Figure 2: TF-IDF score while comparing an article with itself

2. Repeat the previous task with LDA vectors and compare the results. Explain the differences in the results. Use 30 topics, two passes, and a random state parameter.

LDA results for 'Food & Drink' category

The results obtained while using LDA vectors for the 'Food & Drink' category are even more satisfying and interesting compared to the previous ones. As shown in the following Figure 3, the ratio_quality value we have obtained in this case is 0.654, very slightly higher than the TF-IDF score despite the fact that we have applied the same preprocessing operations for both tasks.

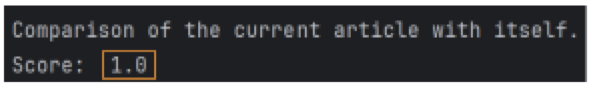


```
ratio_quality =  0.6540983606557377

Execution time model: 0:02:40.919294 seconds
Execution time comparison: 0:00:01.942418 seconds
```

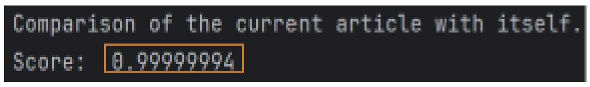
Figure 3: LDA results for 'Food & Drink' category

Unlike TF-IDF, the execution time for building the LDA model is 2 minutes 40 seconds. This can be explained by the fact that the LDA algorithm is more complex than the TF-IDF weighting calculation. However, the comparison time is instantaneous, taking just under 2 seconds, as shown in Figure 3.

Following the same reasoning mentioned above for the TF-IDF vector, the score obtained using LDA while trying to compare an article with itself is still 1 or sometimes 0.99, as shown in Figure 4. This is further proof of the correct execution of the comparison.



```
Comparison of the current article with itself.
Score:  1.0
```



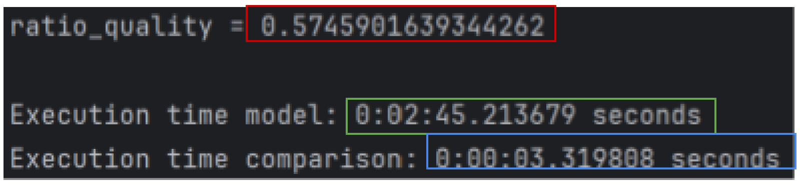
```
Comparison of the current article with itself.
Score:  0.99999994
```

Figure 4: LDA score while comparing an article with itself in 'Food & Drink' category

TF-IDF and LDA results comparison for 'Food & Drink' category

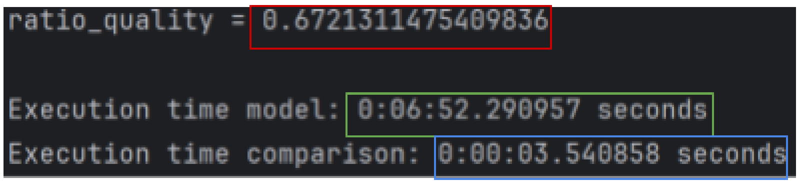
To resume our results, we can say that both vectors are able to give good and similar performance. In particular, we have noticed a slight improvement while using LDA vector instead of TF-IDF, maybe due to the fact that LDA has hyperparameter tuning. Indeed, the latter approach offers the opportunity to adjust its parameters to improve its performance. As mentioned in task2 constraints, we have applied the fine-tuning of the parameters using 30 topics, 2 passes, and a random state parameter with a value of 30. Considering that these parameters are well chosen, we can say that this is the reason why LDA has obtained better scores than TF-IDF, and that by modifying them, this trend can be intensified, or even reversed. Let's take a look at the different parameters:

- **Passes**: this parameter indicates how many times the algorithm should go through the corpus during training. Logically, we expect the score to improve as it increases.
  In order to confirm that, we have run our code changing some parameters and observe the way it worked. Setting the parameter "*passes = 1*", makes the algorithm pass only once through our corpus during the training and the results obtained are worse than TF-IDF, because we obtained the final score equal to 0.574, as shown in Figure 5. In contrast, setting the parameter with a higher number of passes, such as "*passes = 5*", leads to better improvement in the performance of LDA quality score, reaching 0.672, as shown in Figure 6.



```
ratio_quality = 0.5745901639344262

Execution time model: 0:02:45.213679 seconds
Execution time comparison: 0:00:03.319808 seconds
```

Figure 5: LDA results for 'Food & Drink' category with passes = 1



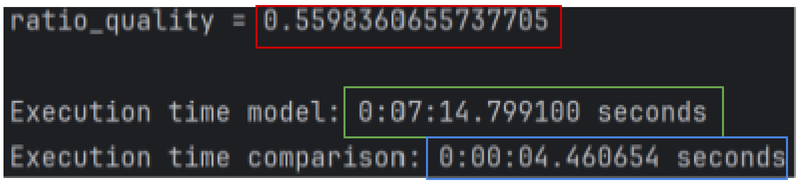
```
ratio_quality = 0.6721311475409836

Execution time model: 0:06:52.290957 seconds
Execution time comparison: 0:00:03.540858 seconds
```

Figure 6: LDA results for 'Food & Drink' category with passes = 5

One important thing to notice is that as we improve the number of passes, the execution time for both the model and the comparison increases. The best approach would be to find the right number of passes in order to have a trade-off between the ratio_quality and the execution time of the whole process. In our case, the number of passes was already given and it was "*passes = 2*".

- **Number of topics / data Dimensionality**: TF-IDF and LDA methods behave differently while creating data matrices. TF-IDF generates a high-dimensional matrix by treating each word as a separate feature while LDA aims to reduce this dimensionality by focusing on a limited number of relevant topics. Since in this latter case we only focused on 30 topics, we had in some way simplified the analysis and this had led to better results.

  As we did previously, in order to prove what we said above, we have kept the number of passes "*passes = 2*" and then increased the number of topics to 100, by setting "num_topics = 100". The score obtained in Figure 7 shows that performance is worse with a higher number of topics: the execution time has risen to more than 7 minutes and the ratio_quality is lower compared to the previous ones.



```
ratio_quality = 0.5598360655737705

Execution time model: 0:07:14.799100 seconds
Execution time comparison: 0:00:04.460654 seconds
```

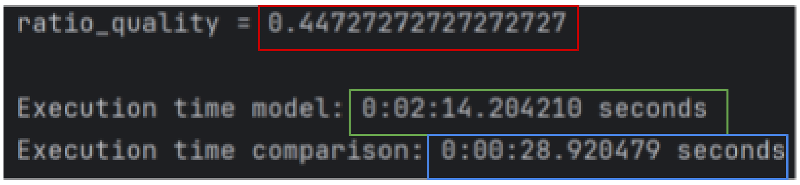
Figure 7: LDA results for 'Food & Drink' category with num_topics = 100

3. Repeat the previous two tasks but with the topic "Sports" and compare the results. Why do you think that the quality of the results is worse than the ones obtained with the topic "Food and Drink"?

Here is the analysis of the results obtained performing task3, using respectively TF-IDF and LDA vectors adhering to all the constraints imposed by the exercise, in order to find the top-10 most similar articles. This time, we're going to observe all the articles related to the topic "Sports".

TF-IDF results for ‘Sports’ category

The results obtained using TF-IDF for the 'Sports' category are less satisfactory than for the 'Food & Drink' category since the ratio_quality value we obtained is 0.447 as shown in Figure 8. This means that 45% of the 10 most similar articles on food and drink are also part of this theme.

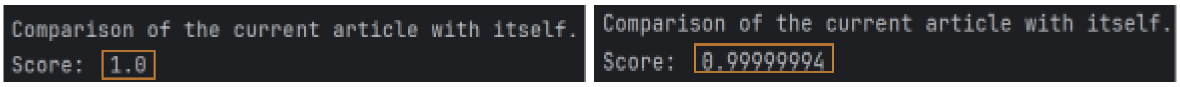


```
ratio_quality = 0.4472727272727272727

Execution time model: 0:02:14.204210 seconds
Execution time comparison: 0:00:28.920479 seconds
```

Figure 8: TF-IDF results for ‘Sports’ category

As far as it concerns the execution times of the task still shown in Figure 8, the process to build the model takes less than 3 minutes while the comparison process takes 29 seconds. These two times are still reasonable.

Following the same reasoning mentioned above for the TF-IDF and LDA algorithms, the score obtained while trying to compare an article with itself is still 1 or sometimes 0.99, as shown in Figure 9. This is further proof of the correct execution of the comparison.



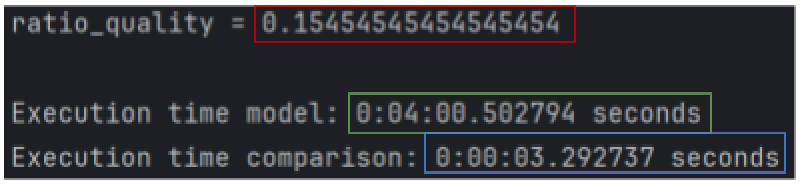
```
Comparison of the current article with itself.
Score:  1.0
```

```
Comparison of the current article with itself.
Score:  0.99999994
```

Figure 9: TF-IDF score while comparing an article with itself in ‘Sports’ category

LDA results for ‘Sports’ category

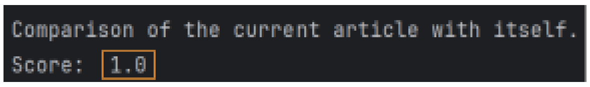
The results obtained using LDA vectors for the "Sports" category are worse than TF-IDF for the "Sports" category and worse than LDA for the "Food & Drink" category. As shown in the following Figure 10, the ratio_quality value we have obtained in this case is 0.155, which is quite low.



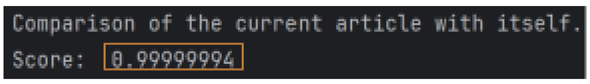
```
ratio_quality = 0.15454545454545454

Execution time model: 0:04:00.502794 seconds
Execution time comparison: 0:00:03.292737 seconds
```

Figure 7: LDA results for ‘Sports’ category

Following the same reasoning mentioned above for the TF-IDF and LDA algorithms, the score obtained while trying to compare an article with itself is still 1 or sometimes 0.99, as shown in Figure 11. This is further proof of the correct execution of the comparison.



```
Comparison of the current article with itself.
Score:  1.0
```



```
Comparison of the current article with itself.
Score:  0.99999994
```

Figure 11: LDA score while comparing an article with itself in ‘Sports’ category

TF-IDF and LDA results comparison between ‘Food&Drink’ and ‘Sports’

The quality of the results is worse than the ones obtained with the topic "Food and Drink for both TF-IDF and LDA algorithms. In our opinion, the reason for the drop in scores lies in the diversity of the data.

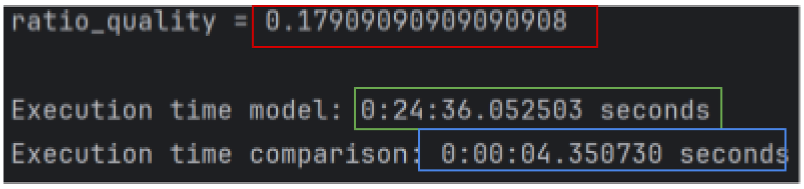
Indeed, the 'Sports' category is more diverse with more complex subjects such as 'game strategies' and 'athlete performance', making the task of detecting subjects more complex. As example, while reading the following article, it's easy to think it belongs to the "Politics" category, but actually it belongs to the "Sports" category:

“In a speech from the Oval Office on Sunday night, President Barack Obama urged Americans to avoid discriminating against Muslims in the wake of last week's shooting in San Bernardino, California. "Muslim-Americans are our friends and our neighbors; our co-workers, our sports heroes -- and, yes, they are our men and women in uniform," Obama said. That drew a response from business mogul and Republican presidential candidate Donald Trump, who questioned the existence of Muslim athletes on Twitter”

But above all, we can highlight a very poor score for LDA. Our explanation for this phenomena is that LDA relies on the distribution of words within documents to identify topics. For this reason, in cases where words related to different topics co-occur, it can struggle to assign a clear topic.

One way to overcome this problem could be increasing the passes attribute, allowing our model to go through the corpus more than once during the training phase. Applying this technique would provide benefits and help make this task easier, though it may not completely resolve it.

To check this, we ran an experiment with ‘*passes = 10*’. The result was indeed better, with a score of 0.179 as shown in Figure 12, but as we might have expected, this hasn’t solved the problem as the score obtained is still very low.



```
ratio_quality = 0.17909090909090908


Execution time model: 0:24:36.052503 seconds
Execution time comparison: 0:00:04.350730 seconds
```

Figure 12: LDA results for ‘Sports’ category with passes = 10

Results summary table

In the following Table 1, we list all the result we have obtained while applying TF-IDF and LDA model to our input data for both ‘Food & Drink’ and ‘Sports’ categories:

	Food & Drink		Sports	
	TF-IDF	*LDA*	*TF-IDF*	*LDA*
Ratio quality	0,647	0.654	0.447	0.155
Execution time model	0.57	2.40	2.43	4.00
Execution time comparison	0.25	1.94	0.43	3.29

Table 1: Summary table with all the results obtained

4. Explain how you can get better results for the previous tasks by resorting to tagging. Note that the articles in the data set are tagged.

Using the tags associated with articles in the corpus can really make the results better. Tags are like labels that help us understand what the articles are about, providing valuable information about the content and topics of the articles. While using tags, we can help the recommender systems to figure out what articles are similar and which ones are more relevant, providing better suggestions to users.

Here are listed the two ideas through which we could improve our tasks and obtain better results using tags, in combination with TF-IDF and LDA vectors:

- **Pre-processing with tags:** we can consider making it a supervised method by adding labels to each document before training. This means the new model takes into account what the documents are about before it learns. In our specific case, tags are considered to be the labels.
- **Post-Processing with tags**: after generating recommendations using TF-IDF or LDA models, apply tags for post-processing. Filter the results to guarantee that recommended articles align with matching tags, delivering more user-relevant content.

It is important to note that the effectiveness of the use of tags will also depend on the quality of the tags assigned to the articles in your corpus of data. So we should also make sure that the tags are relevant and accurate to get the best results:

- **Tag Effectiveness**: before using models, it's important to sort out articles that don't have the right tags. This step makes your dataset better by focusing on content related to the tags you're interested in.
- **Topic Refinement**: tags are like labels that help you pick specific topics. You can use them to find articles that match these labels and create a smaller, more focused dataset. This works well with methods like TF-IDF and LDA for digging into specific subjects.
- **Tag Relevance**: when checking how good your results are, think about how well the tags match what you're looking for. You might want to pay more attention to articles that have the same tags. This can make methods like TF-IDF and LDA give you better recommendations related to your chosen tags or topics.