Mireia Pou Oliveras 251725
Iria Quintero García 254373
Javier González Otero 243078

Part 2: Indexing and Evaluation

Introduction

The objective of this project is to develop a search engine.

After preprocessing the data and making an exploratory data analysis, in this second part we are dealing with indexing, ranking and evaluation tasks. Our second part of this project is structured in the following way:

- Part 1: Indexing

In Part 1, we create an inverted index, propose queries and then add ranking to obtain the top K most relevant results with tf-idf scoring.

- Part 2: Evaluation

In Part 2, we deal with evaluating the part 1 algorithm with different techniques and queries. Additionally, we make a one vector representation to represent the tweets in a two-dimensional scatter plot through the T-SNE.

1. Indexing

This part is structured in the following way:

- 1.1 Inverted index
- 1.2 Querying
- 1.3 Ranking with TF-IDF

1.1. Inverted index

As stated, the first part of the Indexing task consists on creating a basic inverted index structure. We reused the function from practice 1, modifying the way to access the data. To build the index,we simply added each document that contains a given term to the list for that term.

The creation of it took 0.56 seconds.

Mireia Pou Oliveras 251725
Iria Quintero García 254373
Javier González Otero 243078

1.2. Querying

To search a query in the inverted index, we created a function that searches for each term of the query, the documents that include it in the inverted index. Then we made the intersection of the term lists (**AND operation**), to get only the documents that include ALL the terms in the query.

We have proposed the following set of queries to retrieve information from the collection:

1. **India government farmers protest:** It returns many results since it contains the principal words of the database topic, but most of them include generic statements about some of the words of the query, as Indian institutions, without including policy discussions.

   Information need reflected: What has the Indian government said or done in relation to farmers' protests?

   #results: 62

2. **Increasing prices:** It may be one of the concerns in Farmers Protests, but the results often missed direct references to price issues affecting farmers, with results loosely related to the topic.

   Information need: Have farmers' protests increased prices of goods?

   #results: 109

3. **Narendra Modi** (India's prime minister): India's prime minister could probably be searched in this context. Most of the results do refer to him, talking about his actions and their impact on society.

   Information need: What do people think about the prime minister's actions?

   #results: 70

4. **Disha Ravi** (Principal activist for farmer's rights): In this case, since it is a very popular person name, most of the time Ravi followed Disha, therefore, most of the documents gathered made sense.

   Information need: What does Disha Ravi states about the protests?

   #results: 628

5. **Anime:** Has nothing to do with farmers protests therefore, few results should be gathered. However, due to stemming, the word anime is set to 'anim', leading to results related to 'animals'. It could be interesting to use another stemming method.

   Information need: A person that doesn't understand that this is not the proper database for his/her information need.

   #results: 31

Mireia Pou Oliveras 251725
Iria Quintero García 254373
Javier González Otero 243078

1.3. Ranking with tf-idf and cosine similarity

To build the ranking, we created an inverted index by organizing documents based on cosine similarity scores computed with TF-IDF weighting. Again, we reused the function from Practice 1, modifying how the data is accessed. The function *create_index_tfidf* returns the index and the three next dictionaries: tf, df, idf.

Creating the index took us 12s approximately.

For ranking, we adapted the *rank_documents* function to produce a ranking of documents and to return this ranking and each document's predicted relevance score. The ranked list of the documents contains the most relevant documents to each query.

We tried this ranking for the same queries from the previous part and got the same number of total results. Looking at the top 10 ranked we noticed the following things:

- **India government farmers protest:** The top-ranked documents are more targeted, directly discussing government actions and their impact on the protests, as the first results were frequently including generic statements related to Indian institutions or other things.
- **Increasing prices:** The results for this query also show a significant improvement in relevance, emphasizing tweets on commodity price hikes, fuel costs and government inaction, aligning closely with the intended query focus on economic concerns related with the protest.
- **Narendra Modi:** These results kept showing interesting information about the prime minister, putting more focus on the users' opinion about him and his social impact.
- **Disha Ravi:** Tweets with deeper focus on Ravi's activism and her symbolic role in the protest were retrieved, which shows a better performance.
- **Anime:** The results were still bad as the keyword ambiguity persists.

With this, we can conclude that TF-IDF weighting captures relevance based on term frequency but can occasionally retrieve unrelated documents when the terms are too general or require contextual understanding.

2. Evaluation

This part is structured in the following way:

- 2.1 Evaluation Functions
- 2.2 Provided queries evaluation
- 2.3 Evaluation for proposed 5 queries
- 2.4 T-SNE

Mireia Pou Oliveras 251725
Iria Quintero García 254373
Javier González Otero 243078

2.1. Evaluation functions

The evaluation process uses multiple metrics to assess retrieval effectiveness, including Precision@K, Recall@K, F1-Score@K, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG).

In this first part, we include functions to compute each of these metrics.

The functions of *precision_at_k*, *average_precision_at_k, rr_at_k, dcg_at_k*, and *ndcg_at_k* are reused from Practice 2. We only made one change in the part that counts relevant documents: instead of only counting those from the ground_truth, we also check if the predicted_score is greater than 0. That way, we can distinguish between retrieved and non retrieved documents by our retrieval system.

The functions of *recall_at_k* and *f1_score_at_k* were created from scratch. We simply copied the structure from the other functions and adapted the formulas to be the correct ones for these metrics.

Finally, we also made from scratch the functions of *mean_average_precision_at_k* and *mean_reciprocal_rank_at_k*. These functions take two lists with "n" sub-lists (where n is the number of queries) and calculate the average of the average precision and reciprocal rank for those queries.

All these functions are then used in two overall functions we created to evaluate the queries:

1. **evaluate_query(query, query_id, ground_truth, index, k_array, own_query)**

This function evaluates a single query by comparing the predicted relevance of the documents to the ground truth.

- First, we filter the ground_truth to include only the rows corresponding to the query_id. We use a parameter called own_query as a flag because the ground truth when we are the judges is already composed of integers, whereas the ground truth proposed by the teachers is composed of strings.
- Next, we use the search_tf_idf function to generate the rankings and relevance scores. We normalize the relevance scores using min-max scaling. To avoid the last document having a zero score, we take the average of the second last score with zero.
- Then, we merge the ground_truth with the predicted labels and relevance scores into one dataframe called ground_truth_complete.
- Finally, we compute the evaluation metrics (excluding MAP and MRR).

2. **evaluate_map_mrr(ground_truth, query_list, query_id_list, k_array, own_query)**

This function computes the Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) for each query in the query_list.

- First, we loop through each query and query_id in query_list and query_id_list. Inside the loop, we repeat the same steps as before: filter the ground truth, obtain the rankings and relevance scores, and merge everything into one dataframe.
- After that, we append the ground truth and the predicted relevance scores of each query to the all_ground_truth and all_predicted lists, which contain the results for all queries.
- Finally, we calculate MAP and MRR using their respective functions.

Mireia Pou Oliveras 251725
Iria Quintero García 254373
Javier González Otero 243078

2.2. Provided queries evaluations

In this part, we evaluate the two queries proposed by the teachers:

1. query_1 = "What are the people's rights?"
2. query_2 = "What is being said about the Indian government?"

The results obtained can be seen in the Python document.

We observed that in the second query, the results were significantly penalized by the word “said,” which was considered after doing the preprocessing of the query. Because of this, none of the documents in the ground truth had any predicted relevance, thus all the evaluation metrics were zero.

To achieve a better and more meaningful result, we removed the word “said” and then performed the evaluation again with the new query. We have attached two photos that confirm how much the penalization caused by the word “said” affected the results.

```
what is being said about the Indian government
['said', 'indian', 'govern']

======================
Top 10 results out of 3 ranked documents for the searched query:
```

```
indian government
['indian', 'govern']

======================
Top 10 results out of 529 ranked documents for the searched query:
```

Independently of the penalization, we also believe that this word did not contribute any new information to our query.

2.3. Evaluation for proposed 5 queries

In this part we evaluate the 5 queries proposed by us, which can be found in part 1.2. To do it, we simply called the functions created in the evaluation functions part with the parameter own_query = True.

Comment on each of the evaluation techniques:

1. **Precision@K:** This metric gives us the quality of the retrieved documents. High precision indicates that the model returns lots of relevant documents, but it does not consider how many relevant documents are missing.

2. **Recall@K:** This metric indicates how many relevant documents we retrieved out of all the relevant ones. A high recall means that the model retrieves a large proportion of relevant documents, but it might also include more irrelevant documents.+
3. **Average Precision@K**: This metric summarizes a precision-recall (PR) curve by averaging precision values at each relevant document up to rank K. A high score suggests that relevant documents are retrieved early and consistently, while a low score indicates the presence of irrelevant documents throughout the retrieval.
4. **F1-Score@K**: This metric combines precision and recall into a single score, balancing the model's ability to retrieve relevant documents (recall) with its ability to avoid irrelevant ones (precision). A high F1 score indicates strong performance in both aspects, while a low score suggests weaknesses in either precision, recall, or both.
5. **Mean Average Precision (MAP):** This metric gives an overall view of retrieval performance. A high MAP indicates the model reliably retrieves relevant documents across queries, while a low MAP suggests inconsistencies in relevance across queries.
6. **Mean Reciprocal Rank (MRR):** This metric considers the rank position of the first relevant document, averaged over all queries. A high MRR implies relevant documents appear early in the ranking, while a low MRR suggests they tend to appear later, reducing their immediate visibility to users.
7. **Normalized Discounted Cumulative Gain (NDCG):** This metric assesses the relevance of retrieved documents, prioritizing those higher in the ranking. A high NDCG means relevant documents appear early in the ranked list, while a low NDCG indicates relevant documents are mixed in with irrelevant ones or appear lower in the list, reducing their impact.

The detailed scores for each K, can be found in the accompanying Python document. The key findings of the results for each query we proposed are the following:

- **India government farmers protest:** Precision decreases fast as K increases, suggesting that while some relevant documents are present in the top results, many retrieved documents become irrelevant with a larger set. This is confirmed by recall, which remains constant at 0.2 across K = 10, 15 and 20. NDCG starts high but drops significantly at K = 10 and stabilizes thereafter, indicating that while the top results are ranked relatively well, the quality of the ranking diminishes substantially as more documents are included, leading to less effective retrieval beyond the initial set.
- **Increasing prices:** Precision starts at 1.0 but decreases as K increases, indicating that while the top results are excellent, the quality declines when considering more documents. Recall remains constant at 0.5 for all K beyond 5, demonstrating good coverage by consistently retrieving half of the relevant documents. NDCG is high at K=5, but decreases and stabilizes at lower values for larger K, suggesting ideal ranking for the top results.
- **Disha Ravi:** Precision improves as K increases, indicating relevant documents are found deeper in the rankings. Recall reaches 1.0 by K = 20, showing that all relevant documents are retrieved with a larger set. It is important to consider that supportive tweets like "Free Disha" are not considered relevant in the ground truth. NDCG starts low but improves with larger K, indicating better ranking of documents.

- **Narendra Modi** and **Anime:** All metrics remain zero across all K values, indicating a failure to retrieve relevant documents. This highlights a significant struggle for the system with these queries, suggesting potential issues with the query processing or the index creation. However, these results for Narendra Modi can be caused by how we created the ground truth document.

2.4. Two dimensional representation of tweets by T-SNE

The objective of this section is to represent tweets in a two-dimensional space to visually explore the similarities and differences between them. By converting each tweet into a vector representation and using T-SNE for dimensionality reduction, we aim to reveal potential clusters or patterns that may reflect the different topics or queries. This visualization may help us understand the distribution and relationships of tweets related to specific search queries.

For this we focused on the following queries:

- **What are the people's rights?:** T-SNE visualization shows a dense and uniform clustering, with some subclusters suggesting different interpretations on rights. Given the broad nature of rights, it is likely that tweets cover a spectrum of rights, reflecting general concerns about the citizens' rights in times of political tension.
- **Narendra Modi:** This plot shows a more dispersed distribution of points, indicating varied perspectives. Some dense regions could signify focused discussions on Modi's actions, while isolated points may represent less frequent points of view.
- **Disha Ravi:** This visualization shows clear wavy patterns, suggesting polarized conversations for her situation after her controversial arrest. Clusters in this visualization likely represent the temporal aspects of the situation, as some could relate to the immediate reaction to her arrest (a lot of the retrieved tweets were "Free Disha") and others to the responses after her releases. The structure of the plot highlights the emotional and polarized nature of the discourse surrounding her case, which became a focal point of debate on the protest for India's response to the dissenting voices.
- **Increasing prices:** The plot reveals two prominent clusters. One of them has spare points, suggesting a broader range of discussions that may not directly address the issue of rising prices. The other one consists of three or four areas with a high density of points, probably indicating specific discussions focused on price hikes related to the farmers' protests.

So the T-SNE visualizations show distinct patterns in public conversations for each query. Tweets about "What are the people's rights?" form tight clusters, suggesting a unified focus on rights-related concerns. In contrast, the "Narendra Modi" plot is more spread out, reflecting mixed opinions and complex discussions surrounding his involvement in the protests. For "Disha Ravi," the visualization shows a wavy pattern, capturing the polarized and emotional responses to her activism and arrest.