Profile-Based Retrieval

Information Retrieval, Extraction & Integration

Course 2022/23

Víctor Morcuende Castell

Guillermo Nájera Lavid

TABLE OF CONTENTS

1. Introduction

The proliferation of digital content on the internet has led to an ever-increasing volume of information available to users. While this provides access to a vast source of knowledge, it also introduces the challenge of efficiently finding relevant and personalized content. In response to this challenge, profile-based information retrieval systems have emerged as an effective solution for tailoring content recommendations to individual users based on their unique interests and preferences.

Profile-based information retrieval systems construct user profiles that encapsulate users' interests, preferences, and previous interactions with content. These profiles serve as the basis for personalized content recommendations, enabling the system to present users with content that is more relevant to their individual preferences. Such systems have widespread applications across various domains, including news websites, e-commerce platforms, and entertainment services.

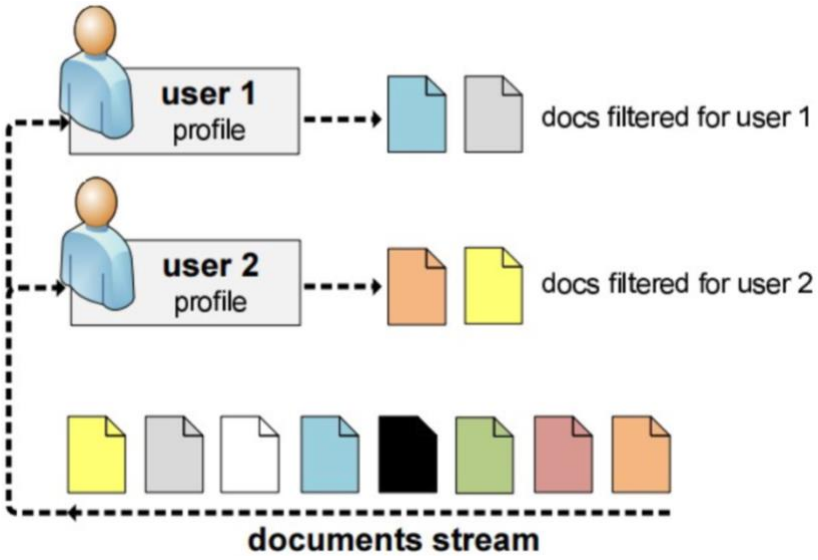

Figure 1: Profile-based information retrieval system

The primary objective in this assignment is to design, implement, and evaluate a content-based recommendation system that identifies and recommends small text snippets (or articles) tailored to user profiles. The system aims to accurately discern articles that are relevant to users' interests, resulting in a more personalized and engaging user experience. In the pursuit of this goal, we will explore different techniques for constructing user profiles, such as the "Weighted Topic Frequency" (WTF) method and assess the efficacy of various classifiers in pinpointing relevant content.

Our project will comprise several stages, beginning with data preprocessing and feature extraction, followed by user profile creation using the WTF method. Subsequently, we will employ similarity measurements to establish connections between user profiles and documents, leading to personalized content recommendations. Finally, we will evaluate

the system's performance using a range of metrics, assessing the results obtained from multiple classifiers.

2. Problem Context and Dataset Selection

To tackle the problem described in the previous section, we require a dataset that contains a substantial number of articles with sufficient information to construct meaningful user profiles and develop a robust recommendation system. For this purpose, we have chosen the "BBC News Train" dataset, which is a collection of news articles from the BBC. The dataset comprises 1.490 articles spanning across five different categories: business, entertainment, politics, sport, and tech.

	ArticleId	Text	Category
0	1833	worldcom ex-boss launches defence lawyers defe...	business
1	154	german business confidence slides german busin...	business
2	1101	bbc poll indicates economic gloom citizens in ...	business
3	1976	lifestyle governs mobile choice faster bett...	tech
4	917	enron bosses in $168m payout eighteen former e...	business
5	1582	howard truanted to play snooker conservative...	politics
6	651	wales silent on grand slam talk rhys williams ...	sport
7	1797	french honour for director parker british film...	entertainment
8	2034	car giant hit by mercedes slump a slump in pro...	business
9	1866	fockers fuel festive film chart comedy meet th...	entertainment

Figure 2: Dataset Appearance

The choice of the BBC News Train dataset offers several advantages. Firstly, the diverse range of topics covered in the dataset allows us to create user profiles that encompass a variety of interests, making it suitable for developing a versatile recommendation system. Secondly, the dataset contains high-quality and well-structured articles, ensuring that our system can accurately capture the fine differences of the content. Additionally, the dataset has been used in numerous research studies and machine learning competitions, which facilitates the comparison of our results with existing work in the field.

3. Preprocessing and Cleaning the Dataset

To develop an effective profile-based information retrieval system, it is essential to preprocess and clean the dataset to ensure that it is in a suitable format for analysis. This process involves several steps:

1. Importing the dataset: the first step is to import the dataset into our Python environment. We read the CSV file containing the BBC News Train dataset and stored it in a suitable data structure, such as a Pandas DataFrame, for easy

manipulation and analysis. This allowed us to access and process the data in an efficient and intuitive manner.

2. Removing duplicate articles: to ensure that our dataset is accurate and representative, we removed any duplicate articles that may be presented, as duplicate articles can lead to biased results and affect the performance of our recommendation system. We achieved this by identifying and removing rows with identical values in the "Text" column using the 'drop_duplicates()' function available in Pandas. This helped us maintain the integrity of the dataset and prevented any distortions in the results of our analysis. Moreover, as it can be appreciated below, the dataset contains an unbalanced number of articles for each topic:

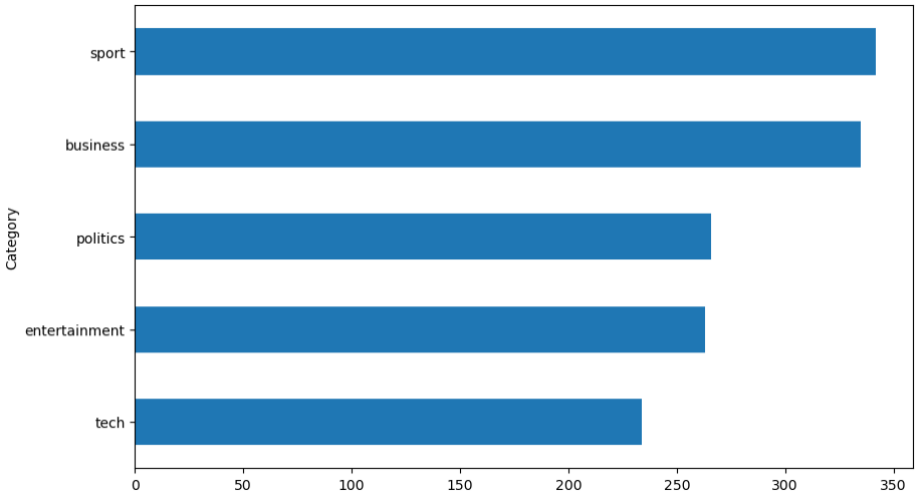

Figure 3: Unbalanced Dataset

3. Text cleaning: as the dataset contains textual data, it is crucial to clean and preprocess the text to ensure it is in a suitable format for analysis. Therefore, to carry out this process we performed the following steps:

    a. Removing punctuation marks: punctuation marks can introduce noise and make it difficult to process the text. By removing them, we simplified the text and facilitated the analysis.

    b. Converting the text to lowercase: this helped us to ensure that the text is consistent and that words are not treated as separate entities due to differences in case.

    c. Tokenization: breaking the text into individual words (or tokens), enabled us to analyze the text at the word level and helped us in identifying meaningful patterns and relationships. Furthermore, we removed the tokens whose length was lower than 3 characters, since typically these words do not provide useful information and they would not help us to achieve our goals.

d. Removing stop words: commonly used words such as “and”, “the”, and “in” do not carry much meaning and can be safely excluded from the analysis. By applying this change, we reduced the size of the dataset and improved the efficiency of subsequent processing steps.

e. Lemmatization: this technique reduces words to their dictionary base forms, further simplifying the text and reducing the number of unique words in the dataset, which could help us in our aim of improving the performance and efficiency of the analysis.

4. Creating a clean dataset: after completing the preprocessing and cleaning steps, we were left with a clean dataset that was ready for further analysis. This dataset was now free of duplicates, missing values, and irrelevant information, and the textual data has been preprocessed and simplified to ensure it is in a suitable format for analysis. With this clean dataset, we can proceed to develop our profile-based information retrieval system, extract relevant features, and evaluate its performance using appropriate metrics and evaluation techniques.

4. Profile’s Creation

After that, we started with the creation of user profiles, which is a pivotal component in a profile-based information retrieval system. This section delves deeper into balancing the dataset, the reasoning behind selecting the TF-IDF and WTF methods for vectorizing documents and profiles, and the formation of topics and profiles.

Balancing the dataset

In the context of a multi-class classification problem, it is crucial to maintain a balanced dataset to prevent potential biases in the model’s performance. These biases could lead to overemphasis on certain categories and neglect of others, thereby compromising the model’s accuracy and efficiency. For this reason, we reduced the size of each topic by the size of the smallest one, which in this case was the tech topic with 234 articles. By ensuring that an equal number of articles are presented for each category, the model is consequently exposed to a representative sample of each topic, which in turn enhances its ability to discern between different categories and provide more accurate results.

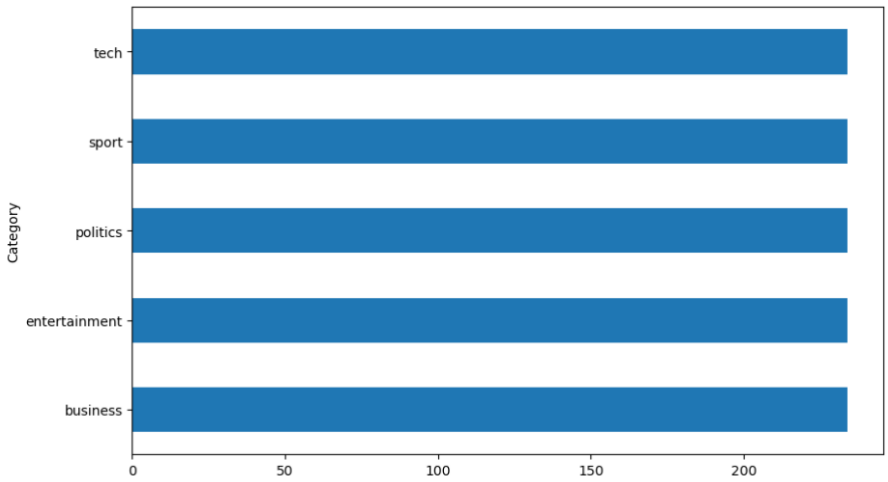

Figure 4: Balanced Dataset

Topics

The selected dataset comprises news articles from five distinct categories: business, entertainment, politics, sport, and tech. Therefore, to create the user profiles, a set of keywords was identified for each category or topic. These keywords encapsulate the topic's primary themes and act as the foundation for defining user interests.

To carry out the keywords selection, we performed an analysis of the dataset, retrieving the 100^{th} most frequent features (words) of each category. Then, we conducted an analysis of these features and filtered them for each topic, keeping only the 50 most important ones, from our point of view. As a result, we ended up with 50 keywords for each of the categories.

TF-IDF and WTF for vectorizing documents and profiles

The Term Frequency-Inverse Document Frequency (TF-IDF) method was chosen for vectorizing the documents and user profiles due to its robust and widely accepted performance in text representation. TF-IDF's popularity stems from its ability to convert text into numerical format, a requirement for machine learning algorithms to effectively process and analyze data. This technique calculates the importance of each term within a document based on its frequency in the document and its rarity across the entire corpus. Consequently, it helps identify terms that are significant and distinctive, which can then be utilized as features to differentiate between various topics. The TF-IDF matrix will be later used to build the user profiles by associating the users' preferred topics with the weighted terms in the articles.

In addition, after obtaining the TF-IDF representation of the news articles, we implemented the "Weighted Topic Frequency" (WTF) method, an algorithm that we developed to create user profiles that represent the users' interests and preferences. The WTF method is a technique for representing user interests by analyzing their interactions

with documents belonging to specific topics. In this approach, we assign a weight to each topic based on the user's interaction with the articles and use this information to create user profiles. The weight represents the importance of the topic for the user and is determined by the number of times the user has interacted with articles from that topic. By identifying user preferences and interests in the different topics, we can create comprehensive profiles of their preferences. To explain this method better, the steps followed to implement it are the following:

1. Term Frequency (TF): first, we use a function which counts the frequency of each word in the profile's interests and normalize it by the total number of words in that same profile's interests.

2. Inverse Topic Frequency (ITF): then, another method is used, in which, for each word in the profile's interests, its presence in all topics is calculated. Then, it computes the inverse of this presence (total number of topics / number of topics containing the word). This will give as a result higher weights to words that are more unique to a particular profile's interests.

3. Weighted Topic Frequency (WTF): finally, by multiplying the TF and ITF functions for each word, the WTF value is obtained. This will emphasize words that are both frequent in the profile's interests and unique to their topics.

Creation of profiles

By combining the TF-IDF and WTF methods, a more precise representation of the user's interests can be achieved, as the WTF method accounts for the term's significance within the topic, while the TF-IDF method considers its significance across the entire corpus. This combination leads to a more robust and accurate user profile.

Therefore, after the detailed processes mentioned above, we combined the WTF method and the TF-IDF matrix to create accurate user profiles that capture the users' interests and preferences within the context of the five predefined topics. We did this by associating the users' preferred topics, as determined by the WTF method, with the weighted terms in the articles derived from the TF-IDF representation. This results in detailed user profiles that can be used to provide personalized recommendations and enhance the information retrieval process.

Consequently, we created several user profiles based on the five predefined topics (business, entertainment, politics, sport, and tech). Being precise, we decided to create first a profile for each topic, that is, the user would only be interested in one topic:

- Profile 1: sport
- Profile 2: business
- Profile 3: entertainment
- Profile 4: politics
- Profile 5: tech

After that, we opted to create profiles by combining different categories for the purpose of analyzing and evaluating how our weights' methods (TF-IDF and WTF) worked:

- Profile 6: sport and business
- Profile 7: entertainment and politics
- Profile 8: tech and sport
- Profile 9: business and entertainment
- Profile 10: politics, tech, and business

5. Profile-based Retrieval Implementation

In this section, we discuss the implementation of the profile-based retrieval system using the user profiles created in the previous step. The primary objective of our system is to deliver personalized content recommendations to users based on their interests and preferences.

Representing User Profiles

First, the user profiles generated earlier are represented as vectors containing the weighted terms derived from the combination of the WTF method and the TF-IDF matrix. These vectors effectively capture the users' preferences across the five predefined topics in the BBC News dataset, enabling us to make personalized recommendations.

Representing Documents

Then, similar to the user profiles, we represented the news articles in the dataset as vectors using their TF-IDF values. This representation allowed us to quantify the relevance of each article to a particular user by comparing their vector representations.

Similarity Calculation

To determine the relevance of a news article to a specific user, we computed the similarity between the user's profile vector and the document's vector representation. To accomplish this, we used the cosine similarity measure, which calculates the cosine of the angle between the two vectors. The cosine similarity ranges from -1 (completely dissimilar) to 1 (identical), with higher values indicating a stronger similarity between the

user's profile and the particular document. This similarity score forms the basis of our content recommendations.

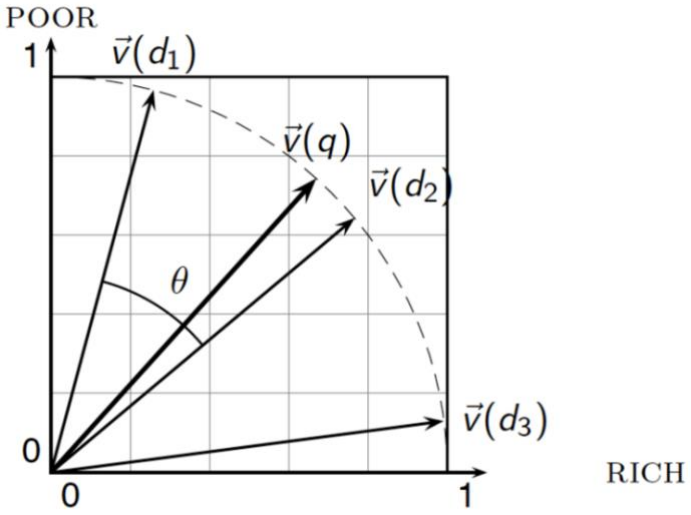

Figure 5: Cosine Similarity Explanation

Ranking and Recommendation

After calculating the cosine similarity between each user's profile vector and the document vectors, we proceeded to rank the articles based on their similarity scores. The top-ranked documents were considered to be the most relevant to the user's preferences and are recommended accordingly. This ranking system ensured that the content recommendations were tailored to each user's unique interests, providing a more personalized and satisfying information retrieval experience.

```
Profile Interests and Predicted Categories:

Profile ID: 1
Interests: sport, match, champion, play, injury, team, chelsea, season, side, second, final, club, rugby, coach, united, arsenal, nation, robinson, great, start,
Top 5 recommended documents:
  Document index: 766, Category: sport
  Document index: 815, Category: sport
  Document index: 805, Category: sport
  Document index: 859, Category: sport
  Document index: 857, Category: sport

Profile ID: 2
Interests: business, year, firm, company, rate, economy, growth, sale, market, bank, price, share, dollar, economic, country, government, month, profit, figure,
Top 5 recommended documents:
  Document index: 183, Category: business
  Document index: 22, Category: business
  Document index: 93, Category: business
  Document index: 16, Category: business
  Document index: 52, Category: business

Profile ID: 3
Interests: entertainment, film, award, band, actor, star, album, oscar, festival, director, song, single, actress, record, singer, nomination, movie, book, music
Top 5 recommended documents:
  Document index: 1054, Category: entertainment
  Document index: 1087, Category: entertainment
  Document index: 976, Category: entertainment
  Document index: 1137, Category: entertainment
  Document index: 1046, Category: entertainment
```

Figure 6: IR System Implementation

6. Performance Assessment

At this point, we started with the performance evaluation of our profile-based retrieval system. To achieve this, we tried several evaluation metrics, including Precision@k, Recall@k, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG)@k. These metrics helped us to determine how well our system was performing in terms of retrieving relevant documents for a given profile.

To evaluate the performance of our system, as we have explained above, we calculated the top "n" matches for each profile using cosine similarity between the profile vector and the document vectors. We then used these top "n" matches to predict the category of the document and compare it with the true category label to calculate the evaluation metrics. After this, we performed the following metrics:

- Precision@k: we calculated the proportion of correctly predicted categories in the top "k" (in this case, we used the same number as "n") recommendations for each profile.

- Recall@k: we calculated the proportion of relevant categories retrieved in the top "k" recommendations.

- MAP: we calculated the average of the precision (AP) across all profiles. AP is the area under the precision-recall curve, and it measures the average precision for each category.

- MRR: we calculated the average reciprocal rank of the relevant categories. This metric gives more weight to the higher-ranked recommendations that are more relevant.

- NDCG@k: we used the DCG and IDCG metrics. DCG stands for "Discounted Cumulative Gain", which is a metric that measures the ranking quality of the retrieved categories. IDCG stands for "Ideal Discounted Cumulative Gain", which is the DCG that would be obtained if all the relevant categories were ranked at the top. NDCG@k is calculated as the ratio between DCG and IDCG at the top k recommendations.

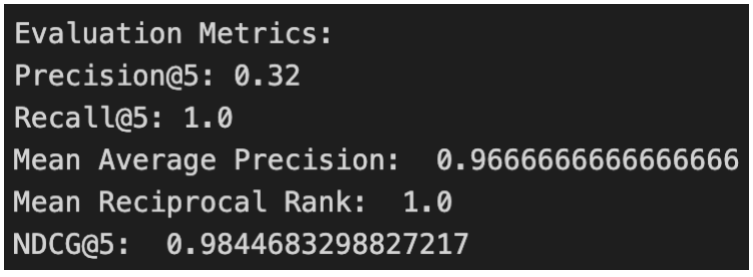

Figure 7: Evaluation Metrics Results

7. Model Creation and Evaluation

In this section, we will discuss the machine learning algorithms used for the classification of different documents for our user profile information retrieval system. We will also analyze the results of each algorithm and determine which one is better based on the results.

To perform this task, we started by splitting our dataset into training and testing sets (80-20 ratio) using the train_test_split() function from the sklearn.model_selection module. We then vectorized the text using the TfidfVectorizer() to transform the text data ("clean_text" column) into numerical format.

After that, the machine learning algorithms that we chose to use were: Multinomial Naive Bayes (MNB), Logistic Regression (LR), Linear Support Vector Classification (LSVC), Random Forest Classifier (RFC), and K-Nearest Neighbors (KNN) classifier. Therefore, each classifier was trained on the training set and evaluated on the testing set using the train_and_evaluate() function, which calculated the accuracy, precision, recall, and F1-score for each class, apart from the confusion matrix of each classifier.

The results of the evaluation were as follows:

Model	Accuracy	Precision	Recall	F1-Score
MNB	0.9744	0.9744	0.9737	0.9740
LR	0.9530	0.9533	0.9535	0.9526
LSVC	0.9615	0.9611	0.9620	0.9611
RF	0.9573	0.9585	0.9547	0.9555
KNN	0.9145	0.9149	0.9148	0.9133

Based on these results, the Multinomial Naive Bayes classifier demonstrated the best performance among all five algorithms. It achieved the highest accuracy, precision, recall, and F1-score. The reason behind is that the Multinomial Naive Bayes classifier is particularly suitable for text classification tasks, as it assumes that the features (words) are conditionally independent given the class. This assumption, although simplistic, works well for text data and enables the model to effectively capture the relationships between words and their corresponding categories.

Furthermore, we performed cross-validation with 5 folds on the entire dataset to verify the reliability of the results. The cross-validation scores were consistent with the initial evaluation, further reinforcing the superiority of the Multinomial Naive Bayes classifier for this particular task. The cross-validation scores for each classifier were:

Model	Accuracy
Multinomial Naive Bayes	0.9650
Logistic Regression	0.9641
Linear Support Vector Classification	0.9624
Random Forest Classifier	0.9496
K-Nearest Neighbors Classifier	0.9162

In conclusion, the Multinomial Naive Bayes classifier is the most suitable algorithm for the task of classifying different documents for our user profile information retrieval system. It outperformed the other classifiers in terms of accuracy, precision, recall, and F1-score, making it the ideal choice for this application.

8. Results

For this section we will discuss the results of both implementations, the cosine similarity approach, which has constituted most of the work done, and the different machine learning algorithms we used.

After individually discussing the results for each of the approaches in previous sections, we can confidently say that machine learning is a far better approach when two key aspects are available: having labeled data and having enough computational resources available for use. This is true since we have seen overall better performance at the time of classifying each document, which then can be fed to the users whose interests are aligned with the category in question.

Nonetheless, the aim of this assignment is to be able to perform the asked tasks by pure comparison of the interests of each profile with the documents, therefore, the use of machine learning and labeled texts defeats the purpose of the same.

9. Performance comparison

In this section, we will compare the results obtained by other groups to the ones from our models. In this case we are going to be comparing our models with Group 2 (Dante, Elena and Javier), who has followed the same approach of implementing both the cosine similarity and a machine learning solution, something that we think will give more

revealing results, since both groups followed the same principles and ideas when doing the assignment.

For the cosine similarity solution, this group opted for the implementation of a classifier that instead of comparing the interests from users and try to match them with the documents present in the dataset, would directly classify each document and match it to a precise category, to then check with the following user profiles and see what possible matches there may be, giving them an 88% of accuracy. This approach is quite interesting from the perspective of a pure classifier, given the fact that it will probably be much more generalist and might work better with simpler queries and general texts, hence the high accuracy seen in the test dataset.

Nonetheless, we think it may be more limited when trying to perform for more complex queries, such as when trying to recommend documents with users with more interests, depending of course on the way these interests are blended and interpreted with one another. For example, it's not the same to have interests in both politics and tech and being recommended either one, or being recommended a document which represents both politics and tech. In this later case, a pure classifier that does not consider the interests of each individual profile will not be able to perform as well as one that does.

To continue, we will now discuss the two machine learning approaches followed by both groups. In this case, both groups implemented the random forest algorithm, with fairly similar results. Our approach gave us a 95% accuracy, opposed to the 97% accuracy shown by this group. However, we tested other algorithms, as previously discussed, with three of them giving better results than random forest. In this case, the Multinomial Naive Bayes gave us the highest accuracy at a 97%.

10. Conclusions

We have seen during the completion of this assignment the potential performance of an algorithm that, at first, may seem quite simple, but in the contrary of what it may seem, the cosine similarity has proven to be not only reliable, but incredibly efficient and precise. We have been able to make precise classifications and predictions without the need for having the label of each text (although for this case we had the labels included in the dataset for the sake of testing measurements). Thanks to the work done with this approach, we were able to see and work with other approaches that did not have to necessarily involve machine learning, which has substantially widened our understanding of the information retrieval process. However, given the fact that we also used machine learning for the purpose of the assignment, room was made for the possibility of comparing the differences in performance and complexity of both implementations.