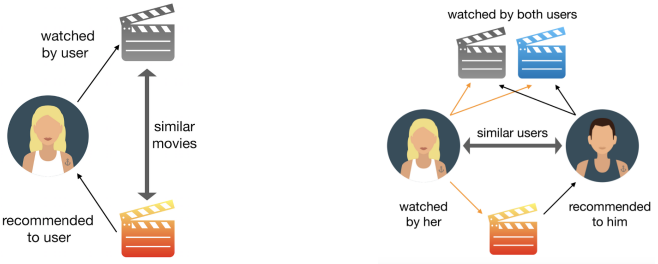
CmpE493 - Assignment 3
A Book Recommendation System

Contact: Gökçe Uludoğan (gokce.uludogan@boun.edu.tr)

Deadline: January 6, 2021, Wednesday, 17:00

1 Description

Building a recommendation system is a common task in many modern applications. The goal of a recommendation system is to identify relevant data for their users. These systems are generally based on two methods: collaborative filtering and content based filtering. While collaborative filtering exploits similar users' rates, content based filtering considers the contents of the items liked by the same user.



(a) Content based filtering (b) Collaborative filtering

Figure 1: Recommendation systems [1]

In this assignment, you will build a simple book recommendation system from scratch. Firstly, you will extract the contents and recommendations of books from Goodreads. To represent the contents of the books, you will implement the vector-space model. For the book descriptions, you are required to use tf-idf weighting, while you are free to choose any model for the genres. After vectorizing all books' contents, you will make recommendations for a given book by getting the most similar K books based on cosine similarity. Finally, you will calculate the evaluation metrics considering the Goodreads recommendations as the relevant (ground truth) books.

The urls for the books are available on Moodle. You can use any programming language of your choice. The libraries you are allowed to use are the standard libraries of the language you use. You are required to extract the contents of the books with regular expressions, thus using a library for parsing HTML is not allowed.

[1] Source: `https://bit.ly/2AncOE4`

2 Implementation

2.1 Data set

2.1.1 Extracting content from Goodreads

Goodreads urls for books are given on Moodle. The web page for an example book, The Little Prince, is shown below (the url is shown in the yellow box in Figure 2).

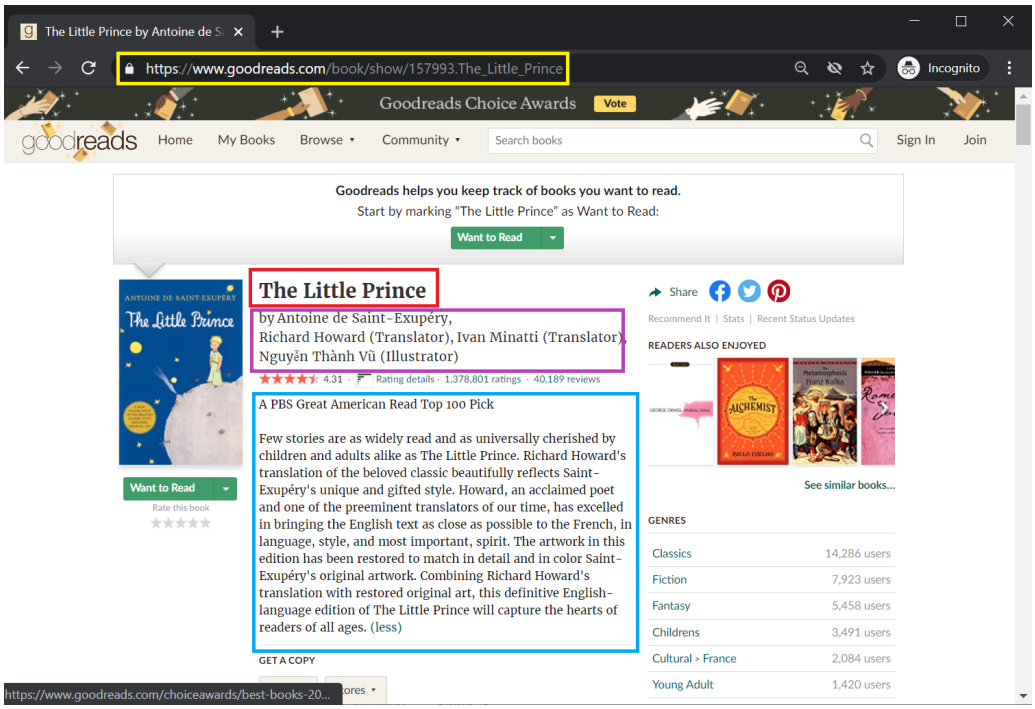


Figure 2: Goodreads web page for *The Little Prince*

The data which must be collected for each book are:

1. Title of a book (seen in a red box in Figure 2)
2. Author(s) of a book (seen in a purple box in Figure 2)
3. Description of a book (seen in a blue box in Figure 2)
4. Urls of all recommended books[2] (seen in a red box in Figure 3)
5. Genres of a book[3] (seen in a gray box in Figure 3)

[2]Not only the ones seen in the current panel, but also those in the next panels.
[3]Not only the ones seen in the current panel, but also those in the next panels.

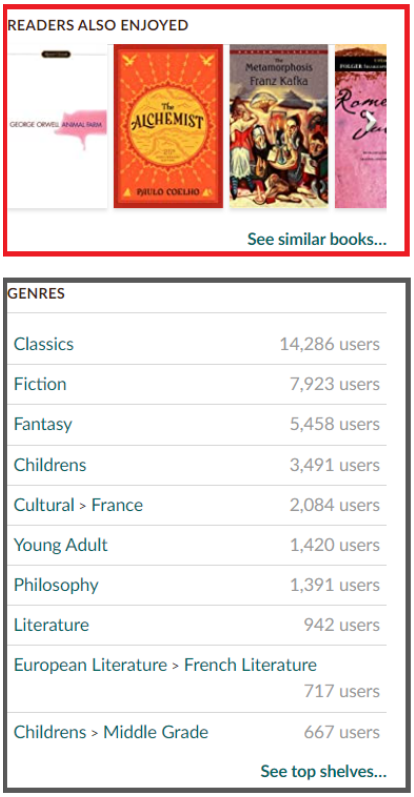


Figure 3: Recommendations and genres for *The Little Prince*

2.2 Recommendation System

In this assignment, you will build a content based book recommendation system. The contents you are allowed to use are the descriptions and genres of the books. To represent the descriptions of the books in a vector space, you are required to implement the tf-idf based vector space model, while you can vectorize the genres with any model you want.

2.2.1 TF-IDF Model

To encode the descriptions of the books, you will process the description of the books, identify the vocabulary, as well as the term and inverse document frequencies. Then, you will select the informative terms and encode each book's description by using the occurrences and scores of these words. To select the informative words, you can either set minimum/maximum thresholds on the frequencies and number of terms or try different variants of tf-idf weighting.

2.2.2 Recommendation

To recommend books, a simple content based strategy will be followed. Given a book, the system will recommend the most similar K books based on cosine similarity. Similarity of two books is computed as a weighted average of the cosine similarities of the descriptions and genres of these books as follows:

$$Sim(Book_a, Book_b) = \alpha * cosSim(Desc_a, Desc_b) + (1 - \alpha) * cosSim(Genre_a, Genre_b)$$

where $Book_a$ and $Book_b$ are two distinct books and $Desc_a$, $Desc_b$, $Genre_a$, $Genre_b$ are representations of descriptions and genres of these books respectively. In addition, α is a parameter specifying importance of descriptions and genres. Set a reasonable value to α.

2.2.3 Evaluation

To evaluate the system, you will recommend 18 similar books for each book in the evaluation set and then calculate the average precisions for these recommendations. The definitions of the precision metrics in this setting are as follows: [4]

- **Precision** is the fraction of Goodreads recommendations (relevant books) among your system's recommendations.

$$P = \frac{\#\ of\ recommendations\ that\ are\ relevant}{\#\ of\ recommendations}$$

- **Average Precision** is the average of precision values determined after each relevant book is retrieved.

$$AP@N = \frac{1}{m} \sum_{k=1}^{N} P(k)\ \ if\ kth\ book\ was\ relevant$$

  where N is the number of our system's recommendations and m is the number of Goodreads recommendations (relevant books) among your system's recommendations.

3 Input & Output

Your implementation must be executable from the command line. The program should take a string as input. This string is either a path to a file containing a list of book urls or a single url. If it is a file path, then the program must run the full pipeline:

- Extract the contents of the books of given urls and save them in a file.
- Identify terms and calculate weights.

[4] Nice post explaining metrics for recommendation: `http://sdsawtelle.github.io/blog/output/mean-average-precision-MAP-for-recommender-systems.html`

- Build and save the model with selected informative terms.

If the string is a single url, then the program must perform the following:

- Extract the content of the book whose url is given and output the content.
- Encode the content of the book and recommend 18 books for this book with the model.
- Evaluate the recommendations and output precision and average precision scores.

4 Submission

You are expected to submit a single zip file named as YourNameSurname.zip. The zip must contain the following files:

1. Report
   (a) Describe the model you used to encode the genres of the books
   (b) Describe the model parameters (minimum/maximum thresholds, number of terms, weight variants, α, etc.)
2. Commented source code and executable.
3. Readme: Describing how to run your program step by step.

Honor Code:

You should work individually on this assignment and all the source code should be written by you. You are NOT allowed to use any available libraries or any code written by other people. Violation of the Honor Code will be strictly penalised, not only by a zero grade from the homework, but also by filing a petition to the Disciplinary Committee.

Late Submission:

You are allowed 7 late days (one week) for this assignment with no late penalty. After 7 days, 10 points will be deducted for each late day.