# Self-assigned Assignment - Book Recommender

**Peter Thramkrongart and Jakub Raszka**

## Github link

Link to the repository: <https://github.com/PeterThramkrongart/cds-language-portfolio.git>

Link to the assignment folder: <https://github.com/PeterThramkrongart/cds-language-portfolio/tree/main/assignments/final-project>

## Contribution

Both Peter Thramkrongart and Jakub Raszka contributed equally to every stage of this project from initial conception and implementation, through the production of the final output and structuring of the repository. (50/50%)

## Description

In this project, the goal is to create a content-based book recommendation engine using information from Goodreads about the books such as summaries, genres, authors, and ratings. The engine sorts the whole dataset of books by customizable feature weights, such as the importance of description similarity, genre similarity, and the count of reviews.

## Methods

**Data**

Originally we wanted to use the continuously updated [Goodreads Book Datasets With User Rating 10M](https://www.kaggle.com/bahramjannesarr/goodreads-book-datasets-10m). This is a very extensive data set with a lot of recent and well-documented information. Problematically, it does not contain book summaries for the first 600k entries. Those entries are where most of the good and well-known books are, so we had to find another dataset.

We then stumbled upon a 8+ GB [dataset containing over 2.3 million books](https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/books?authuser=0). All that was contained in a single JSON file that we could not download using wget or other automatized download methods. Furthermore, we don’t think the size and the trouble of the dataset are justifiable given the small size of this project. We settled for a smaller, but much more concise, [Goodreads dataset](https://www.kaggle.com/meetnaren/goodreads-best-books) that contains only 52 932 best-rated books. Although it is only a fraction of the other datasets, we still deem the number of books sufficient, because those books were selected from a “best of” list.

Our chosen dataset was not without problems, though. The ISBN codes were corrupted, so we could no longer discern title duplicates from one another as easily, and it prohibited enriching the dataset with information from other datasets. The dataset comes with images for each entry. We did not use the images for this project, so we choose to delete them, thereby, making the dataset within the github file’s size limit of 100 MB.

**Data pre-processing**

The data required some pre-processing. Specifically, we dropped all duplicated books to have only one entry per one book. We also decided to keep only books with 50 and more reviews on Goodreads for the sake of optimization assuming that a reader has still many thousands of books to choose from.

The dataset also contained some books in different languages. To ensure that only English-written books are included, we used a module called [Spacy FastLang](https://github.com/thomasthiebaud/spacy-fastlang) which adds a language detection functionality into the *Spacy* world. The module utilizes facebook’s library [fastText](https://github.com/facebookresearch/fastText) which is a very extensive library of word embeddings and text classifications. We further used *Spacy* to lemmatize our book summaries. After the pre-processing, the dataset contained only 23 266 unique English books.

We z-score scaled all numeric values. This allows us to use them as weights in the final recommender ranking. Using *sklearn* vectorizers, we created count vectors of the author, genre columns, and TF-IDF vectors of the book summaries. We used a custom seperator for splitting the authors and genres columns instead of tokenizing single words. This allowed us to keep George Orwell separate from George Saunders and to keep Science Fiction separate from Historical Fiction. The TF-IDF vectorizer used unigrams, bigrams, and trigrams and deleted regular English stopwords.

The data frame was saved as a .csv and the vectorized columns were saved in Numpy’s proprietary binary format for fast loading when using the recommender.

**Recommender**

The recommender itself is very simple. It loads the processed data and calculates similarity scores for each vectorized variable using cosine similarity. Afterwards, we z-score scale them so their scales match. A final recommendation score is calculated by summing the different scaled scores multiplied by their weights. Also, the score is min-max scaled, to be of the same size no matter the weights. The score is in no way universal but gives some aid in understanding scores distribution and when comparing recommendations for different books. Lastly, the recommender sorts the full dataset based on the recommendation score and outputs the first 20 entries of the dataset.

The weighting scheme in Python code:

df["rec\_score"] =   
 df.tfidf\_sim \* tfidf\_weight +  
 df.genres\_sim \* genre\_weight +  
 df.book\_review\_count \* review\_count\_weight +  
 df.book\_rating \* rating\_weight +  
 df.authors\_sim \* authors\_weight

## Results

We find that it is difficult to reliably produce good content-based recommendations that work well in all scenarios. But most of the time our recommendations are somewhat logical. We heavily rely on our cleaning pipeline. Our recommendations are proof of that as the pipeline still lets many errors slip through. We also find it difficult to capture different editions of the same book, e.g., in the case of Animal Farm/ 1984 where the books were joined together to form a new third book.

Our scheme for language detection is also far from perfect as can be seen in the recommendations based on Ulysses by James Joyce. We noticed that while some summaries are in English, their book titles are in different languages suggesting that they might be translations with the original English summary. The discrepancy could be solved by running the language detection also on book titles. However, we think it would do more harm than good. Many titles are single-word strings that are sometimes tweaked or twisted on purpose. Or titles often contain non-English words which could lead to the loss of many great books. Instead, we choose to have here and there a title in the Cyrillic or Phoenician alphabet.

Our recommender also tends to recommend book compilations and spinoff works as shown in the recommendations based on The Hunger Games and A Game of Thrones. This is could be avoided using a much more advanced embedding scheme, that can capture the feel and themes of the books and filter out named entities. But in general, that is where collaborative filtering methods (recommendations based on what other people liked) are much better suited.

We are rather satisfied with our interactive weighting scheme. The default setting emphasizes the similarities in summaries, authors, and genres. If one wishes not to be shown spinoff works, one may deemphasize authors or summaries and mainly rely on the genre. That leads to fairly acceptable results as can be seen in the recommendations based on A Feast for Crows (AFFC) with the deemphasized author, compared to A Game of Thrones with the default setting.

Top 10 recommendations based on **1984** by George Orwell:

|book\_title |book\_authors | rec\_score|  
|:--------------------------------|:-----------------------------------|---------:|  
|A Collection of Essays |George Orwell | 1.0000000|  
|Animal Farm / 1984 |George Orwell, Christopher Hitchens | 0.9189596|  
|Animal Farm |George Orwell | 0.9107122|  
|Nineteen Eighty-Four |George Orwell | 0.8405285|  
|Down and Out in Paris and London |George Orwell | 0.8336836|  
|Keep the Aspidistra Flying |George Orwell | 0.7785718|  
|Shooting an Elephant |George Orwell | 0.7247096|  
|Homage to Catalonia |George Orwell, Lionel Trilling | 0.6990492|  
|Coming Up for Air |George Orwell | 0.6568643|  
|The Road to Wigan Pier |George Orwell, Richard Hoggart | 0.6183259|

Top 10 recommendations based on **Ulysses** by James Joyce:

|book\_title |book\_authors | rec\_score|  
|:------------------------------------------|:-------------------------------------------------------|---------:|  
|Дъблинчани/ Портрет на художника като млад |James Joyce, Джеймс Джойс, Асен Георгиев Христофоров... | 1.0000000|  
|Finnegans Wake |James Joyce | 0.9937516|  
|A Portrait of the Artist as a Young Man |James Joyce, Seamus Deane | 0.9635664|  
|Dubliners |James Joyce, Terence Brown, Colum McCann, Roman Muradov | 0.9014175|  
|Eveline |James Joyce | 0.8856827|  
|Elmer Gantry |Sinclair Lewis | 0.4147498|  
|The Last Temptation of Christ |Nikos Kazantzakis, Nikos Kazantzakis, Peter A. Bien | 0.4130681|  
|Tropic of Capricorn |Henry Miller | 0.3965215|  
|The Sound and the Fury |William Faulkner | 0.3933700|  
|Sometimes a Great Notion |Ken Kesey, Charles Bowden | 0.3870019|

Top 10 recommendations based on **The Hunger Games** by Suzanne Collins:

|book\_title |book\_authors | rec\_score|  
|:--------------------------------------|:---------------|---------:|  
|Mockingjay |Suzanne Collins | 1.0000000|  
|Catching Fire |Suzanne Collins | 0.9928887|  
|The Hunger Games Trilogy Boxset |Suzanne Collins | 0.5725807|  
|The World of the Hunger Games |Kate Egan | 0.5240631|  
|The Hunger Games Tribute Guide |Emily Seife | 0.4442769|  
|Gregor and the Prophecy of Bane |Suzanne Collins | 0.4304272|  
|Gregor and the Code of Claw |Suzanne Collins | 0.4160059|  
|Gregor and the Curse of the Warmbloods |Suzanne Collins | 0.4012589|  
|Gregor and the Marks of Secret |Suzanne Collins | 0.4001130|  
|Gregor the Overlander |Suzanne Collins | 0.3842942|

Top 10 recommendations based on **A Game of Thrones** by George R.R. Martin:

|book\_title |book\_authors | rec\_score|  
|:---------------------------------------------------------------------------------|:--------------------|---------:|  
|A Storm of Swords |George R.R. Martin | 1.0000000|  
|Tormenta de espadas |George R.R. Martin | 0.6612630|  
|A Game of Thrones: The First 5 Books |George R.R. Martin | 0.4452066|  
|A Clash of Kings |George R.R. Martin | 0.3666152|  
|A Dance with Dragons 2: After the Feast |George R.R. Martin | 0.3473805|  
|The World of Ice and Fire: The Untold History of Westeros and the Game of Thrones |George R.R. Martin...| 0.3131561|  
|A Dance with Dragons |George R.R. Martin | 0.3106538|  
|A Storm of Swords: Steel and Snow |George R.R. Martin | 0.3098506|  
|A Storm of Swords: Blood and Gold |George R.R. Martin | 0.3054644|  
|A Dance with Dragons: Dreams and Dust |George R.R. Martin | 0.2973343|

Top 10 recommendations based on **A Feast for Crows** by George R.R. Martin. These recommendations deemphasizes the authors by -1.0:

|book\_title |book\_authors | rec\_score|  
|:---------------------------------------------------------------------------|:-----------------|---------:|  
|The Iron Fey Boxed Set: The Iron King, The Iron Daughter, The Iron Queen... |Julie Kagawa | 1.0000000|  
|King of Thorns |Mark Lawrence | 0.8882314|  
|Treason |Orson Scott Card | 0.8856870|  
|Malice |John Gwynne | 0.8829934|  
|The Iron Legends |Julie Kagawa | 0.8807191|  
|The Name of the Wind |Patrick Rothfuss | 0.8739155|  
|The Iron Man: [A Children's Story In Five Nights] |Ted Hughes | 0.8738872|  
|Kings or Pawns |J.J. Sherwood | 0.8735123|  
|Words of Radiance |Brandon Sanderson | 0.8726171|  
|Black Wolves |Kate Elliott | 0.8722374|

## Reproducibility

**Step 1: Clone repository:**

* Open a Linux terminal
* Navigate the destination of the repository
* Run the following command

git clone https://github.com/PeterThramkrongart/cds-language-portfolio.git

**step 2: Set up the environment and activate it:**

* Navigate to the folder “final-project”

cd assignments/final-project

* We have written a bash script *create\_book\_recommender\_venv.sh* to set up a virtual environment:

bash create\_book\_recommender\_venv.sh  
  
source book\_recommender\_venv/bin/activate

**step 3: Prepare data:**

* Navigate to the folder “src”:

cd src

* Run the python script *prepare\_data.py* to clean the dataset and create necessary matrices:

python prepare\_data.py

**step 4: Run the recommender:**

* Run the python script *recommender.py* with the optional arguments:

flags: -t, --title, default = 1984, type = str,   
 help = str, Title to base recommendations on  
  
 flags: -tw, --tfidf\_weight, default = 1, type = float,   
 help = float, The weight of tfidf on recommendations  
  
 flags: -aw, --authors\_weight, default = 0.2, type = float,   
 help = float, The weight of authors on recommendations  
  
 flags: -gw, --genre\_weight, default = 0.8, type = float,   
 help = "loat, The weight of genre on recommendations  
   
 flags: -rcw, --review\_count\_weight, default = 0.2, type = float,   
 help = float, The weight of review counts on recommendations  
  
 flags: -rw, --rating\_weight, default = 0.3, type = float,   
 help = float, The weight of ratings on recommendations

* Make sure that the book’s title matches perfectly with the book’s title on [Goodreads](https://www.goodreads.com/) and was published no later than in 2017

python recommender -t "The Hunger Games" -tw 1.0 -aw -5.0 -gw 0.3 -rcw 0.0 -rw 2.0

* Note: The dataset was collected in 2017 so books published after cannot be used

**step 5 (optional): Kill the environment:**

* Navigate to the folder “final-project”

cd ..

* Run the bash script *kill\_book\_recommender\_venv.sh* to remove the virtual environment:

bash kill\_network\_analysis\_venv.sh

## Running the project on something else than Linux

Our projects are mainly made for Linux/Mac users. Our python scripts should run on any machine, though our bash scripts may not work. For this case, we recommend using the python distribution system [Anaconda](https://www.anaconda.com/) to set up environments using our requirements.txt files.