

Advanced Programming
Final Project: Report
Book Recommendation System

Made by: Nurlybai Beksultan IT-2207

Nagashybay Mukhtar IT-2205

Introduction

Problem:

I enjoy reading books, and it's a hobby of mine. While I often delve into the fantasy genre, I occasionally seek out good representations of other genres to broaden my reading experience. But sometimes I face difficulty finding books that truly interest me. That's why we decided to create a book recommendation system to solve this problem.

Literature review:

A recommendation system filters information by predicting ratings or preferences of customers for items that the customers would like to use. It tries to recommend items to the customers according to their needs and taste.

There are several methods to develop recommendation systems. We will show 3 of the most popular of them.

1. Basic first cut system would be popularity based or the system that makes use of dominating(popular) features within the organization. This method is the easiest one. To create this recommendation system, there will be a formula. For example, IMDB platform use the formula shown below:

$$W = \frac{Rv + Cm}{v + m}$$

where:

W = Weighted Rating

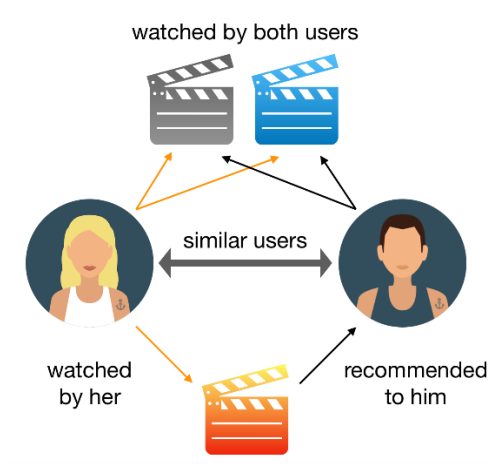
R = average for the movie as a number from 0 to 10 (mean) = (Rating)

v = number of votes for the movie = (votes)

m = minimum votes required to be listed in the Top 250 (currently 3000)

C = the mean vote across the whole report (currently 6.9)

2. Content-Based Filtering uses the item and user features to recommend other similar items to what the user likes, based on their previous actions or explicit feedback.



3. Collaborative Filtering. The motivation for collaborative filtering comes from the idea that people often get the best recommendations from someone with tastes similar to theirs.

The collaborative Filtering technique can be further classified into two groups:

3.1 Memory Based approach: The memory-based approach uses user rating data to compute the similarity between users or items. We will work on both user-user similarity recommendations and item-item similarity-based recommendations.

3.2 Model Based approach: In model-based approach models are developed using different data mining, machine learning algorithms to predict users' rating on unrated items.

The first three approaches can be developed without using deep learning algorithms. But the last "model based collaborative filtering" is what we are going to use.

Source: <https://medium.com/@amitdmlai/book-recommendation-system-61bf9284f659>

Current work:

We will develop a website where users can view other users' favorite books and books they would recommend. Our MLP model takes a user's ID and predicts which books a user might like by analyzing user ratings for different books. The website then displays the results of the query.

Data and Methods

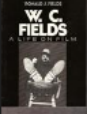



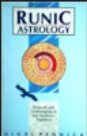
Information about the data:

We found datasets in this website <https://mengtingwan.github.io/data/goodreads>. There are 2 datasets. These are "goodreads_books.json.gz" and "goodreads_interactions.csv".

The dataset "goodreads_books.json.gz" contains book entries. Data such as book's ID, title, url of image, url of webpage, isbn, average_rating, description etc.

```
{'isbn': '0312853122',
 'text_reviews_count': '1',
 'series': [],
 'country_code': 'US',
 'language_code': '',
 'popular_shelves': [{'count': '3', 'name': 'to-read'},
 {'count': '1', 'name': 'p'},
 {'count': '1', 'name': 'collection'},
 {'count': '1', 'name': 'w-c-fields'},
 {'count': '1', 'name': 'biography'}],
 'asin': '',
 'is_ebook': 'false',
 'average_rating': '4.00',
 'kindle_asin': '',
 'similar_books': [],
 'description': '',
 'format': 'Paperback',
 'link': 'https://www.goodreads.com/book/show/5333265-w-c-fields',
 'authors': [{'author_id': '604031', 'role': ''}],
 'publisher': "St. Martin's Press",
 'num_pages': '256',
 'publication_day': '1',
 'isbn13': '9780312853129',
 'publication_month': '9',
 'edition_information': '',
 'publication_year': '1984',
 'url': 'https://www.goodreads.com/book/show/5333265-w-c-fields',
 'image_url': 'https://images.gr-assets.com/books/1310220028m/5333265.jpg',
 'book_id': '5333265',
 'ratings_count': '3',
```

We created a new dataset called "books_dataset.csv" and filled it with data from "goodreads_books.json.gz" to reduce the amount of unusable data.

	book_id	title	url	image_url
0	5333265	W.C. Fields: A Life on Film	goodreaders	
1	1333909	Good Harbor	goodreaders	
2	7327624	The Unschooled Wizard (Sun Wolf and Starhawk, #1-2)	goodreaders	
3	6066819	Best Friends Forever	goodreaders	
4	287140	Runic Astrology: Starcraft and Timekeeping in the Northern Tradition	goodreaders	

The second dataset "goodreads_interactions.csv" contains interactions between users and the books they rated. There are 5 fields. These are user_id, book_id, is_read, rating and is_reviewed. In these fields, we only needed user_id, book_id and rating fields. Also, there were too many records. Therefore, we decided to reduce the number of records to a million.

This is new dataset "dataset.csv" that is shown below:

	user_id	book_id	rating
0	0	948	5
1	0	947	5
2	0	946	5
3	0	945	5
4	0	944	5
...
999995	1970	8801	0
999996	1970	18136	0
999997	1970	33163	0
999998	1970	24332	0
999999	1970	37292	0
1000000 rows × 3 columns			

We got model from the website <https://gilberttanner.com/blog/building-a-book-recommendation-system-usingkeras/>. Type of our model is regression. It was trained with data such as users, books, and users' rating for books. Each user didn't rate all the books. But our model can predict users' rating for all books by analyzing users' actual rating for several books.

```
graph TD; Book[Book Input] --> BookEmbed[Embedding]; BookEmbed --> BookFlatten[Flatten]; User[User Input] --> UserEmbed[Embedding]; UserEmbed --> UserFlatten[Flatten]; BookFlatten --> Dot[Dot]; UserFlatten --> Dot;
```

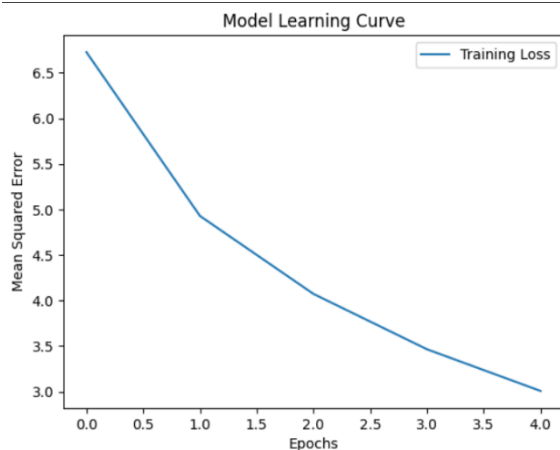
Dot: combines embeddings using a dot product.

```
book_input = Input(shape=[1], name="Book-Input")
book_embedding = Embedding(n_books+1, 5, name="Book-Embedding")(book_input)
book_vec = Flatten(name="Flatten-Books")(book_embedding)

user_input = Input(shape=[1], name="User-Input")
user_embedding = Embedding(n_users+1, 5, name="User-Embedding")(user_input)
user_vec = Flatten(name="Flatten-Users")(user_embedding)

prod = Dot(name="Dot-Product", axes=1)([book_vec, user_vec])
model = Model([user_input, book_input], prod)
model.compile('adam', 'mean squared error')
```

```
history = model.fit([train.user_id, train.book_id], train.rating, epochs=5, verbose=1)
```



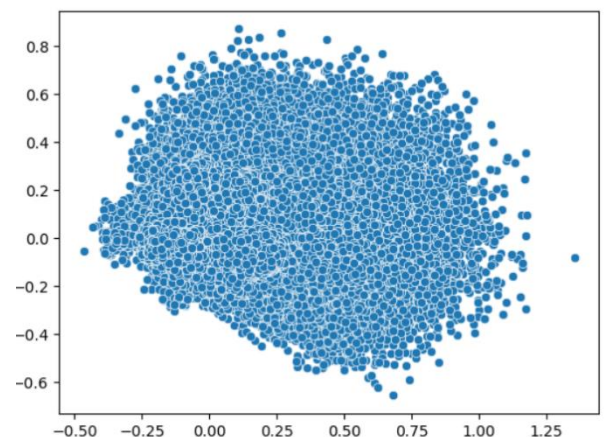
After training the model, we can show training loss in a graph.

```
from sklearn.decomposition import PCA
import seaborn as sns

book_em = model.get_layer('Book-Embedding')
book_em_weights = book_em.get_weights()[0]

pca = PCA(n_components=2)
pca_result = pca.fit_transform(book_em_weights)
sns.scatterplot(x=pca_result[:,0], y=pca_result[:,1])
```

Embeddings can be used to visualize concepts such as the relation of different books in my case. To visualize these concepts, we need to reduce dimensionality further using dimensionality reduction techniques like principal component analysis (PCA).



Results

The model predicts the selected user's rating for each book. Then, we sort the output and get an array of indexes. We use these indexes to get book_ids from dataset "dataset.csv". In this example below, we predict 1st user's ratings, and get top 20 the best books:

```
dataset = pd.read_csv('datasets/dataset.csv')
model = tf.keras.models.load_model('model/model.h5')

book_data = np.array(list(set(dataset.book_id)))
user = np.array([1 for i in range(len(book_data))])
predictions = model.predict([user, book_data])






unique_book_ids = dataset['book_id'].unique()
unique_book_ids.sort()

predictions = np.array([a[0] for a in predictions])
predictions_sorted = (-predictions).argsort()[:20]
recommended_book_ids = np.array([unique_book_ids[a] for a in predictions_sorted])
print(recommended_book_ids)
```

9462/9462 ————— 5s 541us/step









```
[ 68359  7243  23950  34852  23958  27587  46589 127912 100804 108740
 14533  77761  30475  56318  60463  87804  57631  19913  19092   678]
```

We can show a table of books by using these books' IDs and using dataset "books_dataset.csv". But before showing results of prediction, let's look at favorite books of 1st user:

book_id		title	url	image_url
90424	1063	Shibumi	goodreads	
90427	1064	Shibumi	goodreads	
185817	1036	Trump: The Way to the Top: The Best Business Advice I Ever Received	goodreads	
199524	1022	Read My Lips: Sexual Subversion and the End of Gender	goodreads	
205344	1018	Bodies of Subversion: A Secret History of Women and Tattoo	goodreads	

This table contains books' IDs, titles, images of books and URLs of book page in a website "goodreads.com".

After we executed model prediction, execution output was an array of 20 books' IDs. But the table of predicted books contains only 9 rows in example below. Because more than half of the books in the dataset "dataset.csv" are missing from the dataset "books dataset.csv".

book_id		title	url	image_url
184423	60463	The Empty House	goodreaders	
330291	19092	Middlemarch	goodreaders	
595845	57631	Sir Gawain and the Green Knight	goodreaders	
983013	7243	Prodigal Son	goodreaders	
1423650	87804	Yes Man	goodreaders	
1481302	108740	Qur'an and Woman: Rereading the Sacred Text from a Woman's Perspective	goodreaders	
1882380	19913	Master of the Moon (Mageverse #2)	goodreaders	
2229871	30475	The Communist Manifesto and Other Revolutionary Writings: Marx, Marat, Paine, Mao, Gandhi, and Others	goodreaders	

Discussion

Critical review of results:

The idea of this project was to develop a book recommendation system by using Deep Learning. While working on the project we had some difficulties.

Firstly, many recommendation systems can be built without ML/DL. That's why it was hard to find recommendation system examples that were created by using DL. Secondly, the majority of datasets were in forbidden website Kaggle, it was really difficult to find our desired datasets.

We created this project by following a plan, exact steps, they are:

1. Finding project idea
2. Data finding and analyzing
3. Finding the example of book recommendation systems that uses Deep Learning
4. Building our model by applying all information that we gathered before this step
5. Developing an interface (exactly website) to connect the user and our model.

Our system works correctly without any fatal errors. The model can give a list of books to the user that he will rate highly. As it was said before, more than half of the books in the dataset "dataset.csv" are missing from the dataset "books dataset.csv", that's why it is not giving all books.

Next steps:

Our book recommendation system works well and does its job correctly. However, it can be improved.

Firstly, dataset “books_dataset.csv” can be filled with book data about book ids from “dataset.csv” dataset. Secondly, authorization can be developed to allow users to be able to register into the system and get book recommendations themselves. Thirdly, the system can be optimized at all because it works a little bit slowly.

Reference

<https://medium.com/@akash.hiremath25/crafting-an-effective-recommendation-system-for-your-e-commerce-platform-a15d0f4b67e4>

<https://gilberttanner.com/blog/building-a-book-recommendation-system-usingkeras/>

<https://medium.com/@hamza.emra/introduction-to-recommendation-systems-with-tensorflow-recommenders-a116e5e5a940>

<https://mengtingwan.github.io/data/goodreads>