московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №4

по дисциплине «Методы машинного обучения»

на тему«Создание рекомендательной модели»

Выполнил:

студент группы: ИУ5-23М

Ся Тунтун

Москва — 2022 г.

Цель лабораторной работы: изучение разработки рекомендательных моделей.

Задание:

1. Выбрать произвольный набор данных (датасет), предназначенный для построения рекомендательных моделей.
2. Опираясь на материалы лекции, сформировать рекомендации для одного пользователя (объекта) двумя произвольными способами.
3. Сравнить полученные рекомендации (если это возможно, то с применением метрик).

Подключим все необходимые библиотеки

In [1]:
```
import numpy as np
import pandas as pd
import re
import string
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
```

Загрузим непосредственно данные:

In [3]:
```
#goodreads data
books_data = pd.read_csv('books.csv',error_bad_lines = False)
tags_data = pd.read_csv('book_tags.csv')
ratings_data = pd.read_csv('ratings.csv')
book_tags = pd.read_csv('tags.csv')

# book crossing data
user_cols = ['user_id', 'location', 'age']
cross_users_data = pd.read_csv('BX-Users.csv', sep=';', names=user_cols, encoding='la
book_cols = ['isbn', 'book_title' ,'book_author','year_of_publication', 'publisher', '
cross_books_data = pd.read_csv('BX_Books.csv', sep=';', names=book_cols, encoding='la
rating_cols = ['user_id', 'isbn', 'rating']
cross_ratings_data = pd.read_csv('BX-Book-Ratings.csv', sep=';', names=rating_cols, er
```

Посмотрим на данные в данном наборс данных:

In [4]:
```
books_data.head()
```

Out[4]:

	id	book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_
0	1	2767052	2767052	2792775	272	439023483	9.780439e+12	Suzanne Collins	
1	2	3	3	4640799	491	439554934	9.780440e+12	J.K. Rowling, Mary GrandPré	
2	3	41865	41865	3212258	226	316015849	9.780316e+12	Stephenie Meyer	
3	4	2657	2657	3275794	487	61120081	9.780061e+12	Harper Lee	
4	5	4671	4671	245494	1356	743273567	9.780743e+12	F. Scott Fitzgerald	

5 rows × 23 columns

In [5]:
```
cross_books_data.head()
```

Out[5]:

	isbn	book_title	book_author	year_of_publication	publisher	
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.cc
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.cc
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	http://images.amazon.cc
3	0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.amazon.cc
4	0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	http://images.amazon.cc

In [6]:
```
books_data = books_data.drop(columns=['id', 'best_book_id', 'work_id', 'isbn', 'isbn13
                                      'work_text_reviews_count', 'ratings_1', 'ratings_2'
                                      'image_url','small_image_url'])
```

Удаление ненужных данных

In [7]:
```
#drop unnecessary data
books_data = books_data.dropna()
cross_books_data = cross_books_data.drop(columns=['img_s', 'img_m', 'img_l'])
```

Удаление дубликатов из всего набора данных

In [8]:
```
#Drop Duplicates from all the dataset
ratings_data = ratings_data.sort_values("user_id")
ratings_data.drop_duplicates(subset =["user_id","book_id"], keep = False, inplace = T
books_data.drop_duplicates(subset='original_title',keep=False,inplace=True)
book_tags.drop_duplicates(subset='tag_id',keep=False,inplace=True)
tags_data.drop_duplicates(subset=['tag_id','goodreads_book_id'],keep=False,inplace=Tr
cross_ratings_data.drop_duplicates(subset =["user_id","isbn"], keep = False, inplace
cross_books_data.drop_duplicates(subset='book_title',keep=False,inplace=True)
```

очистение текста

In [9]:
```
#clean the text
def clean_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctua
    and remove words containing numbers.'''
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text
```

In [10]:
```
cross_books_data['book_title'] = cross_books_data['book_title'].apply(lambda x:clean_
```

In [11]:
```
merge_data = pd.merge(cross_books_data, cross_ratings_data, on='isbn')
merge_data =  merge_data.sort_values('isbn', ascending=True)
merge_data.head()
```

Out[11]:

	isbn	book_title	book_author	year_of_publication	publisher	user_id	rating
626418	0000913154	the way things work an illustrated encyclopedi...	C. van Amerongen (translator)	1967	Simon & Schuster	171118	8
587195	0001010565	mogs christmas	Judith Kerr	1992	Collins	209516	0
587194	0001010565	mogs christmas	Judith Kerr	1992	Collins	86123	0
441049	0001046713	twopence to cross the mersey	Helen Forrester	1992	HarperCollins Publishers	196149	0
263949	000104687X	ts eliot reading the wasteland and other poems	T.S. Eliot	1993	HarperCollins Publishers	23902	6

Фильтрация на основе содержания

Этот метод использует атрибуты содержимого для рекомендации аналогичного содержимого.он работает с атрибутами или тегами контента, такими как название книги, авторы или рейтинг, так что новую книгу можно сразу рекомендовать.

In [12]:
```
content_data = books_data[['original_title','authors','average_rating']]
content_data = content_data.astype(str)
```

In [13]:
```
content_data['content'] = content_data['original_title'] + ' ' + content_data['authors
```

Рекомендование авторов пользователям на основе содержания

In [14]:
```
content_data = content_data.reset_index()
indices = pd.Series(content_data.index, index=content_data['original_title'])
```

удаление стоп-слов ипостроим требуемую матрицу TF-IDF путем подгонки и преобразования данных

Преимущество кодировки TF-IDF заключается в том, что она будет взвешивать термин (тег для книги в нашем примере) в соответствии с важностью термина в документе: Чем чаще появляется термин, тем больше будет его вес.

In [18]:
```
#removing stopwords
tfidf = TfidfVectorizer(stop_words='english')

#Construct the required TF-IDF matrix by fitting and transforming the data
tfidf_matrix = tfidf.fit_transform(content_data['authors'])
tfidf_matrix.shape
```

Out[18]:
```
(8175, 5484)
```

Вычисление матрицы косинусного подобия

используем простой метод, основанный на сходстве, называемый косинусным подобием

In [19]:
```
#Compute the cosine similarity matrix
cosine_sim_author = linear_kernel(tfidf_matrix, tfidf_matrix)
```

автор мудрая рекомендация

(Получение парных оценок сходства всех книг с этим буой,Сортировка книг на основе оценок сходстваПолучение с 10 наиболее похожими книгами)

In [20]:
```
#author wise recommodation
def get_recommendations_books(title, cosine_sim=cosine_sim_author):
    idx = indices[title]
    sim_scores = list(enumerate(cosine_sim_author[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:11]
    book_indices = [i[0] for i in sim_scores]

    # Return the top 10 most similar books
    return list(content_data['original_title'].iloc[book_indices])
```

In [21]:
```
def author_book_shows(book):
    for book in book:
        print(book)
```

Вернение топ-10 самых похожих книг

In [22]:
```
books1 = get_recommendations_books('The Hobbit', cosine_sim_author)
author_book_shows(books1)
```

```
The Hobbit or There and Back Again
 The Fellowship of the Ring
The Two Towers
The Return of the King
The Lord of the Rings
The Hobbit and The Lord of the Rings
Nikola Tesla: Imagination and the Man That Invented the 20th Century
The Children of Húrin
Entwined
The 7 Habits Of Highly Effective Teens
```

In [23]:
```
books2 =get_recommendations_books('Shadow Kiss', cosine_sim_author)
author_book_shows(books2)
```

```
Shadow Kiss
Spirit Bound
Blood Promise
Last Sacrifice
Bloodlines
The Golden Lily
The Indigo Spell
The Fiery Heart
Succubus Blues
Silver Shadows
```

Коллаборативная фильтрация

В коллаборативной фильтрации рекомендуются элементы, например книги, основанные на том, насколько ваш профиль пользователя похож на профиль других пользователей, находит пользователей, наиболее похожих на Вас, а затем рекомендует элементы, которым они отдали предпочтение.

In [24]:
```
merge_data = merge_data[:40000]
```

In [25]:
```
book_rating = pd.pivot_table(merge_data, index='user_id', values='rating', columns='b
book_rating
```

Out[25]:

book_title	the year china discovered america	a beginners guide	a space odyssey a novel by arthur c clarke	allamerican favorites	an action plan to protect yourself your family your assets and your community on january	backup recovery	barbary lane a tales of the city omnibus	bee and chines me
user_id								
8	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0
73	0	0	0	0	0	0	0	0
86	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...
278692	0	0	0	0	0	0	0	0
278771	0	0	0	0	0	0	0	0
278818	0	0	0	0	0	0	0	0
278843	0	0	0	0	0	0	0	0
278851	0	0	0	0	0	0	0	0

12875 rows × 12020 columns

In [27]:
```
book_corr.shape
```

Out[27]: (12020, 12020)

In [28]:
```
book_list= list(book_rating)
book_titles =[]
for i in range(len(book_list)):
    book_titles.append(book_list[i])
```

Определение функции рекомендации

In [29]:
```
#Define Recommendation function
def get_recommendation_collabarative(books_list):
    similar_books = np.zeros(book_corr.shape[0])

    for book in books_list:
        book_index = book_titles.index(book)
        similar_books += book_corr[book_index]
    book_preferences = []
    for i in range(len(book_titles)):
        book_preferences.append((book_titles[i],similar_books[i]))

    return sorted(book_preferences, key= lambda x: x[1], reverse=True)
```

составнеие список книг

In [30]:
```
# make a book list
list_of_books = ['one hundred years of solitude',
                 'stardust',
                 'mogs christmas',
                 'dragonmede',
                 'twopence to cross the mersey',
                 'the candywine development']
```

In [31]:
```
books3 = get_recommendation_collabarative(list_of_books)
```

лучшие похожие книги коллаборативные

In [33]:
```
#top similar books collabarative
i=0
n =0
while n < 9:
    similar_books_to_read= books3[i][0]
    i += 1
    if similar_books_to_read in list_of_books:
        continue
    else:
        print(similar_books_to_read)
        n += 1
```

```
 the year china discovered america
a beginners guide
a space odyssey  a novel by arthur c clarke
allamerican favorites
an action plan to protect yourself your family your assets  and your community on j
anuary
backup  recovery
barbary lane  a tales of the city omnibus
beers and a chinese meal
```