Book Recommedation system

Author: ***DS Martin M. Waweru***

Project Criteria

This project will follow the CRISP_DM Criteria

Business understanding
Data Understanding
Data preparation
Modeling
Evaluation
Deployment

1.0 Business Understanding

1.1 Business problem

In an era of digital transformation, bookstores, online retailers, and libraries face challenges in effectively recommending books to users. Traditional recommendation methods often fail to personalize suggestions, leading to missed sales opportunities and reduced customer engagement. A data-driven book recommendation system can enhance user experience by providing tailored recommendations based on reading preferences, behavior, and historical data.

1.2 Overview

This project aims to develop a book recommendation system that leverages machine learning techniques to suggest books based on user preferences. The system will analyze user interactions, book ratings, and content-based features to generate relevant recommendations. The model will be designed for scalability, making it applicable to online bookstores, digital libraries, and educational platforms.

1.3 Project Objective

- Build a recommendation system that improves user engagement by providing personalized book suggestions.
- Utilize collaborative filtering, content-based filtering, or hybrid approaches to enhance recommendation accuracy.
- Optimize the system for scalability, allowing integration with e-commerce and library management platforms.
- Analyze user preferences and reading trends to refine recommendation strategies.

2.0 Data Understanding

2.1 Data Source

My project utilizes data obtained from Kaggle Download here, which was entirely scraped via the Goodreads API and was called books.

Data Column Description

1. bookID - Unique identifier for each book.
2. title - Title of the book.
3. authors - Names of the authors.
4. average_rating - Average rating given by users.
5. isbn - 10-digit International Standard Book Number (ISBN).
6. isbn13 - 13-digit ISBN for better identification.
7. language_code - Language in which the book is written.
8. num_pages - Number of pages in the book.
9. ratings_count - Total number of ratings received.
10. text_reviews_count - Number of text reviews submitted by users.
11. publication_date - Date when the book was published.
12. publisher - Name of the publishing company.

3.0 Data prepation

3.1 Preview dataset basic information

```python
# Necessary Imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import string
import re
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

## load the dataset
book = pd.read_csv("Data/books.csv", sep=",", on_bad_lines='skip')

print(f"First 3 rows of the dataset:")
display(book.head(3))
```

```
print(f"Last 3 rows of the dataset:")
display(book.tail(3))

First 3 rows of the dataset:

   bookID                                              title  \
0       1  Harry Potter and the Half-Blood Prince (Harry ...
1       2  Harry Potter and the Order of the Phoenix (Har...
2       4  Harry Potter and the Chamber of Secrets (Harry...

                      authors  average_rating        isbn
isbn13  \
0  J.K. Rowling/Mary GrandPré            4.57  0439785960
9780439785969
1  J.K. Rowling/Mary GrandPré            4.49  0439358078
9780439358071
2                J.K. Rowling            4.42  0439554896
9780439554893

  language_code    num_pages  ratings_count  text_reviews_count  \
0           eng          652        2095690               27591
1           eng          870        2153167               29221
2           eng          352           6333                 244

  publication_date        publisher
0        9/16/2006  Scholastic Inc.
1         9/1/2004  Scholastic Inc.
2        11/1/2003       Scholastic

Last 3 rows of the dataset:

       bookID                              title              authors  \
11120   45634  The Ice-Shirt (Seven Dreams #1)  William T. Vollmann
11121   45639                      Poor People  William T. Vollmann
11122   45641      Las aventuras de Tom Sawyer           Mark Twain

       average_rating        isbn         isbn13 language_code
num_pages  \
11120            3.96  0140131965  9780140131963           eng
415
11121            3.72  0060878827  9780060878825           eng
434
11122            3.91  8497646983  9788497646987           spa
272

       ratings_count  text_reviews_count publication_date
publisher
11120            820                  95         8/1/1993  Penguin
Books
11121            769                 139        2/27/2007
Ecco
```

```
11122              113                    12         5/28/2006  Edimat
Libros
```

```
# Checking the shape of the dataset
print(f"This dataset contains {book.shape[0]} rows and {book.shape[1]}
columns")
```

```
This dataset contains 11123 rows and 12 columns
```

```
# Checking the info
book.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11123 entries, 0 to 11122
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   bookID              11123 non-null  int64
 1   title               11123 non-null  object
 2   authors             11123 non-null  object
 3   average_rating      11123 non-null  float64
 4   isbn                11123 non-null  object
 5   isbn13              11123 non-null  int64
 6   language_code       11123 non-null  object
 7     num_pages         11123 non-null  int64
 8   ratings_count       11123 non-null  int64
 9   text_reviews_count  11123 non-null  int64
 10  publication_date    11123 non-null  object
 11  publisher           11123 non-null  object
dtypes: float64(1), int64(5), object(6)
memory usage: 1.0+ MB
```

3.2 Handle missing values

```
# Check for missing values
print(book.isna().sum())
print("The dataset has no missing values")
```

```
bookID                0
title                 0
authors               0
average_rating        0
isbn                  0
isbn13                0
language_code         0
  num_pages           0
ratings_count         0
text_reviews_count    0
publication_date      0
publisher             0
```

```
dtype: int64
The dataset has no missing values
```

3.3 Checking for duplicates

```
# duplicated rows
print(f"The dataset contains {book.duplicated().sum()} rows")

The dataset contains 0 rows
```

3.4 Columns check-Up

3.4.1 summary plan

This recommendation system uses Content-Based Filtering, focusing primarily on the title and authors columns because they directly describe the content of the books. The title captures the book's theme or subject, while the authors indicate writing style and genre, making them the most impactful features for determining similarity. Additionally, the publisher and language_code columns can be used to provide further context.

To make the system more flexible, it is designed to accept user input from any of the four columns—whether the user knows a book title, an author, a publisher. For example, if a user wants to read books from a specific publisher, the system will find books from that publisher and recommend similar ones, ensuring a personalized and user-friendly experience.

3.4.1 Drop and rename columns

Dropping Unnecessary Columns which are irrelevant to the project renaming Some column names in the dataset which are inconsistent or unclear to improve readability and usability

```
# Checking for columns
book.columns

Index(['bookID', 'title', 'authors', 'average_rating', 'isbn',
'isbn13',
       'language_code', '  num_pages', 'ratings_count',
'text_reviews_count',
       'publication_date', 'publisher'],
      dtype='object')

# rename langauge code to langauge
book.rename(columns={"language_code":"language"}, inplace=True)

"""
Preview how many languages our dataset contains.
If it consists of books in only one language, then
drop the language column; otherwise, we will keep it
"""
book["language"].value_counts()
```

```
language
eng      8908
en-US    1408
spa       218
en-GB     214
fre       144
ger        99
jpn        46
mul        19
zho        14
grc        11
por        10
en-CA       7
ita         5
enm         3
lat         3
swe         2
rus         2
srp         1
nl          1
msa         1
glg         1
wel         1
ara         1
nor         1
tur         1
gla         1
ale         1
Name: count, dtype: int64
```

```
# listing columns to drop
irrelevant_columns = ['bookID', 'average_rating', 'isbn', 'isbn13', '
num_pages', 'ratings_count', 'text_reviews_count','publication_date',]
# drop irrelevant columns
book.drop(columns=irrelevant_columns, inplace=True)

# preview the new data
book.columns
```

```
Index(['title', 'authors', 'language', 'publisher'], dtype='object')
```

3.5 Text Preprocessing

This process involves Converting text to lowercase, Removing text in parentheses (e.g., "Book 1"), Removing special characters, and Removing stopwords like "the", "and", "of". Text processing will be conducted as follows title, authors, publishers and language respectively

```
# Preview before cleaning print 10 r0ws
book1 = book.iloc[25:36]
book1

                                                title  \
25         The Lord of the Rings: Weapons and Warfare
26   The Lord of the Rings: Complete Visual Companion
27  Agile Web Development with Rails: A Pragmatic ...
28                          Hatchet (Brian's Saga  #1)
29  Hatchet: A Guide for Using "Hatchet" in the Cl...
30  Guts: The True Stories behind Hatchet and the ...
31                       Molly Hatchet - 5 of the Best
32    Hatchet Jobs: Writings on Contemporary Fiction
33  A Changeling for All Seasons (Changeling Seaso...
34                        Changeling (Changeling  #1)
35                                 The Changeling Sea

                                              authors language  \
25      Chris   Smith/Christopher  Lee/Richard Taylor      eng
26                                        Jude Fisher      eng
27  Dave Thomas/David Heinemeier Hansson/Leon Bree...      eng
28                                       Gary Paulsen      eng
29    Donna Ickes/Edward Sciranko/Keith Vasconcelles      eng
30                                       Gary Paulsen      eng
31                                      Molly Hatchet      eng
32                                          Dale Peck    en-US
33  Angela Knight/Sahara Kelly/Judy Mays/Marteeka ...      eng
34                                      Delia Sherman      eng
35                                Patricia A. McKillip      eng

                                            publisher
25                           Houghton Mifflin Harcourt
26                           Houghton Mifflin Harcourt
27                                 Pragmatic Bookshelf
28  Atheneum Books for Young Readers: Richard Jack...
29                           Teacher Created Resources
30                                     Delacorte Press
31                           Cherry Lane Music Company
32                                       The New Press
33                                    Changeling Press
34                                     Viking Juvenile
35                                            Firebird

# Standardizing the title column
"""
This function cleans and standardizes book titles by converting them
to lowercase, removing text inside parentheses, and eliminating
special
characters. It also filters out common stopwords like "the" and "and"
to
focus on more meaningful words. This ensures the titles are cleaner,
more
focused, and easier to process for analysis and recommendation
purposes
"""
stop_words = set(stopwords.words('english'))
# defining a function
def clean_title(title):
    title = title.lower()
    title = re.sub(r"\([^)]*\)", "", title)
    title = re.sub(r"[^a-zA-Z0-9\s]", "", title)
    words = title.split()
    words = [word for word in words if word not in stop_words]
    return " ".join(words)
   # Apply the standardization function to the "title" column
book["title"] = book["title"].apply(clean_title)

# Standardizing the author column
"""
Standardizing the authors column:
To ensure consistency, we will first apply text preprocessing on the
authors' names.
Specifically, we will split multi-author entries (e.g., "J.K.
Rowling/Mary GrandPré")
and keep only the first author’s name. This way, we will have one
author name per
record, instead of listing multiple authors separated by a slash.
"""
# defining a function
def standardize_authors(authors):
    # Split to take the first author
    authors = authors.split("/")[0].strip()
    # Remove multiple spaces and replace with a single space
    authors = re.sub(r'\s+', ' ', authors)
    # Convert to lowercase for consistency
    return authors.lower()
    # Apply the standardization function to the "authors" column
book["authors"] = book["authors"].apply(standardize_authors)

# # Standardizing the publisher column
"""
This function cleans and standardizes publisher names by removing
extra spaces,
converting them to lowercase, and eliminating common suffixes like
"Inc." and "Ltd.".
This helps make the publisher names consistent, removing any
variations and ensuring
they are uniform for easier analysis and comparison.
"""
# defining a function
```

```python
def standardize_publisher(publisher):
    publisher = publisher.strip().lower()
    # Remove common suffixes like 'Inc.', 'Corporation', 'Ltd.',
'Co.', etc.
    publisher = re.sub(r"\s*(inc\.|corporation|co\.|ltd\.|company|
corp\.|\(.*\))\s*", "", publisher)
    # Remove multiple spaces and replace with a single space
    publisher = re.sub(r'\s+', ' ', publisher)

    return publisher
    # Apply the standardization function to the "publisher" column
book["publisher"] = book["publisher"].apply(standardize_publisher)

# Standardizing the language column
"""
This function cleans and standardizes language codes by converting
them to
lowercase and mapping variations (e.g., "en-US" and "en-GB") to a
common
standard. It also replaces unrecognized or multiple-language entries
with
"unknown" to ensure consistency. This helps improve data quality for
better
analysis and recommendations.
"""
# defining a function
def clean_language(language):
    # Convert to lowercase and strip extra spaces
    language = language.strip().lower()
    # Map country-specific language codes
    # for example (like 'en-US' or 'en-GB') to a general language code
('en')
    language_mapping = {
        'en-us': 'en', 'en-gb': 'en', 'en-ca': 'en', 'en': 'en',
'eng': 'en',
        'fre': 'fr', 'fra': 'fr',
        'spa': 'es', 'esp': 'es',
        'ger': 'de', 'deu': 'de',
        'por': 'pt',
        'zho': 'zh',
        'jpn': 'ja',
        'rus': 'ru',
        'ita': 'it',
        'grc': 'el',
        'gla': 'ga',
        'mul': 'mix',
    }

    # Standardize language code based on the mapping
    if language in language_mapping:
        return language_mapping[language]
    else:
    # For unrecognized or rare language codes
        return 'unknown'
 # Apply the standardization function to the "language" column
book["language"] = book["language"].apply(clean_language)
# preview the language column
book["language"].value_counts()
```

```
language
en         10537
es           218
fr           144
de            99
ja            46
mix           19
unknown       17
zh            14
el            11
pt            10
it             5
ru             2
ga             1
Name: count, dtype: int64
```

```
# Preview after text preprocessing
book2 = book.iloc[5467:5478]
book2
```

```
                                                   title  \
5467                                      breaking point   
5468             airborne guided tour airborne task force   
5469                                          red rabbit   
5470  john deere farm tractors history john deere tr...   
5471                                 corvette fifty years   
5472                                        tough tackle   
5473                               complete guide onenote   
5474                       goon show volume 4 knees fallen   
5475                                  goon show moriarty   
5476                  goon show volume 11 hes fallen water   
5477                        power die 48 gesetze der macht   

                     authors language  
publisher
5467             steve perry       en  
berkley
5468              tom clancy       en  
berkley
5469              tom clancy       en                    g.p. putnam's
sons
```

```
5470        randy leffingwell       en              motorbooks
international
5471        randy leffingwell       en
motorbooks
5472         matt christopher       en  little brown books for young
readers
5473  w. frederick zimmerman       en
apress
5474                not a book       en                    bbc physical
audio
5475                not a book       en                    bbc physical
audio
5476                not a book       en                    bbc physical
audio
5477             robert greene       de          deutscher taschenbuch
verlag
```

4.0 Feature Engineering

4.1 Combining Important Features

This function combines important book details—title, authors, and publisher—into a single text field. By merging these features, we ensure that the recommendation system captures a broader context of each book, leading to more accurate and relevant suggestions.

```
# Creating a new column
book["combined_features"] = book["title"] + " " + book["authors"] + "
" + book["publisher"] + " " + book["language"]
# preview the combined feature
book.head()

                             title       authors language   publisher
\
0    harry potter halfblood prince  j.k. rowling       en  scholastic

1       harry potter order phoenix  j.k. rowling       en  scholastic

2     harry potter chamber secrets  j.k. rowling       en  scholastic

3    harry potter prisoner azkaban  j.k. rowling       en  scholastic

4  harry potter boxed set books 15  j.k. rowling       en  scholastic


                                   combined_features
0  harry potter halfblood prince j.k. rowling sch...
1  harry potter order phoenix j.k. rowling schola...
2  harry potter chamber secrets j.k. rowling scho...
```

```
3   harry potter prisoner azkaban j.k. rowling sch...
4   harry potter boxed set books 15 j.k. rowling s...
```

4.2 Preview modified data

Since we have made several changes to the dataset, we need to check if it now contains any missing values or duplicate entries. This helps ensure the data remains clean and ready for building the recommendation system.

```
# checking missing values and duplicates
print(f"The modified data contains {book.isna().sum().sum()} missing value\n")
print(f"The modified data contains {book.duplicated().sum().sum()} duplicated rows")

The modified data contains 0 missing value

The modified data contains 113 duplicated rows

# Lets drop the duplicated values
book.drop_duplicates(inplace=True)
# preview
print(f"The modified data contains {book.duplicated().sum().sum()} duplicated rows")

The modified data contains 0 duplicated rows

# Save the cleaned dataset to a new CSV file
book.to_csv("cleaned_books.csv", index=False)
```

4.3 Applying TF-IDF Vectorization

This transforms our text data into a format where each book is represented as a list of numbers, capturing important features from the text. Each book's information is turned into a vector, allowing us to compare and analyze them easily in a mathematical space.

```
# Initialize TF-IDF Vectorizer
tfidf = TfidfVectorizer(stop_words='english')
# Transform the text into TF-IDF feature vectors
tfidf_matrix = tfidf.fit_transform(book["combined_features"])
```

4.4 Calculate Similarity Between Book

This step calculates how similar each book is to the others by comparing their feature vectors using cosine similarity. It measures the angle between the books' vectors in a high-dimensional space, with a value closer to 1 indicating high similarity and a value closer to 0 indicating low similarity.

```python
# Calculate cosine similarity between books
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
```

4.5 Create a function to get recommedations

This function takes a user's input and searches for a match in the book dataset across titles, authors, and publishers. Once a match is found, it calculates the similarity between the selected book and all other books using cosine similarity. The function then ranks the books based on similarity scores and returns the top five most similar recommendations. This approach ensures that users receive relevant book suggestions regardless of whether they input a title, author, or publisher.

```python
# defining a function
def get_recommendations_based_on_input(user_input,
cosine_sim=cosine_sim):
    # Convert user input to lowercase for consistency
    user_input = user_input.lower()
    # Try to match the user input with title, author, and publisher
    # Search in 'title' first
    idx = book[book['title'].str.contains(user_input, case=False,
na=False)].index
    if not idx.empty:
        input_type = 'title'
    else:
        # Search in 'authors' if no match in title
        idx = book[book['authors'].str.contains(user_input,
case=False, na=False)].index
        if not idx.empty:
            input_type = 'authors'
        else:
            # Search in 'publisher' if no match in title or author
            idx = book[book['publisher'].str.contains(user_input,
case=False, na=False)].index
            input_type = 'publisher' if not idx.empty else None

    if idx.empty:
        return "□ Book not found. Please try a different title,
author, or publisher."
    # Get the pairwise similarity scores of all books with the
selected book
    sim_scores = list(enumerate(cosine_sim[idx[0]]))
    # Sort the books based on similarity scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    # Get the indices of the top 10 most similar books
    sim_scores = sim_scores[1:11]
    # Get the book indices
    book_indices = [i[0] for i in sim_scores]
    # Return the top 10 most similar books
    return book['title'].iloc[book_indices]
```

4.6 Creating a user input function

Testing if it functions well, the example used test if the user inputs book title, book author, and book publishers organisation will get book titles that are simillar. and also alert the user if the book is not found

```
# Example 1 usage via title
user_input = "breaking point"
recommended_books = get_recommendations_based_on_input(user_input)
print(recommended_books)
```

```
5524            emperor
110         long shadow
5520          floodtide
5522             maiden
10361     dream kingdom
5523          oak apple
5519     tangled thread
5518       killing time
10520     bizarre world
2579            embrace
Name: title, dtype: object
```

```
# Example 2 usage via author
user_input = "randy leffingwell"
recommended_books = get_recommendations_based_on_input(user_input)
print(recommended_books)
```

```
5521         white road
110         long shadow
5520          floodtide
5522             maiden
10361     dream kingdom
5523          oak apple
5519     tangled thread
5518       killing time
10520     bizarre world
2579            embrace
Name: title, dtype: object
```

```
# Example 3 usage via publisher
user_input = "bbc physical audio"
recommended_books = get_recommendations_based_on_input(user_input)
print(recommended_books)
```

```
5529             dragons fire
5530                maelstrom
6605              dragons kin
7426               dragonsong
7427             dragonsinger
```

```
7425             dragonsong
8990    best lester del rey
6606           dragonflight
7428            dragondrums
4784                 friday
Name: title, dtype: object
```

```python
# Example 3 usage where no match was found
user_input = "wanjiru"
recommended_books = get_recommendations_based_on_input(user_input)
print(recommended_books)
```

```
▯ Book not found. Please try a different title, author, or publisher.
```

5.0 Conclusion

This content-based filtering recommendation system suggests the 10 most similar books based on user input. Users can search by book title, author name, or publisher. If the requested book is not found, the system will notify the user and prompt them to try a different title, author, or publisher.