BOOK RECOMMENDATION SYSTEM

A MINI PROJECT REPORT

Submitted by

Naboth Demetrius R (9517202109037)

Keshavaram B S (9517202109030)

Gliffy Dornick E R (9517202109018)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI

ANNA UNIVERSITY : CHENNAI 600 025

DECEMBER 2022

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI

AUTONOMOUS

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that it is the bonafide work of **"Naboth Demetrius R (95172021037), Keshavaram B S (95172021030), Gliffy Dornick E R (9517202109018)"** for the mini project titled **"BOOK RECOMMENDING SYSTEM"** in 19AD452- Artificial Intelligence Laboratory during the fourth semester January 2023 – May 2023 under my supervision.

SIGNATURE

Dr. A.Shenbagarajan, M.E.,Ph.D
Assistant Professor(SG)
Artificial Intelligence and Data Science
Mepco Schlenk Engineering College
Sivakasi- 626 005,Virudhunagar

SIGNATURE

Dr.J.Angela Jennifa Sujana, M.E.,Ph.D
Associate Professor(SG) & Head
Artificial Intelligence and Data Science
Mepco Schlenk Engineering College,
Sivakasi- 626 005,Virudhunagar

The project report submitted for the viva voce held on ………….

INTERNAL EXAMINER

EXTERNAL EXAMINER

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI

AUTONOMOUS

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that it is the bonafide work of **"Naboth Demetrius R (95172021037), Keshavaram B S (95172021030), Gliffy Dornick E R (9517202109018)"** for the mini project titled **"BOOK RECOMMENDING SYSTEM"** in 19AD451 – Data Analytics Laboratory during the fourth semester January 2023 – May 2023 under my supervision.

SIGNATURE

Dr.P.Thendral,M.E.,Ph.D
Assistant Professor(SG)
Artificial Intelligence and Data Science
Mepco Schlenk Engineering College
Sivakasi- 626 005,Virudhunagar

SIGNATURE

Dr.J.Angela Jennifa Sujana, M.E.,Ph.D
Associate Professor(SG) & Head
Artificial Intelligence and Data Science
Mepco Schlenk Engineering College,
Sivakasi- 626 005,Virudhunagar

The project report submitted for the viva voce held on ………….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

ABSTRACT

This page provides an overview of a data analytics project focused on book recommendations. The project utilizes concepts from natural language processing (NLP), machine learning, and information retrieval to recommend books based on their titles, authors, categories, and descriptions.

The code begins by importing the necessary libraries, including pandas, sklearn, and nltk. It then reads a dataset from a CSV file containing book information and performs initial data preprocessing steps. These steps involve selecting relevant columns, converting data types, splitting author names, and handling missing values.

Next, the code performs feature extraction using the CountVectorizer from sklearn. The 'tags' column, which combines authors, categories, and descriptions, is transformed into a numerical representation suitable for machine learning algorithms. Stemming is applied to further refine the textual data by reducing words to their root form.

Cosine similarity is calculated to measure the similarity between book vectors. This similarity matrix is used to build a recommendation system. Given a book title, the code retrieves its index, computes the distances to all other books, and generates a list of recommended books based on similarity scores.

Overall, this project demonstrates the application of data preprocessing, feature extraction, NLP techniques, and machine learning concepts to create an effective book recommendation system. The code can be extended and customized to handle larger datasets and incorporate additional features for enhanced recommendations.

LIST OF FIGURES

ACKNOWLEDGEMENT

First and foremost we **praise and thank "The Almighty",** the lord of all creations, who by his abundant grace has sustained us and helped us to work on this project successfully.

We really find unique pleasure and immense gratitude in thanking our respected management members**,** who is the backbone of our college.

A deep bouquet of thanks to respected Principal **Dr.S.Arivazhagan M.E.,Ph.D.,** for having provided the facilities required for our mini project.

We sincerely thank our Head of the Department **Dr. J. Angela Jennifa Sujana M.E.,Ph.D.,** Associate Professor(SG) & Head, Department of Artificial Intelligence and Data Science, for her guidance and support throughout the mini project .

We also thank our guide **Dr.A.Shenbagarajan.,M.E.,Ph.D.,** Assistant Professor(SG), **Dr.P.Thendral, M.E(Ph.D)**, Assistant Professor(SG) in Department of Artificial Intelligence and Data Science for their valuable guidance and it is great privilege to express our gratitude to them.

We extremely thank our project coordinator **Dr.A.Shenbagarajan.,M.E.,Ph.D.,** Assistant Professor(SG), **Dr.P.Thendral, M.E(Ph.D)**, Assistant Professor(SG) in Department of Artificial Intelligence and Data Science, who inspired us and supported us throughout the mini project.

We extend our heartfelt thanks and profound gratitude to all the faculty members of Artificial Intelligence and Data Science department for their kind help during our mini project work.

We also thank our parents and our friends who had been providing us with constant support during the course of the mini project work.

CHAPTER 1

INTRODUCTION

Welcome to the Book Recommendation System page. In this project, we aim to provide personalized book recommendations based on the titles, authors, categories, and descriptions of books. The system utilizes data analytics techniques, including natural language processing (NLP) and machine learning, to analyze and extract meaningful information from a dataset of books. The primary objective of this project is to assist book enthusiasts in discovering new books that align with their interests and preferences. By leveraging the power of NLP, we can uncover patterns and relationships within the textual data of books, allowing us to identify similarities between different books and make relevant recommendations.

To achieve this, we employ various data analytics concepts and techniques. The project involves data preprocessing steps to clean and format the book dataset, including handling missing values and converting data types. Additionally, feature extraction is performed using the CountVectorizer method, which transforms textual data into numerical representations suitable for machine learning algorithms.

The core of the recommendation system lies in the calculation of cosine similarity. By measuring the similarity between book vectors, we can identify books that share common characteristics and interests. The system then provides recommendations based on these similarities, helping users explore books that are likely to appeal to their tastes.

Through this project, we aim to demonstrate the application of data analytics in the domain of book recommendations. By combining NLP techniques, machine learning algorithms, and similarity measures, we strive to deliver an effective and personalized book recommendation system that enhances the reading experience and facilitates the discovery of new literary gems.

1.1.SCOPE OF THE PROJECT

The scope of the Book Recommendation System project encompasses the development of a data-driven solution that utilizes natural language processing (NLP) and machine learning techniques to provide personalized book recommendations. The project involves data preprocessing, feature extraction, and similarity measurement to analyze and understand the textual data of books. The system aims to enhance the reading experience by suggesting books that align with users' interests based on their preferences, such as authors, categories, and descriptions. The project's scope also includes the exploration of various NLP concepts, including text tokenization, stemming, and cosine similarity calculation, to build an effective recommendation system. Additionally, the project provides the flexibility to incorporate additional features and expand the dataset to enhance the accuracy and relevance of the recommendations.

1.2.OBJECTIVE OF THE PROJECT

The objective of this project is to develop a book recommendation system that provides personalized recommendations to users based on their interests. The system will leverage content-based filtering techniques to analyze various book attributes, such as titles, authors, categories, descriptions, and thumbnails. By employing data preprocessing techniques, including handling missing values and selecting relevant columns, the system will extract meaningful features from the book dataset. These features will be used to calculate the similarity between books using the cosine similarity metric. Based on the computed similarities, the system will generate recommendations by identifying books that are similar to a given target book. The recommendations will be tailored to each user's preferences, enabling them to discover new books aligned with their interests.

1.3.REPORT SUMMARY

This project report presents the development and implementation of a Book Recommendation System. The objective of the system is to provide personalized book recommendations to users based on their preferences and interests. The project utilizes data analytics techniques, including natural language processing (NLP) and machine learning, to analyze a dataset of books and extract relevant features such as titles, authors, categories, and descriptions. The system preprocesses the data, performs feature extraction using CountVectorizer, and calculates cosine similarity to measure the similarity between books. The recommendation system then generates a list of recommended books based on the user's input. The project demonstrates the effective application of NLP and machine learning concepts in building a book recommendation system that enhances the reading experience and facilitates the discovery of new books. The system can be further improved by incorporating additional features, refining the similarity calculation, and expanding the dataset. Overall, the project successfully achieves its objective of providing personalized book recommendations.

CHAPTER 2

MODULES

2.1 Pandas

Pandas is a popular Python library for data manipulation and analysis. It provides a powerful and flexible data structure called a DataFrame, which is similar to a table in a relational database. With Pandas, you can easily load, manipulate, filter, transform, and analyze data. It offers a wide range of functions and methods for tasks such as data cleaning, data merging, and data aggregation. Pandas also integrates well with other libraries, such as NumPy and Matplotlib, making it a fundamental tool for data scientists and analysts working with tabular data.

2.2 CountVectorizer

The CountVectorizer library from scikit-learn is a text feature extraction tool that converts a collection of text documents into a matrix of token counts. It represents the text data as a matrix where each row corresponds to a document and each column represents a unique word from the corpus. The CountVectorizer performs tokenization, removing punctuation and special characters, and supports options like stop word removal and n-gram generation. It is commonly used as a preprocessing step in natural language processing tasks, such as text classification and clustering, by providing a numerical representation of text data that can be fed into machine learning algorithms.

2.3 Nltk

The Natural Language Toolkit (NLTK) is a popular library for natural language processing (NLP) in Python. It provides various tools, resources, and algorithms for tasks such as tokenization, stemming, lemmatization, part-of-speech tagging, parsing, and more. NLTK offers a wide range of corpora and lexicons for training and experimentation. It also supports text classification, sentiment analysis, and language modeling. With its comprehensive functionality and easy-to-use interfaces, NLTK is a valuable resource for NLP tasks, enabling developers and researchers to process, analyze, and understand text data effectively.

2.4 PorterStemmer

The PorterStemmer library is a part of the Natural Language Toolkit (NLTK) in Python. It provides an implementation of the Porter stemming algorithm, which is a widely used technique for stemming words in natural language processing tasks. Stemming involves reducing words to their base or root form by removing suffixes, allowing different variations of a word to be treated as the same.

2.5 Cosine Similarity

Cosine similarity is a metric used to measure the similarity between two non-zero vectors in a multi-dimensional space. It calculates the cosine of the angle between the vectors, representing how closely they align with each other. The `cosine_similarity` function takes input vectors or matrices and computes pairwise cosine similarities, returning a similarity matrix. The resulting matrix provides a measure of similarity between each pair of vectors, where higher values indicate greater similarity and values close to zero indicate dissimilarity.

2.6 PIL

The Pillow library is a popular Python library for image processing tasks. It provides a wide range of functions and methods for opening, manipulating, and saving various image file formats.

2.7 Tkinter

Tkinter is a standard Python library for creating graphical user interfaces (GUIs). It provides a set of tools and widgets that allow developers to build desktop applications with interactive elements such as buttons, labels, text boxes, and more. Tkinter is based on the Tk GUI toolkit and provides a simple and intuitive way to create cross-platform GUI applications using Python.

CHAPTER 3

PROPOSED WORK

3.1 DATASET

We have taken the dataset from Kaggle that has information about books.

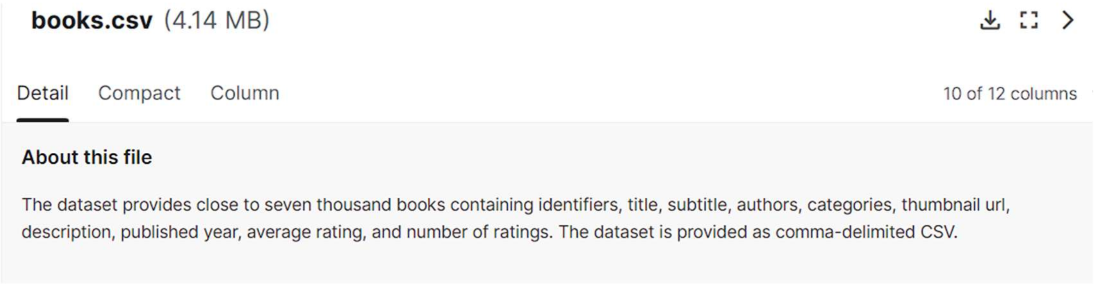
https://www.kaggle.com/datasets/dylanjcastillo/7k-books-with-metadata

The columns we use for this project are

- Title
- Category
- Summary
- Author
- Thumbnail

We create a new column that contains Category, Summary and Author in it.

We feed it to the PortStemmer and CountVectorizer, and finally find the Cosine Similarity between them



About this file

The dataset provides close to seven thousand books containing identifiers, title, subtitle, authors, categories, thumbnail url, description, published year, average rating, and number of ratings. The dataset is provided as comma-delimited CSV.

Figure 3.1.1 – Dataset Info

3.2. FLOW DIAGRAM

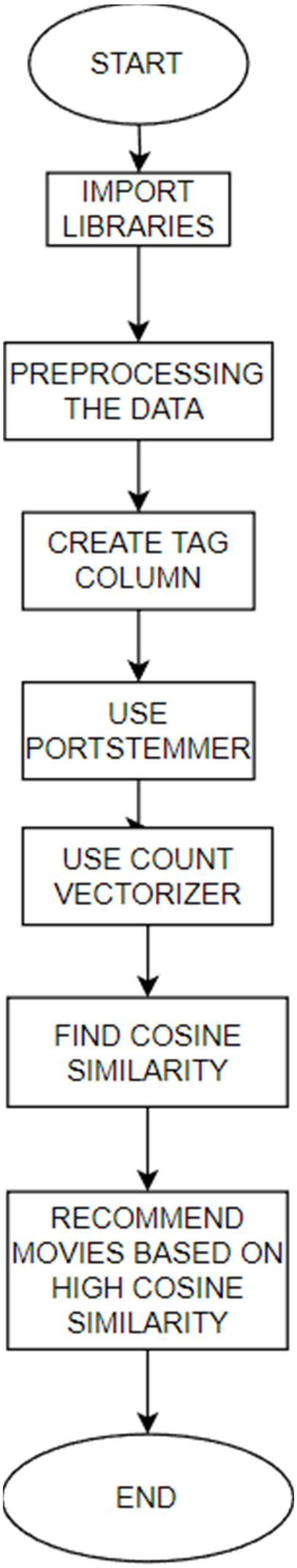


Figure 3.2.1 – Flow Chart

3.3 ARITIFICIAL INTELLIGENCE CONCEPT

1. Importing Required Libraries:

The code begins by importing the necessary libraries. It imports `CountVectorizer` from `sklearn.feature_extraction.text` module, which is used for converting text data into a numerical representation suitable for machine learning algorithms. It also imports `PorterStemmer` from `nltk.stem.porter` module for stemming words, and `cosine_similarity` from `sklearn.metrics.pairwise` module for calculating cosine similarity.

2. CountVectorizer Initialization:

- The code initializes an instance of `CountVectorizer` called `cv`. It sets the maximum number of features to 50,000 (`max_features = 50000`) and specifies 'english' as the stop words to be removed during tokenization (`stop_words = 'english'`).

3. Vectorizing Tags:

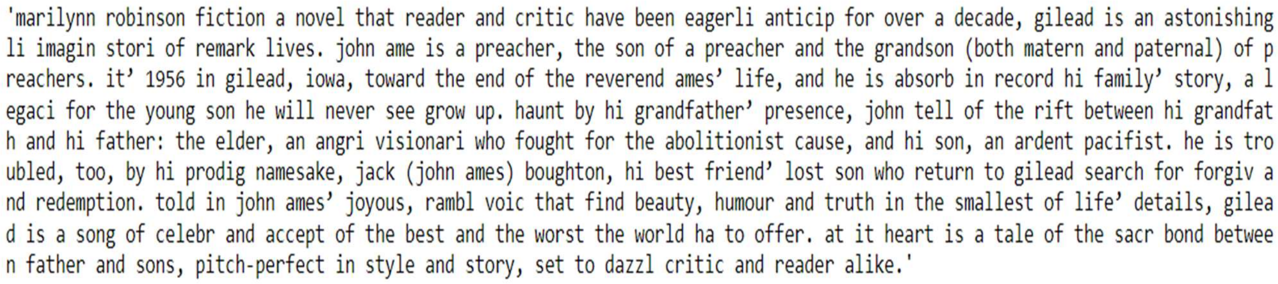
- The code transforms the 'tags' column of the `new_df` DataFrame into a matrix representation using the `fit_transform()` function of `cv`. This function converts the textual data into a sparse matrix where each row represents a book and each column represents a unique word from the 'tags' column. It then converts the sparse matrix to a dense array using `toarray()` function. The shape of the resulting array is displayed using `.shape` to show the number of rows (books) and columns (unique words).

4. Vectorization Using CountVectorizer:

- The code performs vectorization of the 'tags' column again, storing the resulting matrix in the variable `vectors`. This step is necessary as the previous line of code only displayed the shape of the transformed matrix.

5. Stemming Function Definition:

- The code defines a function called `stem()` which takes a text as input. Within the function, it initializes an empty list `y`. It then splits the text into individual words using `.split()` and applies stemming using the `PorterStemmer` instance `ps` on each word. The stemmed words are appended to the list `y`. Finally, the function joins the stemmed words with spaces and returns the result.



```
'marilynn robinson fiction a novel that reader and critic have been eagerli anticip for over a decade, gilead is an astonishing
li imagin stori of remark lives. john ame is a preacher, the son of a preacher and the grandson (both matern and paternal) of p
reachers. it’ 1956 in gilead, iowa, toward the end of the reverend ames’ life, and he is absorb in record hi family’ story, a l
egaci for the young son he will never see grow up. haunt by hi grandfather’ presence, john tell of the rift between hi grandfat
h and hi father: the elder, an angri visionari who fought for the abolitionist cause, and hi son, an ardent pacifist. he is tro
ubled, too, by hi prodig namesake, jack (john ames) boughton, hi best friend’ lost son who return to gilead search for forgiv a
nd redemption. told in john ames’ joyous, rambl voic that find beauty, humour and truth in the smallest of life’ details, gilea
d is a song of celebr and accept of the best and the worst the world ha to offer. at it heart is a tale of the sacr bond betwee
n father and sons, pitch-perfect in style and story, set to dazzl critic and reader alike.'
```

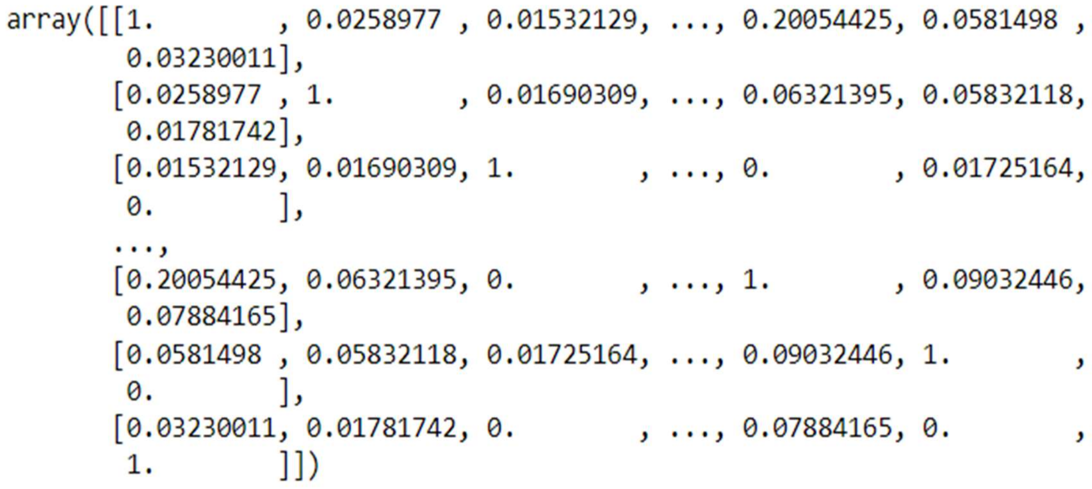
Figure 3.3.1 – Stemmed Output

6. Applying Stemming to 'tags' Column:

- The code applies the `stem()` function to the 'tags' column of the `new_df` DataFrame using the `apply()` function. This step applies stemming to each row of the 'tags' column, replacing the original text with the stemmed version.

7. Calculating Cosine Similarity:

- The code calculates the cosine similarity between the vectors using the `cosine_similarity()` function from `sklearn.metrics.pairwise`. It takes the `vectors` array as input and computes the pairwise cosine similarities between all pairs of vectors. The resulting similarity matrix is stored in the variable `similarity`.



```
array([[1.        , 0.0258977 , 0.01532129, ..., 0.20054425, 0.0581498 ,
        0.03230011],
       [0.0258977 , 1.        , 0.01690309, ..., 0.06321395, 0.05832118,
        0.01781742],
       [0.01532129, 0.01690309, 1.        , ..., 0.        , 0.01725164,
        0.        ],
       ...,
       [0.20054425, 0.06321395, 0.        , ..., 1.        , 0.09032446,
        0.07884165],
       [0.0581498 , 0.05832118, 0.01725164, ..., 0.09032446, 1.        ,
        0.        ],
       [0.03230011, 0.01781742, 0.        , ..., 0.07884165, 0.        ,
        1.        ]])
```

Figure 3.3.2 – Cosine Similarity

8. Displaying Shape of Similarity Matrix:

- The code displays the shape of the similarity matrix using `.shape`. This provides information about the number of rows and columns in the matrix, indicating the number of books and their pairwise similarities.

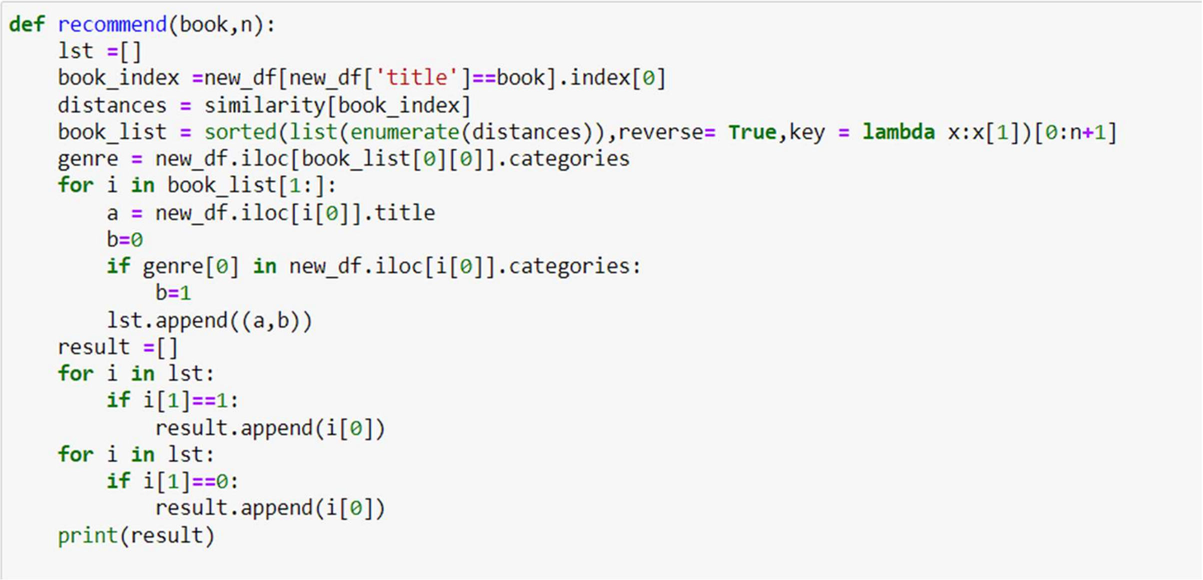
9. Recommendation Function:

- The code defines a function called `recommend()` that takes two inputs: `book` (the title of the book for which recommendations are sought) and `n` (the number of book recommendations to provide).

- Inside the function, it finds the index of the book in the `new_df` DataFrame using the `title` column and assigns it to the variable `book_index`.

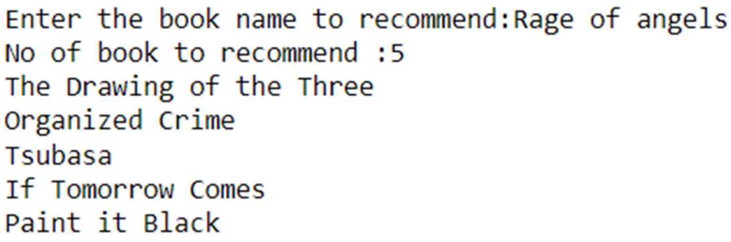
- It then retrieves the distances/similarities between the book at `book_index` and all other books from the `similarity` matrix.

- And then it displays the similar books with same genre first and the other behind it.



```
def recommend(book,n):
    lst =[]
    book_index =new_df[new_df['title']==book].index[0]
    distances = similarity[book_index]
    book_list = sorted(list(enumerate(distances)),reverse= True,key = lambda x:x[1])[0:n+1]
    genre = new_df.iloc[book_list[0][0]].categories
    for i in book_list[1:]:
        a = new_df.iloc[i[0]].title
        b=0
        if genre[0] in new_df.iloc[i[0]].categories:
            b=1
        lst.append((a,b))
    result =[]
    for i in lst:
        if i[1]==1:
            result.append(i[0])
    for i in lst:
        if i[1]==0:
            result.append(i[0])
    print(result)
```

Figure 3.3.3 – Recommend Function



```
Enter the book name to recommend:Rage of angels
No of book to recommend :5
The Drawing of the Three
Organized Crime
Tsubasa
If Tomorrow Comes
Paint it Black
```

Figure 3.3.4 – Recommendations

3.4 DATA ANALYTICS CONCEPTS

1. Importing Required Libraries:

- The code begins by importing the pandas library using the alias `pd`. This library provides various data manipulation and analysis tools for working with structured data.

2. Reading the CSV File:

- The code reads a CSV file named "books.csv" using the `pd.read_csv()` function. The data from the CSV file is loaded into a pandas DataFrame called `books_df`.

3. Displaying the First Few Rows of the DataFrame:

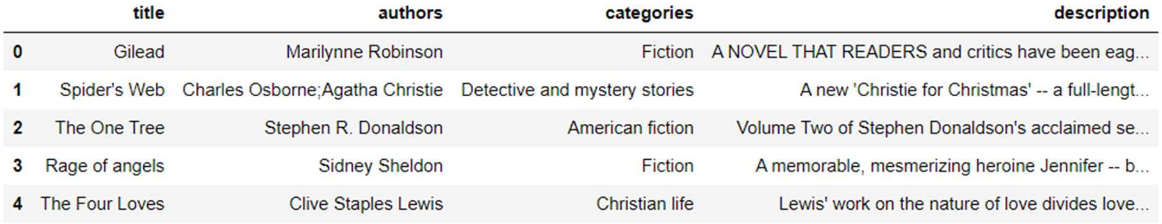
- The code calls the `head()` function on the `books_df` DataFrame to display the first few rows of the dataset. This helps in getting a quick overview of the data and its structure.

4. Displaying Information About the DataFrame:

- The code calls the `info()` function on the `books_df` DataFrame to display information about the dataset, including the column names, data types, and the number of non-null values in each column. This provides a summary of the DataFrame's structure and helps in identifying missing values.

5. Selecting Specific Columns:

- The code selects specific columns ('title', 'authors', 'categories', 'description') from the `books_df` DataFrame using indexing. The resulting DataFrame contains only these selected columns.



	title	authors	categories	description
0	Gilead	Marilynne Robinson	Fiction	A NOVEL THAT READERS and critics have been eag...
1	Spider's Web	Charles Osborne;Agatha Christie	Detective and mystery stories	A new 'Christie for Christmas' -- a full-lengt...
2	The One Tree	Stephen R. Donaldson	American fiction	Volume Two of Stephen Donaldson's acclaimed se...
3	Rage of angels	Sidney Sheldon	Fiction	A memorable, mesmerizing heroine Jennifer -- b...
4	The Four Loves	Clive Staples Lewis	Christian life	Lewis' work on the nature of love divides love...

Figure 3.4.1 – Selected Columns

6. Handling Missing Values:

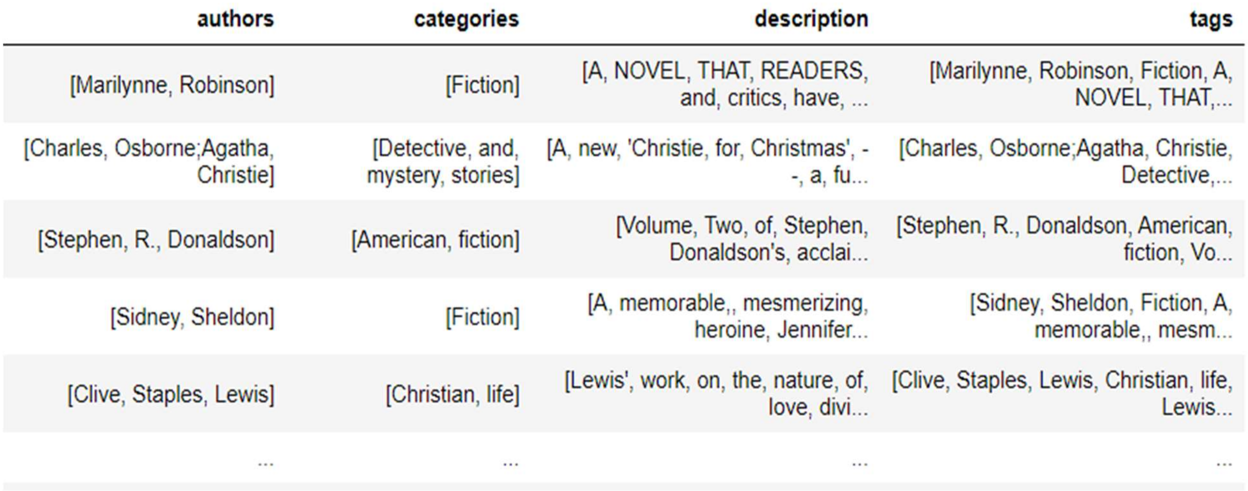
- The code calls the `dropna()` function on the `books_df` DataFrame to drop rows that contain any missing values (NaN). This step ensures that the dataset does not contain incomplete or inconsistent data.

7. Splitting Text Data into Lists:

- The code applies transformations to specific columns ('authors', 'description', 'categories') in the `books_df` DataFrame.
- For the 'authors' column, it applies the `split()` function on each value, which splits the string into a list of individual author names.
- Similarly, for the 'description' and 'categories' columns, it splits the string values into lists of words using the `split()` function.

8. Creating a 'Tags' Column:

- The code creates a new column 'tags' in the `books_df` DataFrame by concatenating the 'authors', 'categories', and 'description' columns using the '+' operator.
- This step combines the different textual features of a book (authors, categories, and description) into a single list called 'tags'.



authors	categories	description	tags
[Marilynne, Robinson]	[Fiction]	[A, NOVEL, THAT, READERS, and, critics, have, ...	[Marilynne, Robinson, Fiction, A, NOVEL, THAT,...
[Charles, Osborne;Agatha, Christie]	[Detective, and, mystery, stories]	[A, new, 'Christie, for, Christmas', --, a, fu...	[Charles, Osborne;Agatha, Christie, Detective,...
[Stephen, R., Donaldson]	[American, fiction]	[Volume, Two, of, Stephen, Donaldson's, acclai...	[Stephen, R., Donaldson, American, fiction, Vo...
[Sidney, Sheldon]	[Fiction]	[A, memorable,, mesmerizing, heroine, Jennifer...	[Sidney, Sheldon, Fiction, A, memorable,, mesm...
[Clive, Staples, Lewis]	[Christian, life]	[Lewis', work, on, the, nature, of, love, divi...	[Clive, Staples, Lewis, Christian, life, Lewis...
...	...	...	...

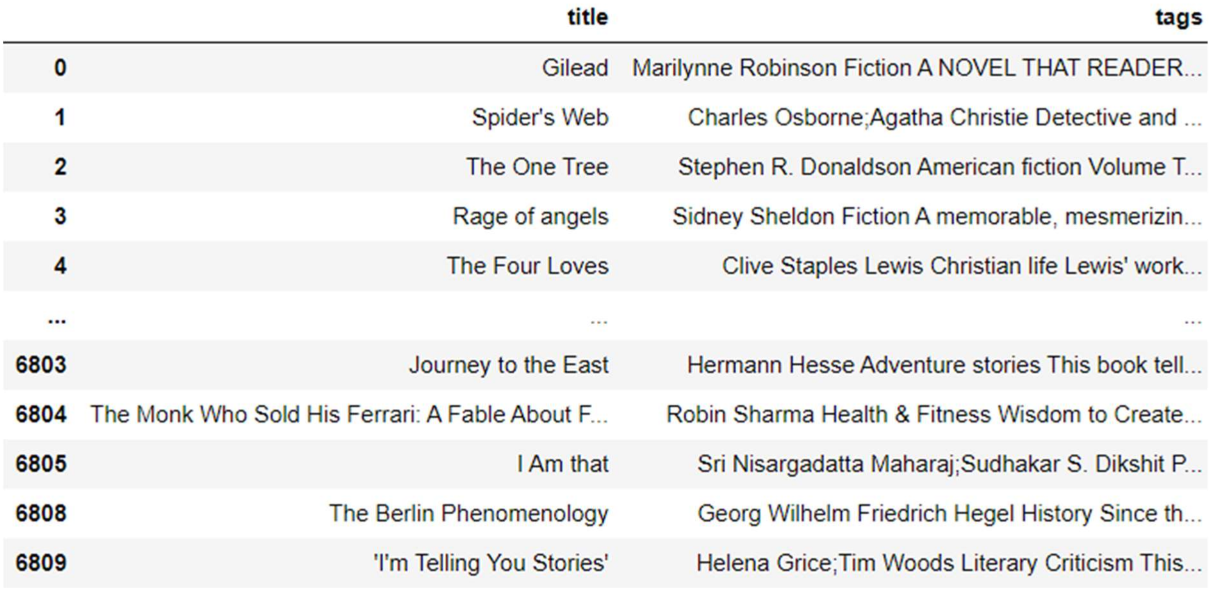
Figure 3.4.2 – Tags Column

9. Creating a New DataFrame:

- The code creates a new DataFrame called `new_df` by selecting only the 'title' and 'tags' columns from the `books_df` DataFrame.

10. Joining List Elements:

- The code applies the `join()` function on the 'tags' column of the `new_df` DataFrame to join the elements of each list into a single string. The resulting 'tags' column contains space-separated words.



	title	tags
0	Gilead	Marilynne Robinson Fiction A NOVEL THAT READER...
1	Spider's Web	Charles Osborne;Agatha Christie Detective and ...
2	The One Tree	Stephen R. Donaldson American fiction Volume T...
3	Rage of angels	Sidney Sheldon Fiction A memorable, mesmerizin...
4	The Four Loves	Clive Staples Lewis Christian life Lewis' work...
...	...	...
6803	Journey to the East	Hermann Hesse Adventure stories This book tell...
6804	The Monk Who Sold His Ferrari: A Fable About F...	Robin Sharma Health & Fitness Wisdom to Create...
6805	I Am that	Sri Nisargadatta Maharaj;Sudhakar S. Dikshit P...
6808	The Berlin Phenomenology	Georg Wilhelm Friedrich Hegel History Since th...
6809	'I'm Telling You Stories'	Helena Grice;Tim Woods Literary Criticism This...

Figure 3.4.3 – Tags Column After Joining

11. Displaying the Modified DataFrame:

- The code displays the modified `new_df` DataFrame, which now contains the 'title' and 'tags' columns with the updated values.

12. Accessing a Specific Value:

- The code retrieves the value at the first row and 'tags' column of the `new_df` DataFrame using the indexing notation `new_df['tags'][0]`. This provides the concatenated tags for the first book in the dataset

13. Feeding the Data to NLTK:

- We apply various methods from NLTK on the tags column which we have clearly explained in the section above.

CHAPTER 4

IMPLEMENTATION

4.1 RECOMMENDING SYSTEM

```
import pandas as pd
books_df= pd.read_csv("books.csv")
books_df.head()
books_df.info()
books_df = books_df[['title','authors','categories','description']]
books_df.head()
books_df['authors'] = books_df['authors'].apply(lambda x:str(x))
books_df['authors'] = books_df['authors'].apply(lambda x:x.split())
books_df
books_df.dropna(inplace=True)
books_df['description'] = books_df['description'].apply(lambda
x:x.split())
books_df
books_df['categories'] = books_df['categories'].apply(lambda
x:x.split())
books_df
books_df['tags'] =
books_df['authors']+books_df['categories']+books_df['description']
books_df
new_df = books_df[['title','tags']]
new_df['tags']=new_df['tags'].apply(lambda x: ' '.join(x))
new_df
new_df['tags'][0]

from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features = 50000, stop_words = 'english')
cv.fit_transform(new_df['tags']).toarray().shape
vectors = cv.fit_transform(new_df['tags']).toarray()
```

```
vectors[0]
import nltk
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()

def stem(text):
    y=[]
    for i in text.split():
        y.append(ps.stem(i))
    return ' '.join(y)
new_df['tags'] = new_df['tags'].apply(stem)
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(vectors)
cosine_similarity(vectors).shape
similarity = cosine_similarity(vectors)
new_df

def recommend(book,n):
    lst =[]
    book_index =new_df[new_df['title']==book].index[0]
    distances = similarity[book_index]
    book_list = sorted(list(enumerate(distances)),reverse= True,key =
lambda x:x[1])[0:n+1]
    genre = new_df.iloc[book_list[0][0]].categories
    for i in book_list[1:]:
        a = new_df.iloc[i[0]].title
        b=0
        if genre[0] in new_df.iloc[i[0]].categories:
            b=1
        lst.append((a,b))
    result =[]
    for i in lst:
        if i[1]==1:
            result.append(i[0])
    for i in lst:
        if i[1]==0:
            result.append(i[0])
    print(result)
recommend(input('Enter the book name to recommend:'),int(input('No of
book to recommend :')))
```

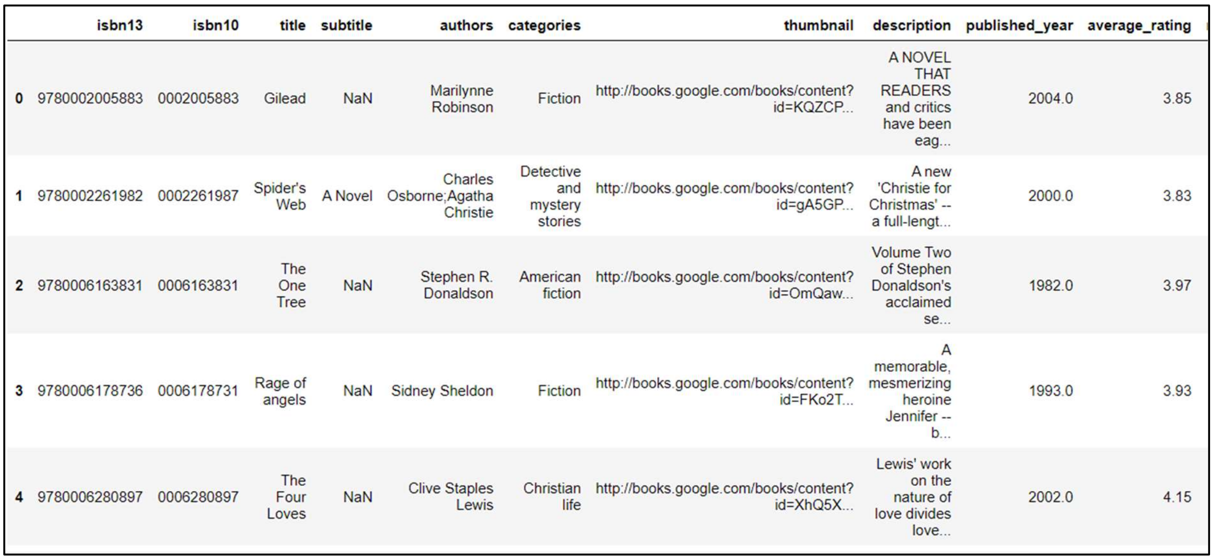
4.2 GUI

```
from tkinter import *
from PIL import Image, ImageTk
import urllib.request
from io import BytesIO
import main

# Define the function to display book results
def show_book_results():
    # Get book name from the Entry widget
    book_name = book_entry.get()
    details = main.recom(book_name)
    names = details[0]
    links = details[1]

    # Create a new window for the book results
    book_window = Toplevel(root)
    book_window.title("Book Results")
    book_window.geometry("800x400")
    book_window.resizable(True,True)
    book_window.configure(bg='black')

    # Load and resize background image
    # bg_image = Image.open("bg.png")
    # bg_image = bg_image.resize((800, 600), Image.ANTIALIAS)
    # bg_photo = ImageTk.PhotoImage(bg_image)
    # bg_label = Label(book_window, image=bg_photo)
    # bg_label.place(x=0, y=0)

    # Create image and text labels for book results
    for i in range(4):
        # Get image from URL and create PhotoImage object
        image_url = links[i]
        with urllib.request.urlopen(image_url) as url:
            image_data = url.read()
        image = Image.open(BytesIO(image_data))
        image = image.resize((150, 200), Image.ANTIALIAS)
        photo = ImageTk.PhotoImage(image)

        # Create image label and place it on the window
        image_label = Label(book_window, image=photo)
        image_label.image = photo
        image_label.place(relx=0.2+i*0.2, rely=0.3, anchor=CENTER)

    text = Label(book_window, text="It is Highly recommended for you
to read these books too!", font=("Helvetica", 16), bg="white")
    text.place(relx=0.5, rely=0.4, anchor=CENTER)
# Create the main window
root = Tk()
root.title("Book Search")
root.geometry("600x600")
root.resizable(False, False)

# Load and resize background image
bg_image = Image.open("bg.png")
bg_image = bg_image.resize((600, 600), Image.ANTIALIAS)
bg_photo = ImageTk.PhotoImage(bg_image)
```

```
bg_label = Label(root, image=bg_photo)
bg_label.place(x=0, y=0)

# Create label and entry widget for book search
book_label = Label(root, text="Enter book name:", font=("Helvetica",
16), bg="white")
book_label.place(relx=0.5, rely=0.4, anchor=CENTER)
book_entry = Entry(root, width=30, font=("Helvetica", 12))
book_entry.place(relx=0.5, rely=0.5, anchor=CENTER)

# Create curved button for book search
search_button = Button(root, text="Search", font=("Helvetica", 12),
bg="white", bd=0, command=show_book_results)
search_button.place(relx=0.5, rely=0.6, anchor=CENTER)

# Run the main loop
root.mainloop()
```

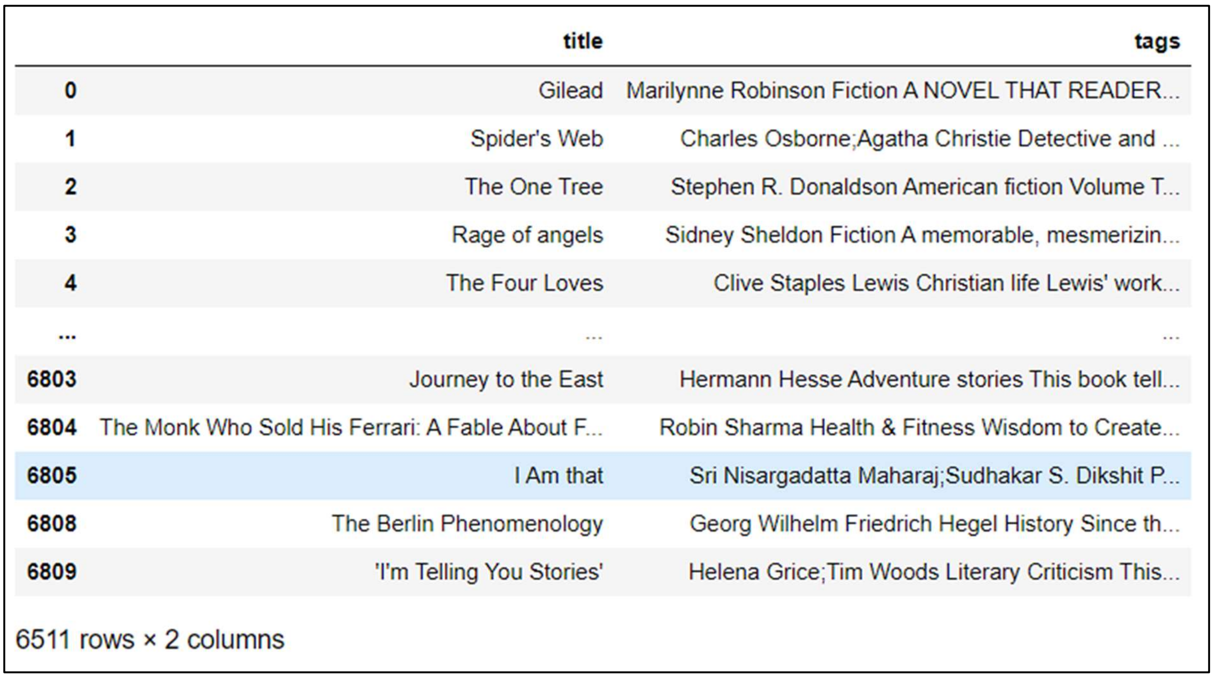
CHAPTER 5

RESULTS :



	isbn13	isbn10	title	subtitle	authors	categories	thumbnail	description	published_year	average_rating
0	9780002005883	0002005883	Gilead	NaN	Marilynne Robinson	Fiction	http://books.google.com/books/content?id=KQZCP...	A NOVEL THAT READERS and critics have been eag...	2004.0	3.85
1	9780002261982	0002261987	Spider's Web	A Novel	Charles Osborne;Agatha Christie	Detective and mystery stories	http://books.google.com/books/content?id=gA5GP...	A new 'Christie for Christmas' -- a full-lengt...	2000.0	3.83
2	9780006163831	0006163831	The One Tree	NaN	Stephen R. Donaldson	American fiction	http://books.google.com/books/content?id=OmQaw...	Volume Two of Stephen Donaldson's acclaimed se...	1982.0	3.97
3	9780006178736	0006178731	Rage of angels	NaN	Sidney Sheldon	Fiction	http://books.google.com/books/content?id=FKo2T...	A memorable, mesmerizing heroine Jennifer -- b...	1993.0	3.93
4	9780006280897	0006280897	The Four Loves	NaN	Clive Staples Lewis	Christian life	http://books.google.com/books/content?id=XhQ5X...	Lewis' work on the nature of love divides love...	2002.0	4.15

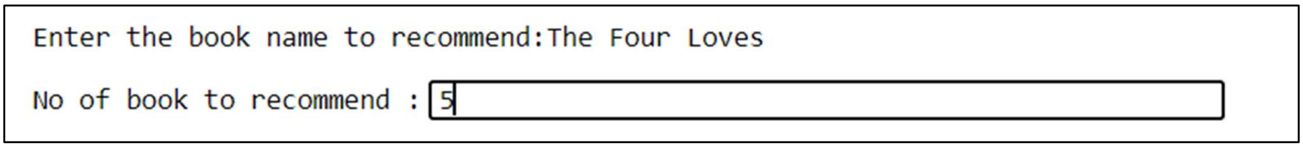
Figure 5.1 – Raw Dataset



	title	tags
0	Gilead	Marilynne Robinson Fiction A NOVEL THAT READER...
1	Spider's Web	Charles Osborne;Agatha Christie Detective and ...
2	The One Tree	Stephen R. Donaldson American fiction Volume T...
3	Rage of angels	Sidney Sheldon Fiction A memorable, mesmerizin...
4	The Four Loves	Clive Staples Lewis Christian life Lewis' work...
...	...	...
6803	Journey to the East	Hermann Hesse Adventure stories This book tell...
6804	The Monk Who Sold His Ferrari: A Fable About F...	Robin Sharma Health & Fitness Wisdom to Create...
6805	I Am that	Sri Nisargadatta Maharaj;Sudhakar S. Dikshit P...
6808	The Berlin Phenomenology	Georg Wilhelm Friedrich Hegel History Since th...
6809	'I'm Telling You Stories'	Helena Grice;Tim Woods Literary Criticism This...

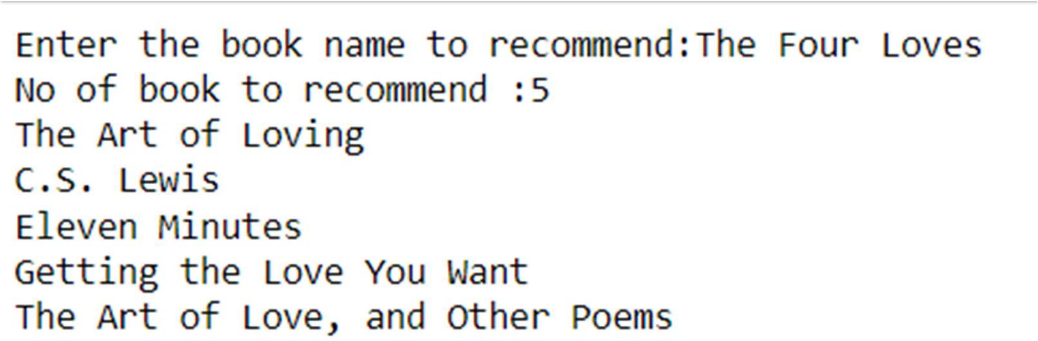
6511 rows × 2 columns

Figure 5.2 – Dataset after pre-processing



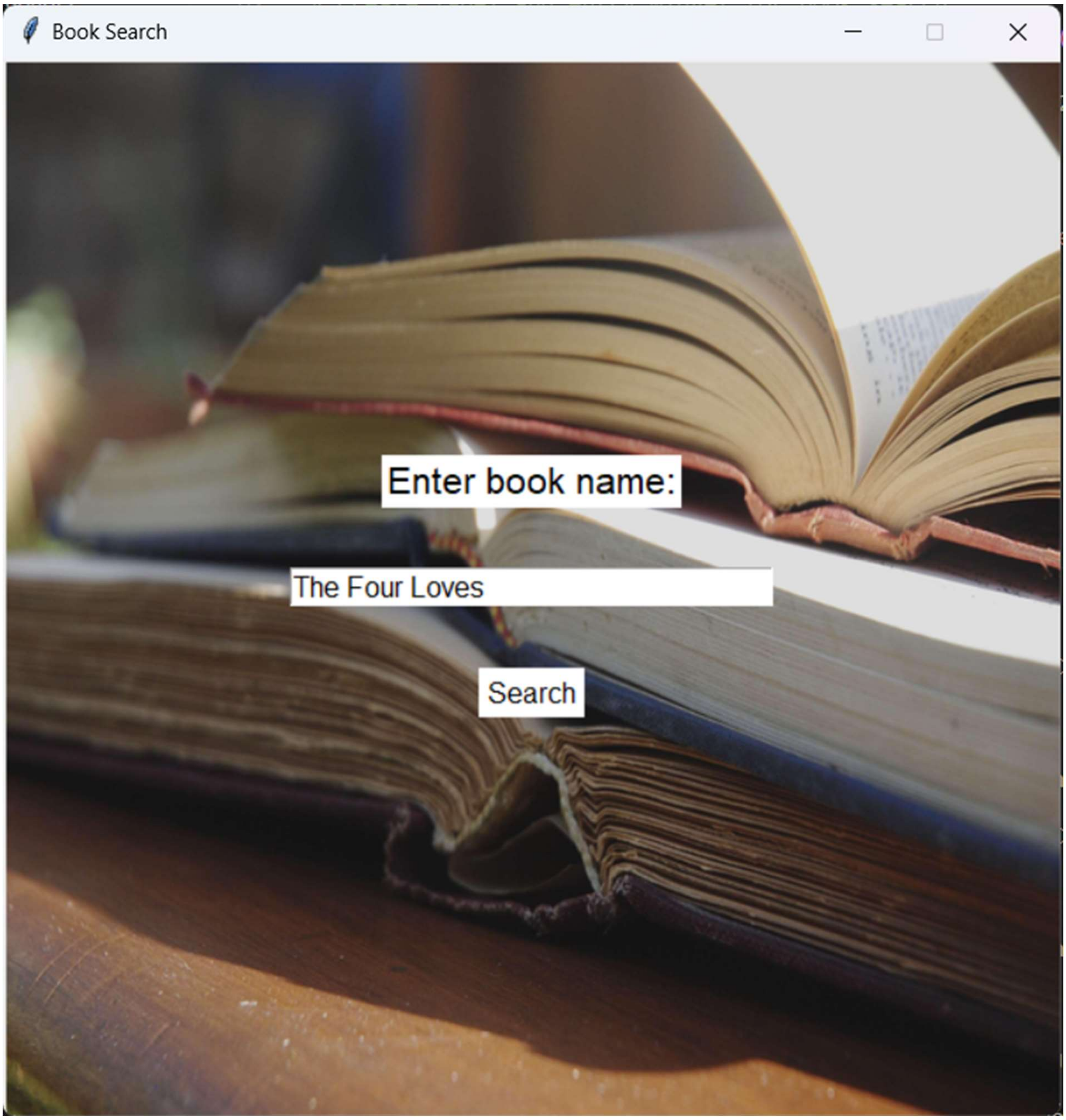
```
Enter the book name to recommend:The Four Loves

No of book to recommend : 5
```

Figure 5.3 – Input for recommending



```
Enter the book name to recommend:The Four Loves
No of book to recommend :5
The Art of Loving
C.S. Lewis
Eleven Minutes
Getting the Love You Want
The Art of Love, and Other Poems
```

Figure 5.4 – Output recommendation



The Four Loves

Figure 5.5 – GUI Home window

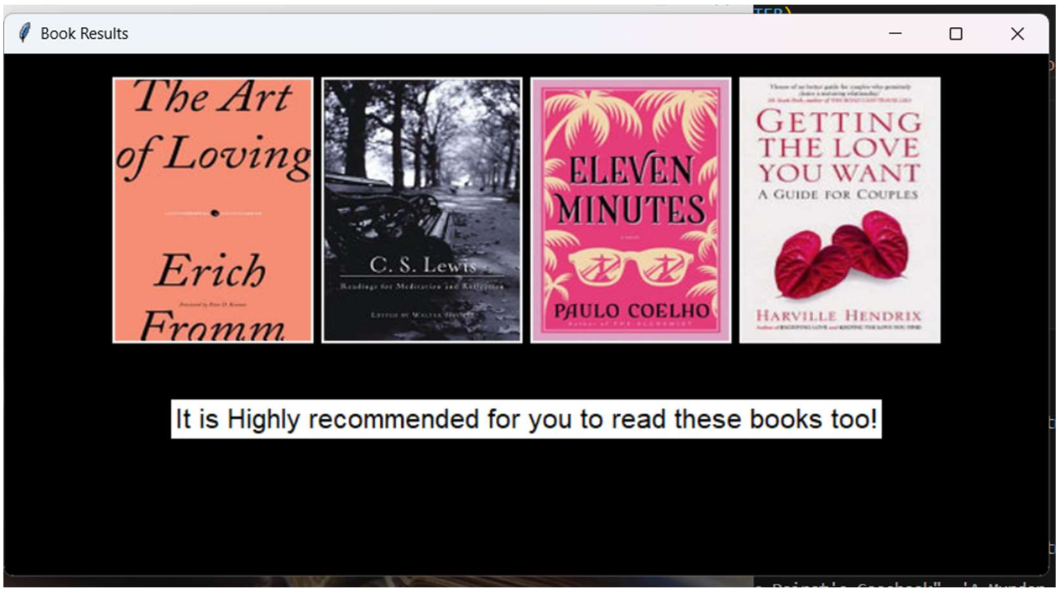


Figure 5.6 – Recommendation window

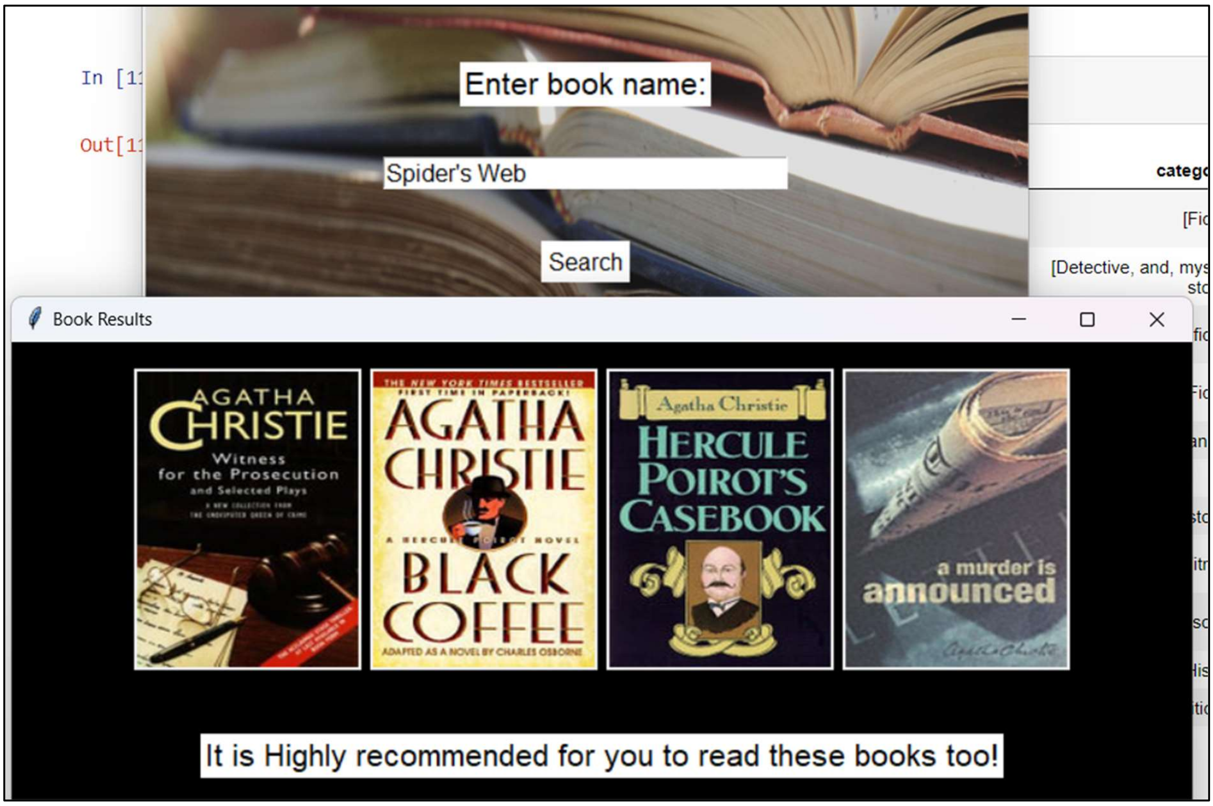


Figure 5.7 – Example 1

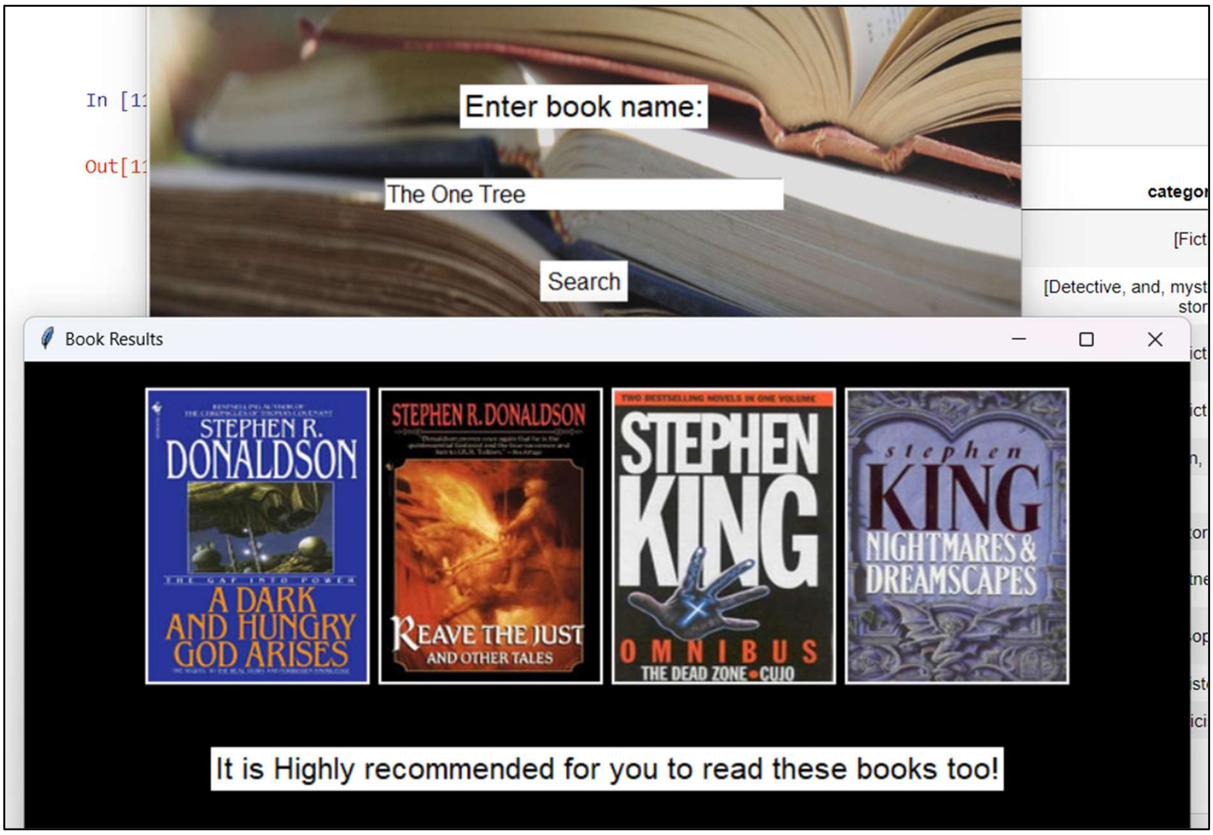


Figure 5.8 – Example 2

CONCLUSION

In conclusion, the Book Recommendation System project has successfully developed an intelligent system that leverages data analytics techniques to provide personalized book recommendations. By implementing natural language processing (NLP) and machine learning algorithms, the system effectively analyzes book data, including titles, authors, categories, and descriptions, to understand user preferences and generate relevant recommendations. The project demonstrates the importance of data preprocessing, feature extraction, and similarity measurement in building an accurate recommendation system. The system's ability to identify similarities between books and suggest relevant titles enhances the reading experience for users, helping them discover new books aligned with their interests. Furthermore, the project highlights the potential for further improvements, such as incorporating additional features and expanding the dataset, to enhance the system's accuracy and relevance. Overall, this project successfully showcases the application of data analytics in the realm of book recommendations, offering valuable insights and opportunities for future enhancements in the field.

REFERENCES

- https://www.nvidia.com/en-us/glossary/data-science/recommendation
- https://developers.google.com/machine-learning/recommendation
- https://towardsdatascience.com/introduction-to-recommender-systems
- https://builtin.com/data-science/recommender-systems
- https://www.xomnia.com/post/recommender-engines-a-different-approach-with-network-science/