Book Recommendation System Using IBM Watson

1. INTRODUCTION

1.1 OVERVIEW

A recommendation system is one of the top applications of data science. Every consumer Internet company requires a recommendation system like Netflix, YouTube, a news feed, etc. In this age of information overload, people use a variety of strategies to make choices about what to buy, how to spend their leisure time, and even where to go. Recommendation systems automate some of these strategies with the goal of providing affordable, personal and high-quality recommendations. So, with the help of different techniques like collaborative and content-based filtering, we can build our own recommendation systems. The main objective of the project is to build an application where the user is prompted with various types of book recommendations based on the input given by the user.A recommendation system helps an organization to create loyal customers and build trust by them desired products and services for which they came on your site. The recommendation system today are so powerful that they can handle the new customer too who has visited the site for the first time. They recommend the products which are currently trending or highly rated and they can also recommend the products which bring maximum profit to the company.

1.2 PURPOSE

This Online book selling websites helps to buy the books online with Recommendation system which is one of the stronger tools to increase profit and retaining buyer. The book recommendation system must recommend

books that are of buyer's-interest. Recommendation systems are widely used to recommend products to the end users that are most appropriate. This system uses features of collaborative filtering to produce efficient and

effective recommendations. Collaborative recommendation is probably the most familiar, most widely implemented and most mature of the technologies. Collaborative recommender systems aggregate ratings of objects, recognize commonalities between users on the basis of their ratings, and generate new recommendations.

2. LITERATURE SURVEY

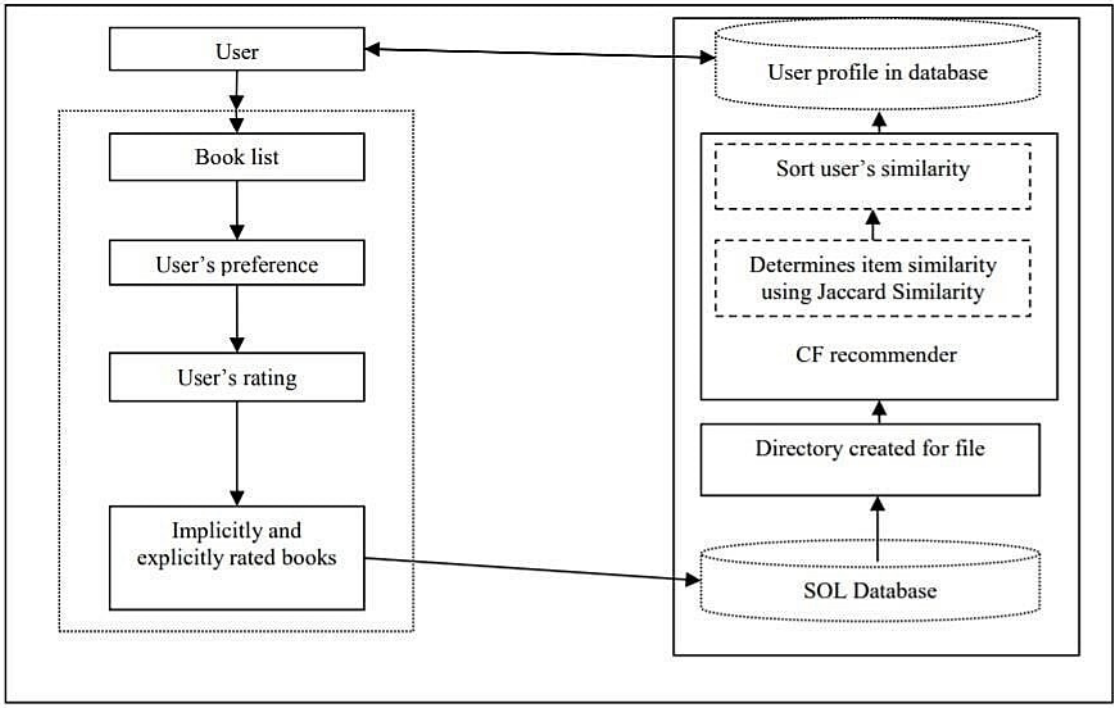
2.1 EXISTING PROBLEM

Following are some of the existing book recommendation engines used by the top rated book purchasing websites. The existing engines make use of conventional algorithms for recommendations. In Content based Recommendation Engine, system generates recommendations from source based on the features associated with products and the users information. Content-based recommender treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on product features. In Collaborative recommendation engines, suggestions are generated on the basis of ratings given by group of people. It locates peer users with a rating history similar to the current user and generates recommendations for the user. In Context based Recommendation Engine, system requires the additional data about the context of item consumption like time,mood and behavioral aspects. These data may be used to improve the recommendation compared to what could be performed without this additional source of information.

2.2 PROPOSED SOLUTION

Existing recommendation services despite their powerlessness need a strong user profile information and history. User register to such systems, browse books, rate them, write their feedbacks, recommend to others, share, read appropriate information and etc. Based on such an information a system makes its recommendations. The examples of such services are which book.net, whatshouldireadnext.com, lazylibrary.com and etc. Instead our recommender system focuses on simplicity and speed. The user makes a registration and is asked to select 10 favorite books from at least 3 categories (genres). Based on this information the system makes recommendations. Further the user can continue to rate the books, buy them and add them to read list and thus allow to improve the quality of recommendations. The system overview is demonstrated in Fig. 1. A user, using an intuitive search and filtering interface updates a database by rating the books and then gets appropriate recommendations. The recommendations in turn are calculated based on collaborative filtering method.

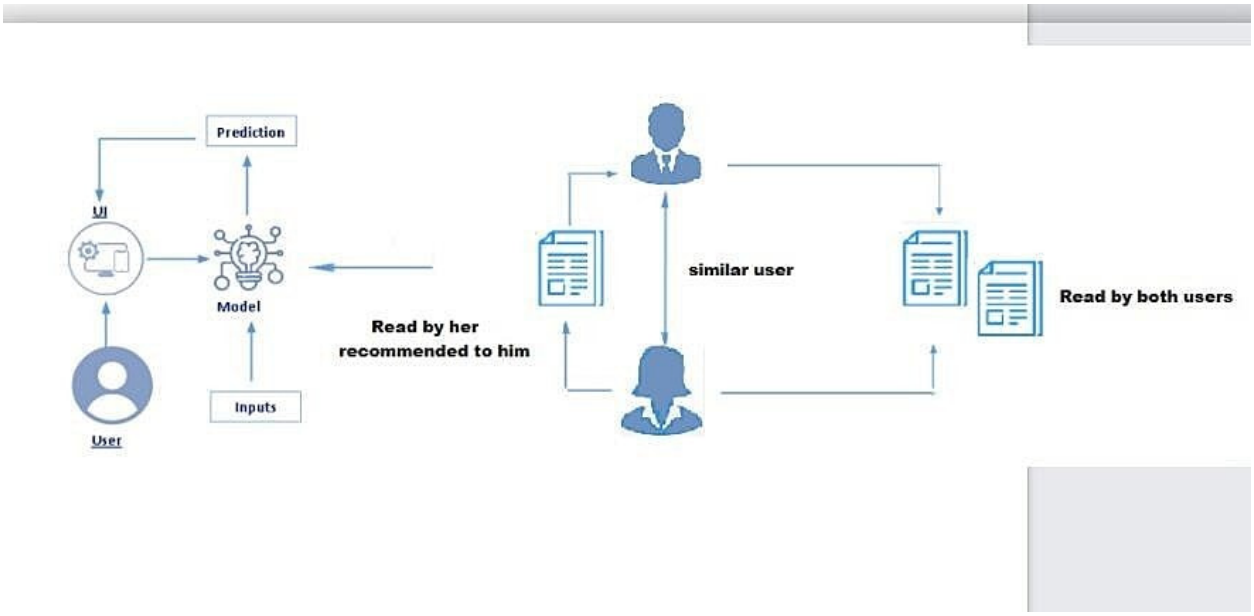
3.



THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

ARCHITECTURE



3.2 HARDWARE AND SOFTWARE REQUIREMENT:

Software Requirement:

REQUIREMENT	SPECIFICATION
Anaconda Navigator	You must have anaconda installed in your device prior to begin.
Spyder, Jupyter Notebook, Flask Frame work	1. One should have Spyder and Jupyter notebook. 2. One should install flask framework through anaconda prompt for running their web application 3. We need to build the model using jupyter notebook with all the imported packages.
Web browser	For all Web browsers, the following must be enabled: • cookies • JavaScript

Hardware Requirement:

REQUIREMENT	SPECIFICATION
Operating system	Microsoft Windows UNIX Linux®
Processing	Minimum: 4 CPU cores for one user. For each deployment, a sizing exercise is highly recommended.
RAM	Minimum 8 GB.
Operating system specifications	File descriptor limit set to 8192 on UNIX and Linux
Disk space	A minimum of 7 GB of free space is required to install the software.

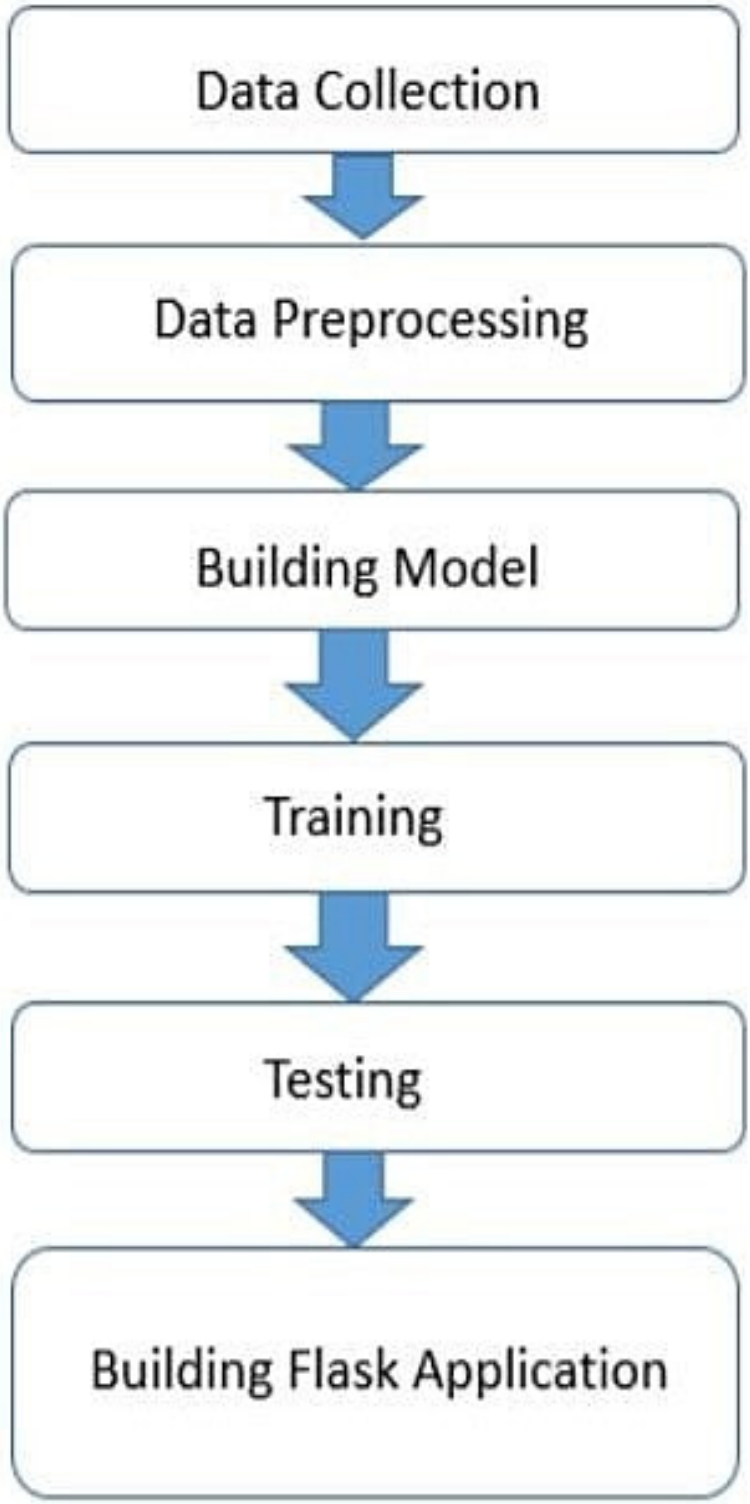
4. EXPERIMENTAL INVESTIGATIONS:

Online recommendation systems are the in thing to do for many e-commerce websites. A recommendation system broadly recommends products to customers best suited to their tastes and traits. For more details on recommendation systems, read my introductory post on Recommendation Systems and a few illustrations using Python.

My journey to building Book Recommendation System began when I came across Book Crossing dataset. This dataset has been compiled by Cai-Nicolas Ziegler in 2004, and it comprises of three tables for users, books and ratings. Explicit ratings are expressed on a scale from 1–10 (higher values denoting higher appreciation) and implicit rating is expressed by 0.Before building any machine learning model, it is vital to understand what the data is, and what are we trying to achieve. Data exploration reveals the hidden trends and insights and data preprocessing makes the data ready for use by ML algorithms.

5. FLOW CHART:

The following diagram shows the control flow of the solution



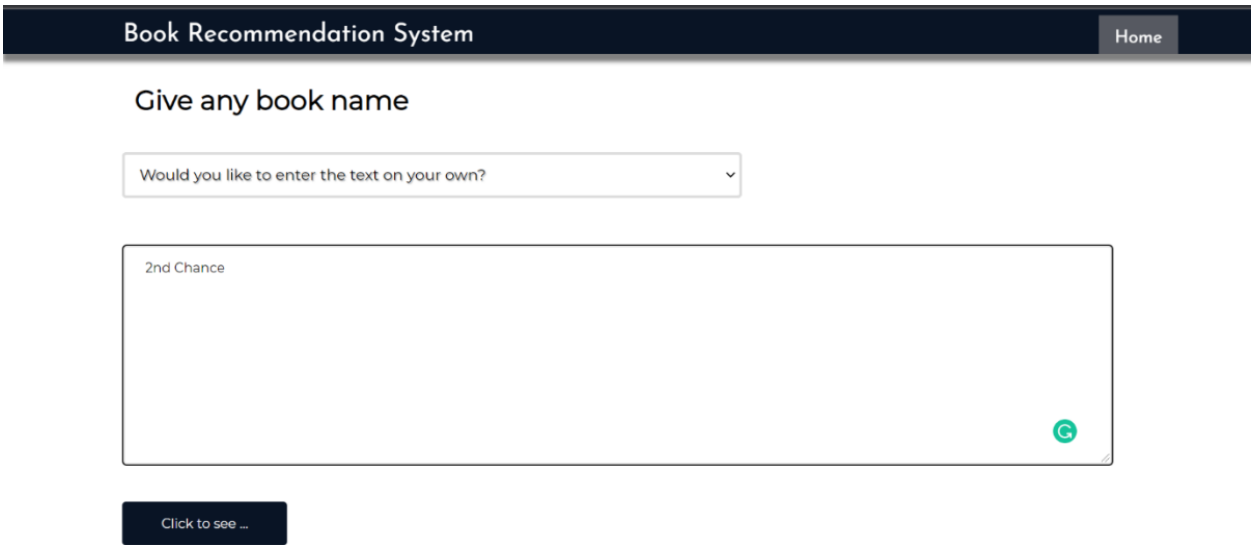
6. RESULT:

Here we used K-Nearest Neighbor to work out on a book recommendation system using IBM. where the user is prompted with various types of book recommendations based on the input given by the user.

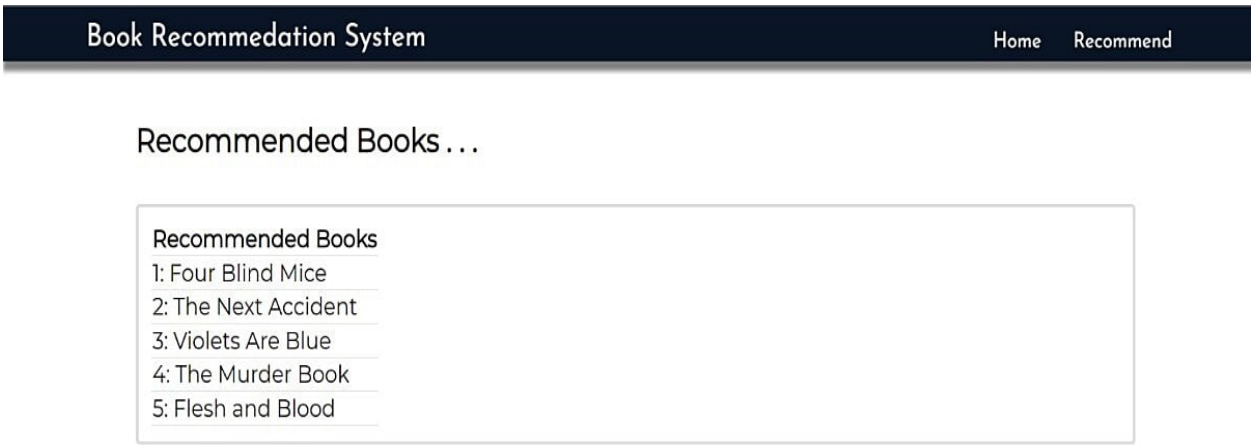
This can be checked by giving the details of any book we want like name of the author etc.., and search.

The output is as follows:

On selecting “Recommend” we will get directed to the following page:



After giving a sutiable input we get the following output:



7. ADVANTAGES & DISADVANTAGES:

Advantages:

(i)Drive Traffic : A recommendation engine can bring traffic to your site. It accomplishes this with customized e-mail messages and targeted blasts.

(ii) Engage Customers : Consumers end up being more engaged in the website when individualized item recommendations are made. They are able to dive even more deeply into the product line without needing to carry out search after search.

(iii) Boost Number of Items per Order : In addition to the average order value rising, the number of products per order likewise typically increases when a recommendation engine is used. When the customer is revealed options that fulfill his interest, he is most likely to add choices to his purchase.

(iv) Offer Recommendations and Direction : An experienced carrier can provide suggestions on ways to utilize the data gathered and reported to the customer. Acting as a partner and a consultant, the supplier needs to have the expertise to assist direct the ecommerce site to a prosperous future.

Disadvantages:

(i)The cold-start problem

Collaborative filtering systems are based on the action of available data from similar users. If you are building a brand new recommendation system, you would have no user data to start with. You can use content-based filtering first and then move on to the collaborative filtering approach

(ii) Scalability

As the number of users grow, the algorithms suffer scalability issues. If you have 10 million customers and 100,000 movies, you would have to create a sparse matrix with one trillion elements.

(iii) Synonymy

Synonymy is the tendency of very similar items to have different names or entries. Most recommender systems find it difficult to make distinction between closely related items.Collaborative Filtering systems usually find no match between the two terms to be able to compute their similarity.

8. APPLICATIONS:

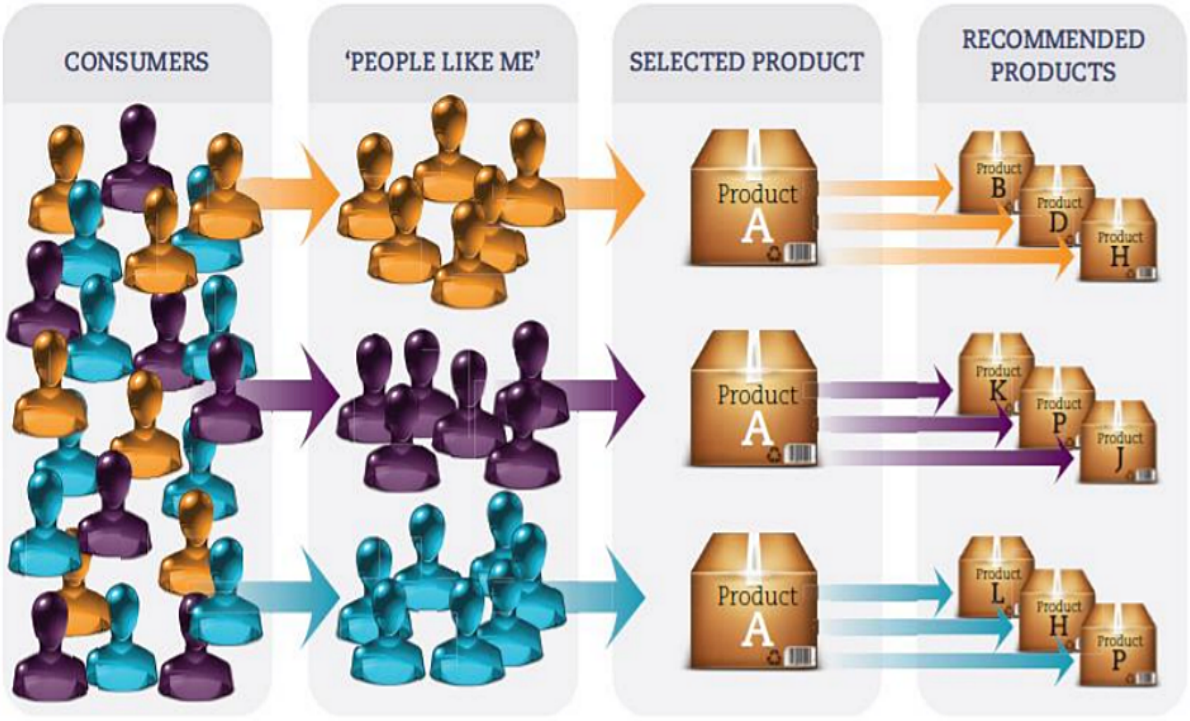
Almost any business can benefit from a recommendation system.
There are two important aspects that determine how much a business benefits from a recommendation system:

1) Breadth of data: A business serving only a handful of customers that behave in different ways will not receive much benefit from an automated recommendation system. Humans are still much better than machines in the area of learning from a few examples. In such cases, your employees will use their logic, qualitative and quantitative understanding of customers to make accurate recommendations.

2) Depth of data: Having a single data point on each customer is also not helpful to recommendation systems. Deep data about customers online activities and if possible offline purchases can guide accurate recommendations

9. CONCLUSION:

Recommender systems are an extremely potent tool utilized to assist the selection process easier for users. The implemented book recommendation engine is a competent system to recommend Books for e-users. This recommender system will definitely be a great web application implemented in Java language. Such type of web application will be proved beneficial for today's high demanding online purchasing web sites. This hybrid recommender system is more accurate and efficient as it combines the features of various recommendation techniques. The book recommendation engine will reduce the overhead associated with making the best choices of books among the plenty.



10. FUTURE SCOPE:

The recommendation system proposed here takes the number of users who have rated the books into account, without factoring in the absolute rating. Due to this, a recommendation might arise from a book that a user has given low ratingto, in which case a book might be recommended from a genre that the user dislikes. This recommendation system relies on the ratings given by users. So, trust is a major issue,like whether the feedback and rating given by the user is genuine or not. This recommendation system does not solve the trust issue. Therefore future research should focus on resolving both these issues.

11. BIBILOGRAPHY:

1. Isinkaye, F., Folajimi, Y. and Ojokoh, B. (2019). Recommendation systems: Principles, methods and evaluation.
2. Agarwal, Ajay, and Minakshi Chauhan. "Similarity Measures used in Recommender Systems: A Study." International Journal of Engineering Technology Science and Research IJETSR, ISSN (2017): 2394-3386.
3. Sarwar, B. M., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. Www, 1, 285-295.
4. Bennett, James, and Stan Lanning. "The netflix prize." Proceedings of KDD cup and workshop. Vol. 2007. 2007.
5. Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM, 35(12), 61-70.

6. Ponnam, L. T., Punyasamudram, S. D., Nallagulla, S. N., & Yellamati, S. (2016, February). Movie recommender system using item based collaborative filtering technique. In 2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS) (pp. 1-5). IEEE.

7. Liu, Haifeng, "A new user similarity model to improve the accuracy of collaborative filtering." Knowledge-Based Systems 56 (2014): 156-166.

8. Elnagar A. and Einea O. 'BRAD 1.0: Book reviews in Arabic dataset'. 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), pp. 1-8, Nov 2016. DOI:10.1109/AICCSA.2016.7945800.

9. Hug, Nicolas. "Surprise, a Python library for recommender systems." URL: http://surpriselib.com (2017).

Al Hassanieh, Lamis, "Similarity measures for collaborative filtering recommender systems." 2018 IEEE Middle East and North Africa Communications Conference (MENACOMM). IEEE, 2018

12.APPENDIX:

SOURCE CODE:

https://www.kaggle.com/arashnic/book-recommendation-dataset/code

CODE FOR THE SOLUTION WE BUILT:

https://github.com/smartinternz02/SI-GuidedProject-5758-1634105444/blob/main/Model%20Building/Book.ipynb

Python app1.py:

```
from flask import Flask, request, render_template
import pandas as pd # used for data manipulation
import pickle
from gevent.pywsgi import WSGIServer
import os
import requests
app = Flask(__name__) # initializing a flask app
model=pickle.load(open("book.pkl",'rb')) #loading the model
#loading the updated dataset
us_canada_user_rating_pivot=pd.read_csv("us_canada_user_rating_pivot1.csv",encoding = "ISO-8859-1", index_col='bookTitle')
@app.route('/')# route to display the home page
def home():
                                    return render_template('home.html')#rendering the home page
```

```
@app.route('/extractor')
def extractor():
    return render_template('extractor.html')

#extractor page
@app.route('/keywords',   methods=['POST'])# route to show the predictions in a web UI
def keywords():
    typ=request.form['type']
    output=request.form['output']
    #if typ=="text":
                #output=re.sub("[^a-zA-Z.,]"," ",output)
    print(output)
    distances,indices=model.kneighbors(
us_canada_user_rating_pivot.loc[output,:].values.reshape(1, -1), n_neighbors = 6)
    keyword=[]
                for    i    in    range(0, len(distances.flatten())):
        if i == 0:
```

```
31                     print('Recommendations for
  {0}:\n'.format(output))
32             else:
33                                 keyword.append('{0}:
  {1}'.format(i,
34
  us_canada_user_rating_pivot.index[indices.fla
  tten()[i]]))
35     # showing the prediction results in a UI
36                                             return
  render_template('keywords.html',keyword=keywo
  rd)
37 port = os.getenv('VCAP_APP_PORT','8080')
38 if __name__ == "__main__":
39     # running the app
40      app.secret_key = os.urandom(12)
41
  app.run(debug=True,host='0.0.0.0',port=port)
```