**Book Recommendation System**

**Project Report By**:

Advait Patil

Jerin Easo Thomas

Thathva Sri Sai Reddy Yenuga

**ABSTRACT**

In today’s digital world, with everything being readily available on the internet, the world lacks a standard of recommending books the way everyone is recommended movies, products or even music. Amazon alone has movies, shopping and music which recommends the users on what they could watch or buy based on their previous history or the genre they are most frequently visiting. Such a system is non-existent for books where if one likes a particular book, the most obvious choice is he/she looks for a book from the same author or another part of the same book. It is essential that there is a way that the users can be recommended books based on the genre, the author or even based on ratings of the book given by other users. This project addresses this issue by recommending books for a user based on the book he/she is currently reading or based on ratings of the book.

**Keywords:** Recommendation System, K Nearest Neighbors, Unsupervised Learning, Preprocessing, Exploratory Data Analysis.

**Dataset**

The goodreadsbooks dataset from kaggle has been used in this project. This dataset has about 11000 data entries with 12 features describing various information regarding each book such as title, author, number of pages, ratings, etc.

**Table

Description automatically generated**

**Fig 1: Details of the data.**

**Column description:**

**bookID:** Helps in identifying unique book records.

**title:** Conveys the name of each book.

**authors:** Conveys the author/authors of each book.

**average\_rating:** Conveys the mean rating by the users all the book.

**isbn & isbn13:** The International Standard Book Number is a numeric commercial book identifier that is intended to be unique. These columns help in identifying some additional information about the books.

**language\_code:** Conveys the language in which the book is published.

**num\_pages:** Helps in identifying the number of pages in the book.

**ratings\_count:** Conveys the total number of ratings given to the book.

**text\_reviews\_count:** Conveys the count of total textual reviews given to the book.

**publication\_date:** Determines the publishing date.

**publisher:** Conveys the name of the publisher.

**METHODS**

* Unsupervised Nearest Neighbour:  
  The idea behind nearest neighbor approaches is to select a set number of training samples that are geographically closest to the new point and then estimate the label based on them.

Implementing nearest neighbors learning without supervision is Unsupervised NearestNeighbors. Three distinct nearest neighbors algorithms—BallTree, KDTree, and a brute-force technique based on sklearn.metrics.pairwise routines—are accessible through a consistent interface. The term "algorithm," which must be one of ["auto," "ball tree," "kd tree," or "brute,"] controls which neighbors search algorithm is used. The algorithm looks to the training data to find the optimum strategy when the default value of "auto" is passed.

* K-D Tree:

Computer scientists can divide spatial points using a tree data structure called a K-D tree. This is particularly useful when trying to find a point's "nearest neighbors" or closest point. At each level of the tree, a K-D tree compares points along a different axis. For instance, in the image above, level 0, or the level the root is on, decides based on the x-value whether a point should go to the left or the right. Then, based on the y-value, it decides at the following level whether a point should move to the left or the right. Up till all the points are inserted, this process is repeated. The same procedure would apply if someone wanted to operate in more than two dimensions, but each necessary dimension would have its own levels rather than just comparing the x and y values at each level.

* Cosine Similarity:  
  Cosine Similarity is a calculation that expresses how similar two or more vectors are. In our project we have primary used Cosine Similarity to compare the results with the results of the Nearest Neighbour.

The cosine of the angle between two vectors is the measure of similarity. Usually not zero, the vectors are located in an inner product space. The divide between the dot product of vectors and the product of the euclidean norms or magnitude of each vector is how the cosine similarity is formally defined. In widely used libraries and tools like Matlab, SciKit-Learn, TensorFlow, etc., the cosine similarity measurement approach is one of the most widely used similarity measurement methods.

* Content based filtering:  
  A sort of recommender system called content-based filtering makes an educated estimate as to what a user could enjoy based on that user's activities.

The similarity between items is used by content-based recommendation systems to suggest items to users. Based on their description or features, this recommender system makes suggestions for goods or products. Using the descriptions of the products, it determines how similar they are. In order to suggest a similar product, it also considers the user's prior usage. For instance, if a user like the Sidney Sheldon book "Tell Me Your Dreams," the recommender system may suggest that the user read further Sheldon books or a book in the "non-fiction" genre. (The novels by Sydney Sheldon go under the non-fiction category.)

**DISCUSSION**

**Data Preprocessing**

Data preparation is the process of putting raw data into a format that is comprehensible. Given that we cannot work with raw data, it is also a crucial stage in data mining. Before using machine learning or data mining methods, the data's quality should be examined.

Steps Involved in Data Preprocessing which we have implemented in our project:

1. Data Cleaning:

There may be a lot of useless information and gaps in the data. Data cleansing is completed to handle this portion. It entails dealing with erroneous data, noisy data, etc.

* 1. Missing data:

We have check if our data has any missing features, if yes, these features were filled with null values.

* 1. Duplicate data:

As a part of pre-processing, we have checked for duplicate data in our data set and removed the duplicate records, which would in turn yield better accuracy model.

* 1. Noisy Data:

In our data there were four outlier data having misplaced feature data conveying false meaning, hence we have removed these records from our dataset.

Table

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

**Fig 2: Code snippet for checking of duplicates and null values.**

The above code checks if there are any duplicates or null values in the dataset.

1. Data Transformation:

As a part of data transformation, we have seen that the feature names ’authors’ were having some additional junk data with the name of the author, hence we have then standardized the same name across all instances.

For example: Author named ‘J.K. Rowling-Mary GrandPré’ were changed to ‘J.K. Rowling’.

1. Data Reduction:

Here we have used the dimensionality reduction to remove the unwanted feature like ‘isbn’ and ‘isbn13’ which aren’t being used in our recommendation system.

Graphical user interface, text, application

Description automatically generated

**Fig 3: Code snippet for data transformation and feature reduction.**

The above code performs the data transformation by changing the author’s name and performs the feature reduction on unwanted columns.

**Exploratory Data Analysis**

The first part of the exploratory data analysis revolves around basic analysis of the given dataset. This involves finding out the number of data entries, finding null values if any, and checking if the data is homogeneous throughout the dataset. We find that there are no null values in the entire dataset, and the data has 11123 rows and 12 columns - meaning it has 11123 unique book IDs and 12 attributes (Book ID, Author, Title, Ratings, etc).

We have first found out if there are any books which are occurring multiple times. We can observe that there are quite a few books which appear multiple times, with a unique Book ID and ISBN. These books have a different publisher and different page count, but the same rating (as the books are essentially the same). This does not give a lot of information regarding the book itself, but can be quite helpful if any analysis regarding the publishing company needs to be carried out, with respect to the books. We can combine these entries if we only want to base our analysis on the book itself, with no other attributes such as the publisher / number of pages involved.

Next, we found out the most commonly used language, and it was English. The books were tagged as English, English UK as well as English US. If we consider all these as the same, then we get the second most common language as Spanish, followed closely by French. This gives an idea as to how many books of each language are being read/are popular. Since they are not very popular, they will eventually be suppressed by the prediction model unless that particular language is specifically given as a filter.

We also found out the ratings count of the ratings. This is done to get a rough estimate as to how many reviews can be considered as a decent number of reviews. We have considered the top 10 rating counts and the highest is above 40000, while the lowest in the top 10 is around 20000. There are quite a few books with rating count of less than 100, but they also happen to be the highest rated books on average as they have only been reviewed by a small number of people. Thus, to find out the highest rated books, we can set a basic filter of minimum 5000 reviews, which will be a fair assumption of the unbiased nature of the average rating.

The graphs for top authors and top publishers show the most popular authors/ publishers. This is a basic observation which tells us that these are the particular authors/ publishers which are most trusted by readers when it comes to choosing a good book.

We have also plotted a graph of the number of pages in all the books. This is almost a normally distributed curve, with most of the books having a number of pages around 400. This graph on its own does not tell us a lot, but we can use the same along with average ratings to find out if there is a relation between the number of pages in a book and how much the book is actually liked by readers. There is often said to be a correlation between the two and while some people do not mind, some people like to have a quick read on their hands.

**Chart, bar chart

Description automatically generated**

**Fig 4: Plot showing the top 10 authors with most books**

Here we plot the authors with the count of books against their name in the dataset. Stephen King and PG. Wodehouse have the number of books.

**Chart, histogram

Description automatically generated**

**Fig 5: Plot showing the distribution of number of pages in a book**

Plotting with the number of pages as a Normal Distribution. The highest point is at around 0.0025 as the density and number of pages is approximately 700 pages.

**Chart, bar chart

Description automatically generated**

**Fig 6: Plot showing the books with the highest rating count**

The plot shows the books with the highest rating counts (that is the book that has the highest number of ratings). 1 unit is equal to 100000 ratings count in the x-axis.

Text

Description automatically generated

**Fig 7: Wordcloud plot showing the most common authors in the dataset**

The word cloud plot shows the authors that are repeated the most in the dataset.

Chart, bar chart

Description automatically generated

**Fig 8: Plot showing the books occurring the most number of times**

The plot shows the books that are occurring the most of times in the dataset. These books are part of a larger series of books by the same name.

**Model Building**

To build the model for the recommendation system, first the data is supposed to be cleaned and in proper format. After some data cleaning like removing duplicates and unnecessary columns, we make sure that the ratings column is converted to a categorical column by adding a new column with a range of ratings mapped to each category. This is done since the ‘average\_rating’ column contains continuous variable data. The encoding is done using Label Encoder. The ‘language\_code’ column is encoded with Label Encoder as well. The new ratings column is concatenated to the dataset, and the entire dataset is normalized between 0 and 1 using MinMax scalar. With the dataset now ready, we use the Nearest Neighbors to get the 10 most similar books based with ‘kd\_tree’ as the base algorithm.

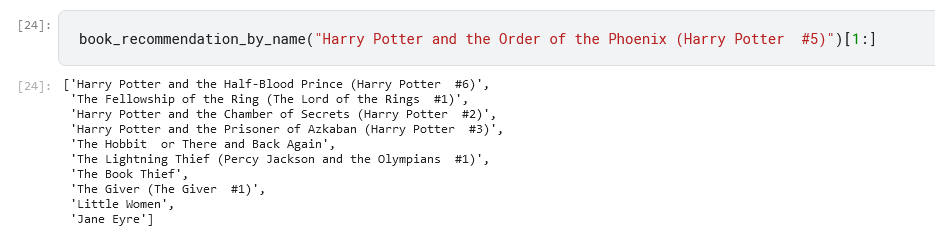
For cosine similarity, the title column is first vectorized using Tfidf vectorizer and the cosine similarity is found for these vectors. This cosine similarity model is used as base for finding similarities whenever cosine similarity is used.

**Recommendation System**

Next, the recommendation system was implemented using the unsupervised version of Nearest Neighbors which gave accurate results. For example, the input being Harry Potter and the Order of Phoenix, the output was similar to the genre of the input or books similar to the input. Cosine similarity also was experimented, which only gave out results that had the words of the input in their titles. So ‘Control of Nature’ produced output that had ‘control’ or ‘of’ or ‘nature’ in its title. So Nearest neighbors gave good results and would be a good algorithm to fine tune for more better results. Using the same feature matrix, books were recommended based on the Author’s name. For example, J.R.R Tolkien is a science fiction author, and the books recommended were of science fiction genre.

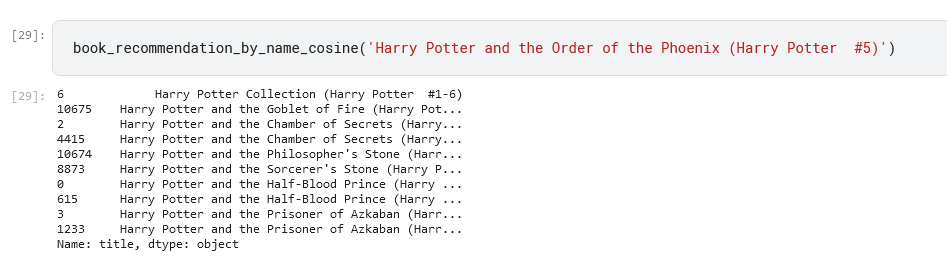
**KEY RESULTS**

**Recommendation System**

****

**Fig 6: Output of recommendation system for ‘Harry Potter and the Order of the Phoenix’ using Nearest Neighbors**

Using the nearest neighbors algorithm, the output produced were of the same genre as the input or were other books of the same series.

****

**Fig 7: Output of recommendation system for ‘Harry Potter and the Order of the Phoenix’ using Cosine Similarity**

Using the cosine similarity metric, the output produced were of the books that contained the keywords of the input.

**CONCLUSION**

**REFERENCES**

[1] https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks

[2] https://scikit-learn.org/stable/modules/neighbors.html#unsupervised-neighbors

[3] https://scikit-learn.org/stable/modules/neighbors.html

[4] https://www.upwork.com/resources/what-is-content-based-filtering

[5] https://www.hindawi.com/journals/wcmc/2021/7036357/

[6] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

[7]https://towardsdatascience.com/tree-algorithms-explained-ball-tree-algorithm-vs-kd-tree-vs-brute-force-9746debcd940