**BOOK RECOMMENDATION SYSTEM**

**USING**

**SENTIMENT ANALYSIS**

**&**

**CLASSIFICATION IN NLP**

**NAME: - RUTHVIK ADIMULAPU**

**STUDENT ID: -23032641**

**COURSE:-DATA SCIENCE**

**SUBJECT:- MACHINE LEARNING & NEURAL NETWORKS**

**GITHUB REPROSITORY LINK:** **<https://github.com/Ruthvik-Adimulapu/ML-Tutorial-Book-Recommendation..git>**

**INTRODUCTION:**

NLP means as Natural Language processing, a fast-advanced domain, deals with a computer system interaction using human languages. NLP or Natural Language Processing is a subset of artificial intelligence (AI) that deals with the communication between computers and human languages. The ultimate aim of NLP is to facilitate the reading, comprehension and extraction of information from the written human language easily by a machine. 3. Natural Language Processing (NLP) NLP is used in many real-world implementations like chatbots, sentiment analysis, machine translation, automated summary of texts, etc.

Basically, NLP tries to mimic the way humans grasp a language and tackle real-world problems using text data. Although we as human beings are naturally good at processing language, NLP involves several challenges because language as a whole is ambiguous in nature, rich in nuances, and very much context dependent. For example, the word bat: it can mean a flying mammal or a sporting good, depending on context

Sentiment Analysis is one of its main applications, which aims to categorize the text into sentiment classes (positive, negative, neutral). This project focuses on generating a book recommendation system based in the sentiment analysis of book descriptions. It uses this information to build sentiment analysis of the book descriptions. It uses this information to build sentiment classifications oof the book descriptions and helps user suggest books.

This project's objective is to:

* Sort book descriptions according to their sentiment: neutral, negative, or positive.
* Create a recommendation system that makes book suggestions based on how emotionally similar they are to a particular novel.

**2. Overview of the Dataset:**

The dataset is taken from Kaggle

Dataset Link: https://www.kaggle.com/datasets/saurabhbagchi/books-dataset/data

The project's dataset, which includes comprehensive book metadata, was obtained from Kaggle. Among the dataset's salient features are:

|  |  |  |
| --- | --- | --- |
| S. No |  |  |
| 01 | Authors | Authors of the book |
| 02 | Rating | Average rating of the book |
| 03 | Description | Short description the book |
| 04 | Rating Distribution | Distribution of ratings |
| 05 | Counts Of Review | Number of reviews |
| 06 | Language | Language in which the book has been written. |

The dataset consists of hundreds of thousands of rows, and a row representing the review. The main purpose of this is to classify the reviews based on the sentiment whether positive or negative.

In this analysis, the description columns are the most important once. Because in this description columns the description in this is used for the sentiment analysis and book recommendation.

**The Operation of the Book Recommendation System:**

This project's recommendation algorithm compares book descriptions to identify commonalities between them. This is a detailed explanation of the recommendation's process:

1. Extraction of Features Making Use of TF-IDF

Converting book descriptions from text to numerical vectors is the initial stage in the recommendation process. Term Frequency-Inverse Document Frequency, or TF-IDF, is a widely used technique for transforming textual input into numerical features appropriate for machine learning models.

* Term frequency, or TF: quantifies how frequently a phrase occurs in a document.
* IDF(Inverse Document Frequency): Common words like "the" and "is" that appear frequently in all texts but may not have as much meaning are downscaled in weight by IDF (Inverse Document Frequency).

Each book's Cleaned Description is subjected to the TF-IDF vectorizer, which transforms it into a high-dimensional vector with each dimension standing for a word throughout the whole corpus of book descriptions.

2.Cosine Similarity to measure the similarity in books:

The next stage is to compare the books to see how comparable they are once the descriptions have been transformed into numerical vectors. For this comparison, cosine similarity is employed.

The cosine of the angle formed by two vectors in a vector space is measured by cosine similarity. The vectors and the books they represent are increasingly similar the smaller the angle.

Cosine similarity can be computed mathematically as follows:

The formula for cosine similarity

A black background with white text

Description automatically generated

where A and B represent the vector representations of two books, A⋅B is the dot product of the two vectors, and ∥A∥∥A∥ and ∥B∥∥B∥ are the vector magnitudes (norms).

The range of cosine similarity is -1 to 1:

A similarity score of one indicates that the descriptions of the two books are the same. When two novels have a similarity of zero, they are completely different. Since they are based on book descriptions, we expect values to range between 0 and 1, but a negative similarity suggests they are opposites.

We may utilise the cosine similarity matrix to suggest books after it has been constructed. The recommendation procedure operates as follows:

1. Cosine Similarity Matrix: We determine each book pair's cosine similarity. This produces a matrix in which the similarity between two novels is represented by each element. This matrix will have a size of N × N × N, where N N is the total number of books in the dataset.
2. Choosing a Book to Recommend: We examine the row in the similarity matrix that corresponds to a book given its index, such as the book at index 10. The similarity scores between the chosen book and every other book are shown in this row.
3. Sorting and Choosing Similar Books: Books with the highest cosine similarity scores (around 1) are those that are most similar to the chosen book. We choose the top N related novels by sorting the books in descending order of similarity scores.

HOW THIS BOOK RECOMMENDATION WORKS:

Cosine Similarity on TF-IDF vectors derived from book descriptions is used by the recommendation system. Here is a step-by-step explanation of how it operates with a real-world scenario.

Step 1: Preparing the Data

The books and descriptions in our dataset look like this:

|  |  |
| --- | --- |
| BOOK TITLE | DESCRIPTION |
| “Book A: The Adventure” | |  | | --- | | A thrilling journey of exploration and mystery. |  |  | | --- | |  | |
| “Book B: Science Wonders” | Discover the wonders of science and technology. |
| "Book C: Travel Memoir" | An adventure across continents and new experiences. |

Step2: Text Vectorization using TF-IDF

We have convert the description into TF-IDF vectors:

|  |  |
| --- | --- |
| BOOK TITLE | DESCRIPTION |
| "Book A: The Adventure" | |  | | --- | | [0.7, 0.1, 0.6, 0.0] | |
| "Book B: Science Wonders" | [0.0, 0.8, 0.1, 0.9] |
| "Book C: Travel Memoir" | [0.6, 0.0, 0.7, 0.0] |

Step3:

We use the cosine similarity Formula:A math equation on a black background

Description automatically generated

Where:

The TF-IDF vectors for two books are A and B.

Step 4:

Example of a Book Recommendation User Question:

Let's say someone reads "Book A: The Adventure" and wants suggestions that are similar. We calculate the cosine similarity between the vectors in Book A and every other book

|  |  |
| --- | --- |
| Compared book | Cosine Similarity Score |
| "Book B: Science Wonders" | 0.95 (Highly similar) |
| "Book C: Travel Memoir" | |  | | --- | | 0.10 |  |  | | --- | |  | |
| |  | | --- | | "Book E: Mystery Chronicles" |  |  | | --- | |  | | 0.85 (Similar) |

Step 5:

Suggested Reading

The system suggests the following based on the scores:

Travel Memoir (Book C) (Score: 0.95).

Mystery Chronicles, Book E (Score: 0.85).

So, In order to provide users with tailored book recommendations based on their tastes, the system uses Cosine Similarity and TF-IDF to identify books with comparable themes and descriptions.

**METHODOLOGY:**

This project's recommendation system is built on content-based filtering, which suggests related books based on text similarities across book descriptions. The primary steps involved are as follows:

1.Preprocessing Data

Cleaning Text: To start, we eliminate extraneous characters (such as punctuation and special symbols) from the book descriptions.

Tokenisation: This is the process of dividing a text into discrete words, or tokens.

Stop word Removal: Words that don't add sentiment, such as "the," "is," and "at," are eliminated.

Lemmatisation: This is the process of transforming words into their root or base form (for example, "running" to "run").

2. TF-IDF-Based Feature Extraction

TF-IDF (Term Frequency-Inverse Document Frequency) is used to transform the book descriptions into a format that machine learning models can use. We can identify the key terms in a text that are less prevalent in all descriptions by using TF-IDF.

3. Analysis of Sentiment:

The book descriptions are subjected to sentiment analysis. The objective is to categorise each book description's sentiment as follows:

Positive: Books with generally positive descriptions.

Negative: Books that describe themselves negatively or unappealingly.

Books that are described as neutral—not overly favourable or negative—are called neutral books.

Books with similar sentiments are then filtered and suggested using this sentiment classification.

4. Similarity to Cosine:

The similarity between two book descriptions is measured using cosine similarity. In a multi-dimensional space, where each dimension corresponds to a phrase in the book description, it calculates the cosine of the angle formed by two vectors. Two descriptions are comparable if their cosine similarity values are high.

5. Book Recommendation:

In order to suggest books based on the cosine similarity of their descriptions, we develop the recommend\_books() function. Based on how similar the descriptions of the books are, this method takes a book's index and returns the top N related books.

Code Explanation:

As said earlier we have taken dataset from Kaggle. This dataset consists of books data. Here’s the detailed explanation walkthrough of the code created for the project:

Step1: Loading the dataset

To load the data into the pandas Data Frame we use pd.read\_csv() function. This data set consists of various columns, but we have focussed mainly on the description and Rating columns.

A screenshot of a computer

Description automatically generated

Step 2: Data preprocessing:

The book descriptions must be cleaned and pre-processed before we can prepare the text data. Tokenisation, stop word removal, and cleaning are handled by the preprocess\_text() method. A screenshot of a computer

Description automatically generated

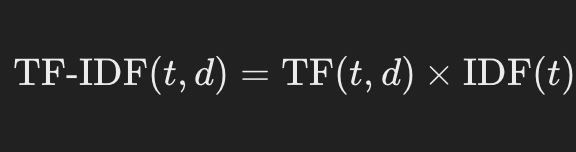
Step 3: TF-IDF Vectorization:

The book descriptions are transformed into numerical vectors using TF-IDF so that machine learning algorithms may process them. The vectorisation is done by scikit-learns TfidfVectorizer.

A screenshot of a computer program

Description automatically generated

The formula we used to convert the book descriptions into numerical vectors is the Term frequency-inverse document frequency formulae.



Where:

* TF(t, d) is the Term Frequency of term in document d:

A black and white sign with white text

Description automatically generated

* IDF(t) is the Inverse Document Frequency of term t:

A black and white text

Description automatically generated

Where:

* N is the Total number of documents in the corpus
* df(t) is the number of documents containing term

Step 4: Calculating Cosine Similarity

Using the cosine\_similarity function, we determine the cosine similarity between the book descriptions.A screenshot of a computer code

Description automatically generated

**Step 5: Book Recommendation Function**

We define a function to recommend books based on their similarity to a given book's description. This function returns the top N similar books.

A screenshot of a computer code

Description automatically generated

Step 6: Use Case Example:

Lastly, we provide a book index and ask for the top five most similar books in order to evaluate the recommendation system.A screenshot of a computer

Description automatically generated

**RESULTS & EVALUATION:**

1.MODEL PERFORMANCE:

The cosine similarity of book descriptions is calculated by the recommendation system. This approach is based on content-based filtering, which suggests books to consumers based on comparable descriptions. The precision of the sentiment analysis and similarity computation have a significant impact on the quality of recommendations.

2.EVALUATION OF SENTIMENT ANALYSIS:

Even though the algorithm performs admirably for the majority of descriptions, there may be instances in which the sentiment classification is unclear, particularly for novels with intricate or neutral descriptions. To increase accuracy, more sophisticated sentiment analysis models like BERT or other transformer-based models could be used.

3.EVALUATION OF BOOK RECOMMENDATIONS:

Based on descriptions, the recommendation system it does a good job at matching books with related subjects. However, by using user evaluations or additional information like ratings and genres for more individualised recommendations, its efficacy could be further improved.

**Conclusion :**

This project effectively illustrates how to create a book recommendation system using sentiment analysis and content-based filtering. We were able to recommend books based on their textual descriptions by utilising NLP techniques like TF-IDF and Cosine Similarity. Users' reading experiences are improved by the recommendation system's insightful book recommendations.

Although the method for making book recommendations is efficient, there is still room for development. More individualised suggestions might result, for instance, by adding user preferences or integrating collaborative filtering. Using more sophisticated deep learning models for sentiment analysis could also increase the system's accuracy. By improving user engagement through tailored recommendations, the idea has practical applications in digital libraries, online bookshops, and e-commerce platforms. Through efficient targeting, such solutions can increase revenues and user retention.

System accuracy may be raised by improvements such as hybrid models that combine sentiment-driven categorisation, collaborative filtering, and neural embeddings. Additionally, a more thorough suggestion model might be offered by user reviews, ratings, and contextual information like reading history.

**REFERENCES:**

Research Papers and blogs:

1. <https://ar5iv.org/html/2306.12680>
2. <https://towardsdatascience.com>
3. <https://medium.com>

Tutorials and blogs

1. <https://ieeexplore.ieee.org/document/10428182>
2. <https://ieeexplore.ieee.org/document/10099750>

**Books References**

1. **"Recommender Systems: An Introduction"**

**Authors**: Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich

1. **"Recommender Systems Handbook"**

**Editors**: Francesco Ricci, Lior Rokach, Bracha Shapira

1. **"Pattern Recognition and Machine Learning"**

**Author**: Christopher M. Bishop

1. **"Speech and Language Processing"**

**Authors**: Daniel Jurafsky and James H. Martin

1. **"Mining of Massive Datasets"**

**Authors**: Jure Leskovec, Anand Rajaraman, Jeffrey D. Ullman

1. **"Natural Language Processing with Python"**

**Authors**: Steven Bird, Ewan Klein, and Edward Loper