Content-Based Movie Recommender App with Streamlit

Problem Statement:

The goal of this project is to develop a **Movie Recommendation System** that provides personalized movie suggestions to users based on the content of movie overviews. With an overwhelming amount of movies available, it is challenging for users to find movies aligned with their tastes. This system uses a **content-based filtering approach** to analyze the plot summaries (overviews) of movies, and recommend similar movies based on textual similarities. The solution should be easy to use, efficient, and capable of delivering real-time recommendations via a user-friendly **Streamlit web interface**.

Approach: The code you've written implements a **Content-Based Movie Recommendation System** using the TF-IDF vectorizer and cosine similarity. Here's a breakdown of the key steps:

1. Loading and Merging Data

- You first load two datasets: `tmdb_5000_movies.csv` and `tmdb_5000_credits.csv`.
- Then, the data from both files is merged based on the common `id` column.

2.Preprocessing the Overview Text

- You handle missing values in the `overview` column by filling NaN values with an empty string.

3. TF-IDF Vectorization

- You use the `TfidfVectorizer` to convert the text data (movie overviews) into numerical vectors while ignoring common English stop words.

4. Cosine Similarity Calculation

- You compute the cosine similarity between the TF-IDF vectors to identify similar movies.

5. Building a Reverse Index

- You create a mapping from movie titles to indices in the dataset, which will allow you to look up movies easily.

6. Recommendation Function

- The `get_recommendations` function takes in a movie title, retrieves its cosine similarity scores, and returns the top 10 most similar movies.

7. Getting Recommendations

- You can now get movie recommendations by calling the `get_recommendations` function with any movie title.

8.Streamlit App:

- A basic **Streamlit** interface is set up to take a movie title as input from the user and display the top 10 recommendations.

Conclusion

In this case study, I developed a **Content-Based Movie Recommendation System** through the following steps:

1. **Data Merging**: Combined movie and credits datasets for comprehensive information.
2. **TF-IDF Vectorization**: Converted movie overviews into a numerical format for analysis.
3. **Cosine Similarity Calculation**: Measured similarity between movies based on their overviews.
4. **Recommendation Function**: Implemented a function to retrieve the top 10 similar movies based on user input.
5. **Streamlit Deployment**: Created an interactive interface for users to input movie titles and receive recommendations.

This project enhanced my skills in data preprocessing, modeling, and web app development, paving the way for future enhancements.