EVALUATION REPORT

MOVIE RECOMMENDATION SYSTEM

Prepared by :
VISHAL NAIK

INTRODUCTION

This report evaluates the performance of a movie recommendation system implemented using four different filtering methods: collaborative filtering, content-based filtering, popularity-based filtering, and hybrid filtering. The goal is to determine the effectiveness of each method and compare their performances.

Dataset and Experimental Setup

The evaluation was conducted using the Open source dataset, which contains information related to movies. We employed data wrangling techniques to clean and prepare the movie dataset for recommendation system training. We utilized data modeling techniques to develop and optimize recommendation algorithms, ensuring robust performance. Complete code can be found here

Model Evaluation

To assess the performance of each recommendation method, we used the following evaluation metrics:

- **Root Mean Squared Error (RMSE):** A measure of the differences between the predicted and actual user ratings.
- **Mean Absolute Error (MAE):** A measure of the average magnitude of the errors in the predictions.

Results

The results and inferences are as follows:

Collaborative Filtering

The collaborative filtering method, implemented using the Surprise library and the SVD algorithm, yielded an RMSE of 0.89 and an MAE of 0.69. These values indicate that, on average, the predicted ratings differ from the true ratings by approximately 0.89 units, and the absolute difference between predicted and true ratings is 0.69 units. The assessment of these results should be considered in the context of the specific characteristics and requirements of the recommendation system and dataset.

Model Prediction

```
In [18]: movies[movies["id"] == "8844"][['title', 'id']]
```

Out[18]:

	title	id
1	Jumanji	8844

Let's check for movie "Jumanji"

```
In [19]: # Making a prediction for user with ID 15 and movie with ID 8844
         svd.predict(15,8844)
```

```
Out[19]: Prediction(uid=15, iid=8844, r_ui=None, est=2.35865495104768, details={'was_impossible': False})
```

Inference:

- **uid:** The user ID (uid) for whom the prediction is made. In this case, the user ID is 15.
- **iid:** The item ID/movie ID (iid) for which the prediction is made. Here, the item ID is 8844(Jumanji).
- **r_ui:** The actual or known rating given by the user for the item. In this case, it's set to None, indicating that the actual rating is not provided (perhaps because this is a prediction on an unseen item).
- **est:** The estimated rating for the user-item pair. In this case, the estimated rating is approximately **2.36**.
- **details:** Additional details about the prediction. In this case, 'was_impossible': False indicates that the prediction was possible and not flagged as impossible by the recommender system algorithm.

Popularity–Based Filtering

For the popularity–based filtering method, we utilized a weighted rating formula that considers both the average rating and the number of votes. This approach prioritizes movies that not only received high ratings but also garnered a substantial number of votes.

$$WR = (v / (v + m)) * R + (m + (v + m)) * C$$

v - number of votes for a movie
m - minimum number of votes required
R - average rating of votes required
C - average rating across all movies

	title	weighted_rating
15480	Inception	80882.101819
12481	The Dark Knight	70735.803784
14551	Avatar	69863.894656
17818	The Avengers	69223.615522
26564	Deadpool	66099.887645
22879	Interstellar	64656.696209
20051	Django Unchained	59656.186657
23753	Guardians of the Galaxy	58066.329123
2843	Fight Club	56179.000750
18244	The Hunger Games	55930.421897

Inference:

- **The Popularity-Based Filtering method** was employed to recommend movies based on their overall popularity. The **weighted rating formula**, considering both the average rating and the number of votes, was utilized to prioritize movies that not only received high ratings but also garnered a substantial number of votes.
- The **top 10 movies** recommended by this approach, as displayed above, showcase the most popular and well-received movies according to the weighted rating criterion.
  Users looking for widely acclaimed and widely watched movies can refer to this list for their next cinematic experience.

Content-Based Filtering

For the content-based filtering method, we used cosine similarity scores based on TF-IDF vectors to recommend movies with similar content features to a given movie.

Find the most similar movies to a certain movie

```
def similar_movies(movie_title, nr_movies):
    """
    Get a list of similar movies based on the cosine similarity between movie vectors.

    Parameters:
    - movie_title (str): The title of the movie for which similar movies are to be found.
    - nr_movies (int): The number of similar movies to retrieve.

    Returns:
    - List[str]: A list of titles of similar movies.
    """
    # Find the index of the movie in the DataFrame
    idx = movies_sampled.loc[movies_sampled['title']==movie_title].index[0]

    # Get the similarity scores for the given movie
    scores = list(enumerate(similarity_matrix[idx]))

    # Sort the movies based on similarity scores in descending order
    scores = sorted(scores, key=lambda x: x[1], reverse=True)

    # Extract the indices of similar movies (excluding the input movie)
    movies_indices = [tpl[0] for tpl in scores[1:nr_movies+1]]

    # Get the titles of similar movies
    similar_titles = list(movies_sampled['title'].iloc[movies_indices])

    return similar_titles
```

```
similar_movies('The Fifth Musketeer',4)
```

```
['Return of the Fly',
 'David',
 'The Scorpion King: Rise of a Warrior',
 'Trucks']
```

Inference:

- **Content-Based Filtering**, implemented through the `similar_movies` function, employs **cosine similarity** scores based on **TF-IDF vectors** to recommend movies with similar content features to a given movie.
- In this demonstration, movies similar to 'The Fifth Musketeer' have been identified, showcasing the capability of the Content-Based Filtering approach to provide personalized recommendations based on content characteristics.
- This method is particularly valuable for suggesting movies that share thematic elements, genres, or content features, delivering a personalized viewing experience aligned with the user's preferences.

Hybrid Filtering

The hybrid filtering method effectively combines content-based and collaborative filtering approaches to deliver personalized movie suggestions. This approach considers content similarities, such as genres, keywords, directors, and cast, alongside user-item interactions.

By leveraging the strengths of both content-based and collaborative filtering methods, the hybrid approach addresses the limitations of individual techniques and offers a more robust solution for movie recommendations, ultimately enhancing recommendation accuracy and diversity.

```
Content-Based Recommendations:
                      title
2420  My Sucky Teen Romance
4070           My Name Is Ki
4527            Ask the Dust
3266      Over Her Dead Body
4150       The Nanny Diaries
2244               City Girl

Collaborative Filtering Recommendations:
                            title
188  How to Win the US Presidency
557           Travelling with Pets
411       Jönssonligan spelar högt
273       Life Is Hot in Cracktown
219                    Kid Galahad
563                      City Zero
```

Inference:

The hybrid movie recommendation system effectively combines content-based and collaborative filtering approaches to deliver personalized movie suggestions. By leveraging content similarities like genres, keywords, directors, and cast, alongside user-item interactions, the system enhances recommendation accuracy and diversity.

This hybrid method addresses the limitations of individual techniques, providing a more robust solution for movie recommendations.

CONCLUSION

In conclusion, the evaluation demonstrated that the hybrid filtering method was the most effective recommendation method for our movie dataset, offering personalized and diverse movie recommendations by combining the strengths of content-based and collaborative filtering approaches. Future improvements may involve exploring additional features or incorporating other recommendation techniques to further enhance the system's accuracy and user satisfaction.

THANK YOU!