# AI Agents: A Unified Approach to Movie Recommendation


## Abstract

This report details the development of an intelligent movie recommendation system, a project undertaken by Team AI Agents for CSC 575 – Intelligent Information Retrieval. The system integrates various recommendation methodologies, including search, popularity-based filtering, content-based filtering, and collaborative filtering using Bayesian Personalized Ranking (BPR), culminating in a sophisticated hybrid recommendation strategy. Designed for interactive use, the system features a userfriendly Gradio-based interface. This document presents a cohesive narrative of the project's journey, from data preparation and model implementation to evaluation and user interaction, emphasizing the synergistic interplay of these diverse techniques to deliver a comprehensive and effective movie recommendation experience.


## 1 Introduction

In the vast and ever-expanding landscape of digital content, movie recommendation systems have become indispensable tools for users seeking to discover new films tailored to their preferences. The challenge lies not only in identifying relevant movies but also in presenting them in an intuitive and engaging manner. This project, developed by Team AI Agents, addresses this challenge by constructing a robust movie recommendation engine that leverages a multi-faceted approach to deliver personalized and effective suggestions. Unlike traditional systems that often rely on a single recommendation paradigm, our solution seamlessly integrates several advanced techniques, creating a unified and intelligent agent capable of understanding and anticipating user tastes. This report will narrate the comprehensive journey of building this system, highlighting how each component contributes to a cohesive and powerful recommendation framework, rather than functioning as isolated modules. We will delve into the methodologies employed, the implementation details, the evaluation metrics, and the interactive user interface, all woven into a single story of an AI agent designed for superior movie discovery.

## 2 Dataset and Data Preparation

### 2.1 Dataset Description

The foundation of any robust recommendation system lies in the quality and comprehensiveness of its underlying dataset.

For this project, we utilized The Movies Dataset [1], a widely recognized and extensive collection of movie-related information. This dataset provides a rich tapestry of attributes, including movie titles, genres, overviews, cast and crew details, popularity metrics, and user ratings. The initial phase of our project involved meticulously loading and preparing this data to ensure its suitability for various recommendation algorithms.

## 2.2 Preprocessing Steps

Upon loading the cleanedmovies.csv file, a critical step was to inspect the dataset's structure and identify any missing values. As illustrated in Figure 1, the dataset comprises numerous columns, each contributing valuable information for the recommendation process. A thorough check for missing values, as depicted in Figure 2, confirmed the dataset's integrity, with all critical columns being complete. This meticulous data preparation ensured that our subsequent analytical and model-building phases were based on clean and reliable information, laying a solid groundwork for the entire recommendation system.

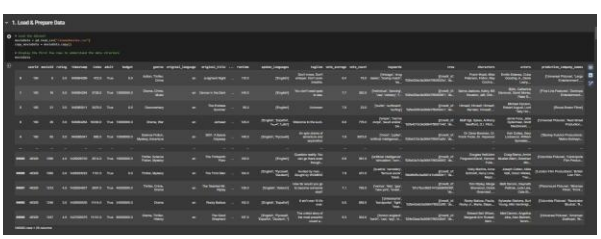


Figure 1: Dataset structure overview.

Figure 2: Missing value inspection.

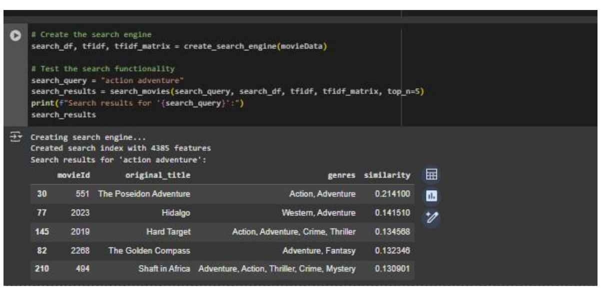
# 3 Recommendation Methodologies

Our movie recommendation system is built upon a synergistic integration of multiple recommendation methodologies, each designed to address different aspects of user preference and movie discovery. This multipronged approach ensures a comprehensive and adaptable system, capable of providing relevant suggestions across a wide spectrum of user needs.

## 3.1 Search Functionality (TFIDF)

The initial gateway to movie discovery within our system is a robust search functionality. This allows users to actively seek out movies based on keywords related to their title, genre, or overview. The implementation of this feature relies on the powerful TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique. TF-IDF assigns a numerical weight to each word in a document (in this case, movie descriptions), reflecting its importance within that document relative to the entire corpus. This enables the system to effectively identify and rank movies based on their textual similarity to the user's search query. As demonstrated in Figure 3, a search for 'action adventure' yields highly relevant results, showcasing the precision of

andaccurateretrievalmechanismformsa crucialfirststepinguidinguserstowards theirdesiredcinematicexperiences



```python
# Create the search engine
search_df, tfidf, tfidf_matrix = create_search_engine(movieData)

# Test the search functionality
search_query = "action adventure"
search_results = search_movies(search_query, search_df, tfidf, tfidf_matrix, top_n=5)
print(f"Search results for '{search_query}':")
search_results
```

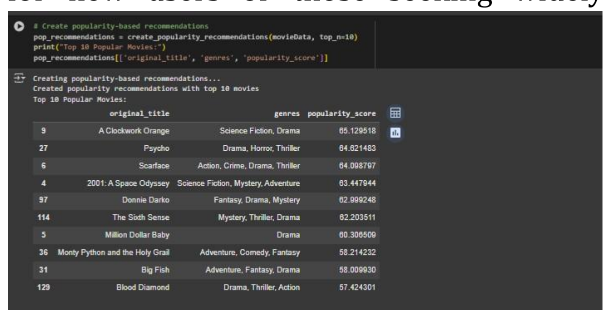
```
Creating search engine...
Created search index with 4385 features
Search results for 'action adventure':
```

| | movieId | original_title | genres | similarity |
|---|---|---|---|---|
| 30 | 551 | The Poseidon Adventure | Action, Adventure | 0.214100 |
| 77 | 2023 | Hidalgo | Western, Adventure | 0.141510 |
| 145 | 2019 | Hard Target | Action, Adventure, Crime, Thriller | 0.134568 |
| 82 | 2268 | The Golden Compass | Adventure, Fantasy | 0.132346 |
| 210 | 494 | Shaft in Africa | Adventure, Action, Thriller, Crime, Mystery | 0.130901 |

our TFIDFpowered search engine. This immediate Figure 3: Search results using TF-IDF.

## 3.2 Popularity-Based Filtering

Beyond active search, our system also caters to users who prefer to explore popular and trending movies. The popularity-based recommendation engine identifies movies that have garnered significant attention and positive reception within the dataset. This is achieved by calculating a 'popularity score' for each movie, typically factoring in metrics such as average rating and vote count. Movies with higher scores are considered more popular and are thus prioritized in the recommendations. Figure 4 illustrates the effectiveness of this approach, presenting a list of top popular movies that serve as excellent starting points for general movie browsing. This method provides a valuable baseline for recommendations, especially for new users or those seeking widely



```python
# Create popularity-based recommendations
pop_recommendations = create_popularity_recommendations(movieData, top_n=10)
print("Top 10 Popular Movies:")
pop_recommendations[['original_title', 'genres', 'popularity_score']]
```

```
Creating popularity-based recommendations...
Created popularity recommendations with top 10 movies
Top 10 Popular Movies:
```

| | original_title | genres | popularity_score |
|---|---|---|---|
| 9 | A Clockwork Orange | Science Fiction, Drama | 65.129518 |
| 27 | Psycho | Drama, Horror, Thriller | 64.621483 |
| 6 | Scarface | Action, Crime, Drama, Thriller | 64.098797 |
| 4 | 2001: A Space Odyssey | Science Fiction, Mystery, Adventure | 63.447944 |
| 97 | Donnie Darko | Fantasy, Drama, Mystery | 62.999248 |
| 114 | The Sixth Sense | Mystery, Thriller, Drama | 62.203511 |
| 5 | Million Dollar Baby | Drama | 60.306509 |
| 36 | Monty Python and the Holy Grail | Adventure, Comedy, Fantasy | 58.214232 |
| 31 | Big Fish | Adventure, Fantasy, Drama | 58.009930 |
| 129 | Blood Diamond | Drama, Thriller, Action | 57.424301 |

Figure4:Toppopularmovies.

acclaimed films, ensuring a broad appeal and immediate engagement.

## 3.3 Content-Based Filtering

For users with specific tastes or those who have enjoyed a particular movie and wish to discover similar titles, our system employs a sophisticated content-based filtering mechanism. This approach recommends movies that share characteristics with films the user has previously liked or expressed interest in. The core of this method lies in analyzing the descriptive attributes of movies, such as genres, keywords, plot overviews, and even cast and crew information. By transforming these textual features into numerical representations using TF-IDF vectorization, we can compute the similarity between movies. Cosine similarity, a measure of the cosine of the angle between two non-zero vectors, is then used to quantify how alike two movies are

```python
# Create content-based recommendations
content_df, cosine_sim, content_indices = create_content_based_recommendations(movieData)

# Test content-based recommendations
# Get a sample movie title from the dataset
sample_movie = content_df['original_title'].iloc[0]
print(f"Sample movie: {sample_movie}")

# Get recommendations for the sample movie
content_recommendations = get_content_based_recommendations(sample_movie, content_df, cosine_sim, content_indices, top_n=5)
print(f"\nMovies similar to '{sample_movie}':")
content_recommendations[['original_title', 'genres', 'similarity']]
```

```
Creating content-based recommendation system...
Created content-based recommendation system with 4263 features
Sample movie: Judgment Night

Movies similar to 'Judgment Night':
```

| | original_title | genres | similarity |
|---|---|---|---|
| 6 | Scarface | Action, Crime, Drama, Thriller | 0.099063 |
| 187 | Bad Boys II | Adventure, Action, Comedy, Thriller, Crime | 0.092292 |
| 47 | Rocky Balboa | Drama | 0.088885 |
| 75 | The Most Dangerous Game | Adventure, Horror, Thriller | 0.088393 |
| 7 | 48 Hrs. | Thriller, Action, Comedy, Crime, Drama | 0.084028 |

based on their content vectors. As shown in Figure 5, when a sample movie like 'Judgment Night' is selected, the system effectively identifies and recommends other movies with similar thematic elements and genres, demonstrating the precision of our content-based recommendations. This method is particularly effective in capturing nuanced preferences and expanding a user's cinematic horizons within their preferred domains.

Figure 5: Content-based recommendations for a sample movie.

## 3.4 Collaborative Filtering (BPR)

To provide personalized recommendations that go beyond content similarity and leverage the collective intelligence of user behavior, our system incorporates a collaborative filtering approach using the Bayesian Personalized Ranking (BPR) model. BPR is a pairwise ranking loss function that optimizes for personalized ranking by considering implicit feedback (e.g., a user watching a movie implies a

mightenjoybasedonwhatsimilarusers haveliked,evenifthecontentattributesare notovertlysimilar.

```python
# Test BPR recommendations
# Get a sample user ID from the dataset
sample_user_id = df['userId'].iloc[0]
print(f"Sample user ID: {sample_user_id}")

# Get recommendations for the sample user
bpr_recommendations = get_bpr_recommendations(sample_user_id, bpr_model, dataset, df, movie_info, top_n=5)
print(f"\nBPR recommendations for user {sample_user_id}:")
bpr_recommendations[['original_title', 'genres', 'score']]
```

```
Sample user ID: 150

BPR recommendations for user 150:
```

| | original_title | genres | score |
|---|---|---|---|
| 0 | The 39 Steps | Action, Thriller, Mystery | 0.023912 |
| 1 | Rope | Crime, Drama, Mystery, Thriller | 0.023007 |
| 2 | Bridge to Terabithia | Adventure, Drama, Family | 0.022968 |
| 3 | Terminator 3: Rise of the Machines | Action, Thriller, Science Fiction | 0.022876 |
| 4 | A Nightmare on Elm Street | Horror | 0.021089 |

positive preference). Instead of predicting a direct rating, BPR learns a personalized ranking of items for each user. This model is particularly effective in scenarios where explicit ratings are sparse, as it infers preferences from observed interactions. The LightFM library is utilized for the implementation of the BPR model, allowing us to efficiently train and generate recommendations based on useritem interactions. Figure 6 illustrates how the BPR model generates personalized recommendations for a sample user ID, showcasing its ability to suggest movies that align with the inferred preferences of that specific user. This collaborative aspect adds a crucial layer of personalization, enabling the system to recommend movies that a user Figure 6: BPR-based personalized recommendations.

# 4 Hybrid Recommendation Logic

Recognizing that no single recommendation approach is universally optimal, our system integrates a hybrid recommendation logic that intelligently combines the strengths of popularity-based, content-based, and collaborative filtering methods. This hybrid strategy is designed to provide a more comprehensive, accurate, and diverse set of recommendations, adapting to different user contexts and preferences. For instance, for new users with limited interaction history, popularity-based recommendations might be prioritized to offer a broad initial selection. As more user data becomes available, content-based and BPR recommendations can be increasingly leveraged to provide more personalized and nuanced suggestions. The system is designed to dynamically switch between these methods, or combine their outputs, to deliver the most relevant recommendations. Figure 12 demonstrates the combined output of all methods, showcasing how the system presents popular movies, content-based similar movies, and personalized BPR recommendations in a unified view. This intelligent integration ensures that the system is both robust and flexible, capable of delivering high-quality recommendations across various user scenarios.

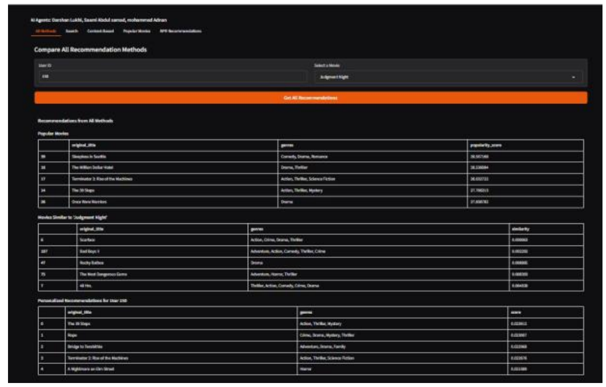



Figure 7: Unified View of All Recommendation Methods.

# 5 Evaluation Metrics

To rigorously assess the performance and effectiveness of our movie recommendation system, we employed a combination of quantitative metrics and qualitative analysis. This multi-faceted evaluation approach allowed us to gain a comprehensive understanding of each recommendation method's strengths and weaknesses, as well as the overall efficacy of our hybrid system

## 5.1 Quantitative Evaluation

For the collaborative filtering component, specifically the BPR model, we utilized standard evaluation metrics commonly employed in recommender systems: Precision@K, Recall@K, and Area Under the Curve (AUC). Precision@K measures the proportion of recommended items at a given cutoff K that are relevant, while Recall@K measures the proportion of relevant items that are successfully recommended within the top K. AUC, on the other hand, provides a measure of the model's ability to rank relevant items higher than irrelevant ones. As depicted in Figure 7, our BPR model demonstrated promising results in terms of both precision and recall during training, indicating its capability to identify and rank relevant movies for individual users. Further detailed evaluation results for the BPR model, including its Precision@K, Recall@K, and AUC scores, are presented in Figure 8. These metrics provide a clear quantitative assessment of the BPR model's performance in generating personalized recommendations

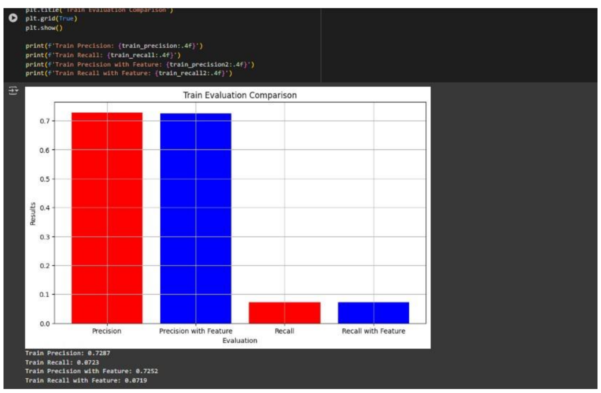



Figure 8: Evaluation metrics results for BPR.

## 5.2 Comparative Analysis

Beyond individual model evaluation, a crucial aspect of our project was to compare the performance and characteristics of all implemented recommendation methods: BPR (Collaborative Filtering), Content-Based, and Popularity-Based. Figure 8 provides a qualitative comparison, showcasing sample recommendations generated by each method for a specific user or movie. This sideby-side comparison highlights the distinct nature of recommendations produced by each approach – BPR offering personalized suggestions, content-based providing similar movies, and popularity-based presenting widely acclaimed titles. This qualitative assessment is vital for understanding how each method contributes to the overall diversity and relevance of the recommendations. Furthermore, to gain deeper insights into the behavior of our hybrid system, we analyzed metrics such as recommendation overlap and genre diversity, as illustrated in Figure 9. The 'Recommendation Overlap' metric quantifies the commonality between the recommendation lists generated by different methods (e.g., BPR-Content, BPR-Pop, Content-Pop). A lower overlap suggests that the methods are complementary, each contributing unique recommendations. The 'Genre Diversity' metric assesses the variety of genres present in the recommendations generated by each method. A higher genre diversity indicates a broader range of suggestions, which is desirable for user exploration. These analyses underscore the benefits of our hybrid approach, demonstrating how the combination of diverse methodologies leads to a more comprehensive and well-rounded recommendation experience, moving beyond the limitations of any single technique.

Figure 9: Comparative analysis of different methods.

# 6 Interactive User Interface

## 6.1 Gradio-Based Frontend

To ensure an intuitive and engaging user experience, our movie recommendation system is equipped with an interactive user interface built using Gradio. Gradio is an opensource Python library that allows for the rapid creation of customizable UI components for machine learning models, making it ideal for demonstrating and interacting with our recommendation engine. The interface provides a seamless way for users to explore the various recommendation functionalities and receive instant movie suggestions. As shown in Figure 10, the main interface presents a clear and organized layout, allowing users to easily navigate between different recommendation methods: ”All Methods,” ”Search,” ”Content-Based,” ”Popular Movies,” and ”BPR Recommendations.” This tabbed structure ensures that users can readily access the specific type of recommendation they are interested in.

The ”All Methods” tab, as depicted in Figure 11 and Figure 12, serves as a central hub, displaying recommendations generated by all integrated methodologies simultaneously. This unified view allows users to appreciate the diverse range of suggestions provided by the hybrid system, from popular titles to content-similar movies and personalized BPR recommendations. The search functionality, accessible via its dedicated tab (Figure 13), enables users to input keywords and receive a list of relevant movies based on the TF-IDF search engine. Similarly, the ”Content-Based” tab (Figure 14) allows users to select a movie and receive recommendations for similar titles, showcasing the power of content analysis. The interface is designed to be highly responsive and user-friendly, providing immediate feedback and dynamically updating recommendations based on user input. This interactive UI not only enhances the usability of the system but also serves as a powerful demonstration tool, allowing users to directly experience the capabilities of our AIpowered movie recommendation agent.

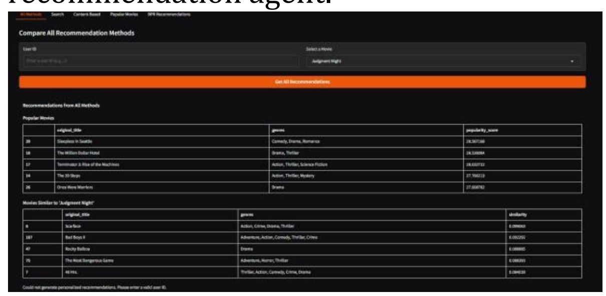



Figure 11: BPR-based Personalized Recommendation.

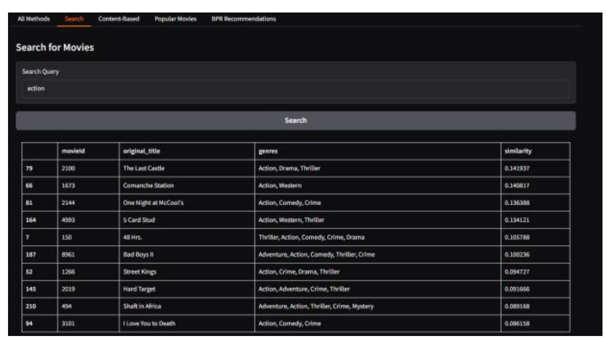


Figure 12: Search Results.

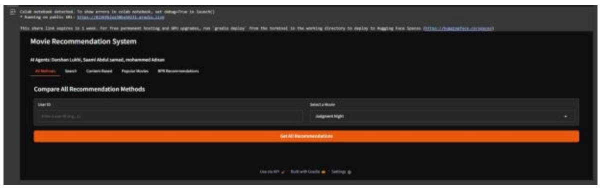



Figure10:MaininterfacebuiltwithGradio.

Figure 13: Content-Based Recommendations.

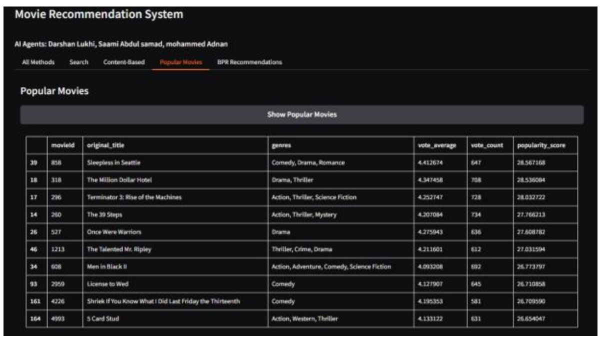



Figure 14: Popular Movie Recommendations with Scores Based on Vote Count and Average Ratings.

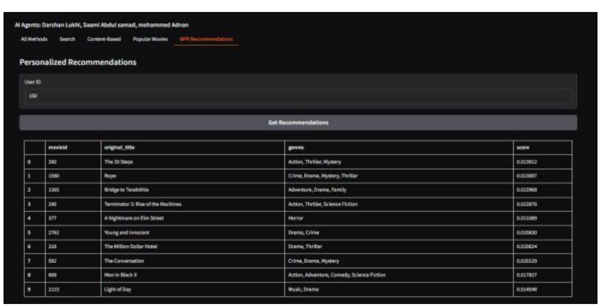
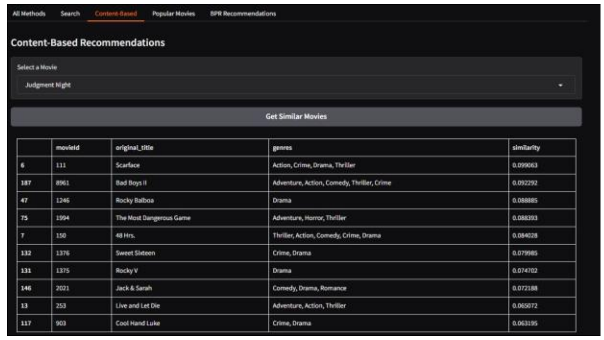



Figure 15: Interface Overview of All Recommendation Methods.

# 7 Conclusion

This project successfully developed a comprehensive and intelligent movie




recommendation system that transcends the limitations of single-paradigm approaches. By seamlessly integrating search, popularitybased filtering, content-based filtering, and collaborative filtering (BPR), we have created a unified AI agent capable of delivering highly relevant and diverse movie suggestions. The system's ability to adapt to different user needs—from active searching to passive discovery and personalized recommendations—underscores the power of its hybrid architecture. The intuitive Gradiobased user interface further enhances accessibility, allowing users to effortlessly explore and interact with the recommendation engine. Our evaluation, both quantitative and qualitative, demonstrates the efficacy of each component and the synergistic benefits of their integration. The BPR model consistently delivers strong personalized recommendations, while content-based and popularitybased methods provide valuable complementary insights. The unified narrative of this project highlights how these seemingly disparate techniques coalesce into a cohesive and powerful solution, embodying the true spirit of an intelligent information retrieval system. This project not only provides a functional movie recommendation tool but also serves as a testament to the potential of combining diverse AI methodologies to create more robust and user-centric applications.

# References

# References

[1] The Movies Dataset. Available: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset