
ML: Recommendation System Report

Hamzah Mukadam, Shreya Maher
College of Engineering
Northeastern University
Boston, MA
mukadam.ha@northeastern.edu
maher.sh@northeastern.edu

Abstract

In our project, we delved into the domain of movie recommendation systems using an extensive dataset. Our objective centered around the creation of recommendation algorithms employing three core filtering methods: content-based filtering, collaborative filtering, and hybrid filtering. Through analysis, we compared and assessed the performances of these methodologies in suggesting movies to users. Additionally, we developed a user-friendly front-end application using Streamlit, showcasing the practical execution and functionality of these recommendation algorithms. This collective exploration provided insights into the strengths and intricacies of each filtering technique, not only by examining performance metrics but also by demonstrating real-world applicability through an intuitive interface. Our findings contribute to a deeper understanding of diverse strategies employed in movie recommendation systems.

1 Datasets

We used Movie and Ratings Dataset. We obtained this dataset from Kaggle

Table 1: The basic feature of both datasets.

	Data Set Characteristics	Attribute Characteristics	Associated Tasks	Number of Instances	Number of Attributes
Dataset 1	Multivariate	Real	Recommendation System	10,48,576	28

1.1 Data characteristics

The dataset comprises a collection of movies and rating, encompassing a substantial volume of 10,48,576 entries. Our dataset encompasses comprehensive information across various categories such as adult status, belonging to collections, budget allocation, genres, homepage availability, unique identifiers (IDs), IMDb ID, original language, titles, summaries, popularity metrics, poster paths, production companies, countries of production, release dates, revenue generated, runtime duration, spoken languages, status, taglines, titles, video availability, average voting scores, vote counts, and ratings. Due to resource constraints and to facilitate manageable analysis, a sample subset of the data was utilized to gain insights, ensuring a more efficient and feasible examination of the dataset.

1.2 Why are these interesting datasets?

The movie dataset is fascinating for recommendation system studies as it covers a wide range of aspects such as genres, popularity, and user ratings. Being passionate movie enthusiasts, my teammate and I are intrigued by platforms like Netflix and their recommendation systems. We were eager to explore how these systems utilize extensive movie data to offer tailored suggestions, motivating us to delve into this dataset for a deeper understanding.

2 Data Preparation

2.1 Merging Datasets

We combined two separate datasets by aligning common attributes, allowing us to consolidate diverse information into a single, comprehensive dataset. This merging process enabled us to enrich our data analysis by integrating complementary data sources and uncovering deeper insights through a more holistic view.

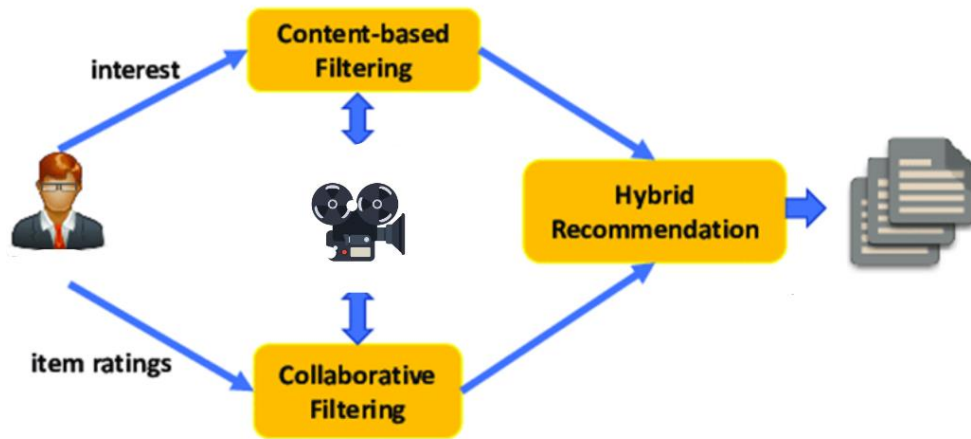
2.2 Converting Categorical Data: Enhancing Readability and Analysis

In our data preparation phase, we transformed categorical data originally represented in the format of dictionaries – containing 'id' and 'name' pairs such as {'id': 28, 'name': 'Action'}, {'id': 80, 'name': 'Crime'}, {'id': 18, 'name': 'Drama'}, {'id': 53, 'name': 'Thriller'} – into a more user-friendly and analytically convenient form. This involved extracting and restructuring the data to contain distinct genres directly, such as 'Action', 'Crime', 'Drama', and 'Thriller'. By making this conversion, we aimed to simplify the dataset structure, improving its readability and facilitating easier analysis. This step allowed for a more straightforward representation of movie genres or similar categorical information, optimizing the subsequent data processing and modeling stages in our analytical workflow.



2.3 Data Transformation into Matrix Form

We transformed the raw data into a matrix structure to address the sparsity issue inherent in recommendation system datasets. By organizing user-item interactions into a matrix format, we aimed to streamline computational efficiency while enabling the utilization of matrix factorization techniques. This transformation facilitates the extraction of latent patterns and features within the data, essential for generating accurate and personalized recommendations, thus enhancing the effectiveness of our recommendation system modeling.



84
85 **3.1 Content Based Filtering**

86 **What we performed:**

- 87 1. TF-IDF Vectorization:
- 88 Transformed movie overviews into numerical representations using TF-IDF, assigning
- 89 importance scores to words based on their frequency in movie descriptions.
- 90 Generated TF-IDF vectors to capture unique content characteristics for each movie.
- 91 2. Cosine Similarity Calculation:
- 92 Computed cosine similarity scores between movies using TF-IDF vectors.
- 93 Identified higher scores as indicative of greater content similarity between movies.

94
95 **What we learned about content based filtering:**

- 96 1. Content-Based Recommendation Basics:
- 97 Explored content-based recommendation principles relying on item characteristics to
- 98 suggest similar items, rather than user-item interactions.
- 99 2. TF-IDF and Cosine Similarity Significance:
- 100 Understood TF-IDF's role in capturing unique content features and cosine similarity's
- 101 importance in quantifying content likeness for personalized recommendations based on
- 102 content similarities.

103
104 **Output:**

105 The output from `content_based_recommendations("The Dark Knight")` exemplifies content-

106 based recommendation by suggesting movies closely related in content to "The Dark Knight."

107 These recommendations are derived from the textual content (movie overviews) processed

108 through TF-IDF vectorization and cosine similarity, allowing the system to identify movies

109 with similar thematic elements, storylines, or genres, without relying on user-item

110 interactions. The function selects recommendations based on the textual similarity of movie

111 descriptions, ensuring that the suggested movies share comparable content characteristics,

112 aligning with the user's interest in "The Dark Knight."

```
# Example usage
content_based_recommendations("The Dark Knight")

[10]
... 18252 The Dark Knight Rises
      1328 Batman Returns
      15511 Batman: Under the Red Hood
      21194 Batman Unmasked: The Psychology of the Dark Kn...
      150 Batman Forever
      20232 Batman: The Dark Knight Returns, Part 2
      40974 LEGO DC Comics Super Heroes: Batman: Be-Leaguered
      41982 Batman Beyond Darwyn Cooke's Batman 75th Anniv...
      19792 Batman: The Dark Knight Returns, Part 1
      18035 Batman: Year One
```

3.2 Collaborative Filtering

What We Performed:

1. Data Loading and Preparation:
Utilized the Surprise library to load the ratings dataset, creating a format suitable for collaborative filtering.
Split the dataset into training and testing subsets using the `train_test_split` function to evaluate the model's performance.
2. Collaborative Filtering Model Training:
Employed the SVD (Singular Value Decomposition) algorithm, a matrix factorization technique, to build the collaborative filtering model.
Trained the model on the training set containing user-item interactions to learn latent factors representing users' preferences and item characteristics.
3. Making Predictions:
Generated predictions for ratings on the test set using the trained collaborative filtering model (`algo.test(testset)`).
4. Top-N Recommendations Generation:
Defined a function (`get_top_n`) to extract and compile the top-N movie recommendations for each user based on the predicted ratings.
Provided an example of obtaining the top 10 recommendations for User ID 1 using the trained model and printing those recommendations.

What we learned about collaborative filtering:

1. Collaborative Filtering Fundamentals:
Explored collaborative filtering, a technique that recommends items by leveraging similarities between users or items based on historical interactions.
Understood SVD as a matrix factorization method, capable of discovering latent factors representing user preferences and item characteristics to make personalized recommendations.

147 2. Model Evaluation and Recommendations:
148 Learned to split data into training and test sets for model evaluation.
149 Gained insights into generating top-N recommendations for users by sorting and
150 extracting the highest predicted ratings, tailoring recommendations based on individual
151 user preferences inferred from collaborative filtering.

152

153

154 **3.3 Hybrid Filtering**

155 **What We Performed:**

156 1. Hybrid Recommendation Function:

157 Defined a `hybrid_recommendations` function that takes in a movie title, user ID, content-
158 based model (`content_model`), and collaborative filtering model (`collaborative_model`).

159 Utilized the content-based model to obtain content-based recommendations (`content_rec`)
160 for the given movie title.

161 Extracted collaborative filtering recommendations (`collaborative_rec`) for the specified
162 user ID from the collaborative model.

163 Merged both sets of recommendations to form a hybrid set of recommendations
164 (`hybrid_rec`) by combining content-based and collaborative filtering suggestions.

165

166 Example Usage:

167 Provided an example demonstrating the usage of the `hybrid_recommendations` function
168 by passing a user ID (1) and a movie title ("The Dark Knight").

169 Printed the hybrid recommendations generated for the user by combining both content-
170 based and collaborative filtering approaches.

171

172 **Learning:**

173 1. Hybrid Filtering Concept:

174 Explored the concept of hybrid recommendation systems that leverage multiple
175 techniques (content-based and collaborative filtering in this case) to improve
176 recommendation quality and coverage.

177 Understood the method of integrating recommendations from different models to create a
178 unified set of suggestions aimed at providing diverse and potentially more accurate
179 recommendations for users.

180

181 2. Combining Recommendations from Different Models:

182 Learned to merge recommendations obtained from distinct recommendation systems
183 (content-based and collaborative filtering) to form a hybrid recommendation set.

184 Recognized the flexibility of hybrid systems in catering to different user preferences and
185 improving recommendation diversity by combining multiple recommendation
186 approaches.

187

188 **Output:**

189 The output from the hybrid filtering process presents a combined set of movie
190 recommendations derived from both content-based and collaborative filtering approaches. It
191 includes a diverse list of movies, predominantly from the Batman franchise, encompassing

various formats and themes related to the superhero series. The recommendations comprise sequels like "The Dark Knight Rises" and "Batman: The Dark Knight Returns, Part 1 & 2," documentaries such as "Batman Unmasked: The Psychology of the Dark Knight," and adaptations like "Batman: Year One" and "Batman: Under the Red Hood." Notably, the hybrid recommendations contain a mix of specific movie titles along with numerical identifiers (e.g., 68358, 2762), which might refer to additional movies or identifiers within the dataset used for recommendation generation. Overall, this diverse set of recommendations suggests a broad array of Batman-related content catering to different formats and storylines within the franchise, aiming to offer a comprehensive selection for the user's potential interests based on combined content-based and collaborative filtering strategies.

```
title = "The Dark Knight"
hybrid_rec = hybrid_recommendations(title, user_id, content_based_recommendations, top_n)
print(hybrid_rec)
```

['The Dark Knight Rises', 'Batman: The Dark Knight Returns, Part 1', "Batman Beyond Darwyn Cooke's Batman 75th Anniversary Special"]

4 Building the Application

4.1 Collaborative Filtering App

After exploring various filtering methods to recommend movies, including content-based, collaborative, and hybrid approaches, we decided to implement a user interface centered around collaborative filtering. This choice reflects our focus on personalized recommendations derived from user-item interactions. By opting for a collaborative filtering-centric approach in our user interface, we aim to prioritize tailored movie suggestions based on individual user behaviors. This decision aligns with our commitment to enhancing the user experience by emphasizing recommendations derived from collaborative filtering techniques, ensuring that users receive movie suggestions that closely match their preferences.

What we performed:

In our project, we employed Streamlit to create an interactive platform for collaborative filtering-based movie recommendations. By leveraging Streamlit's user-friendly interface, users can input their user ID, triggering the Surprise library's SVD algorithm to generate personalized movie suggestions based on their historical interactions. This integration combines Streamlit's simplicity with collaborative filtering's power, providing users with tailored movie recommendations, enhancing their movie exploration experience through a seamless and personalized interface.

What we learned:

- ### 1. Streamlit for Interactive Applications:

Explored the usage of Streamlit, a user-friendly and efficient library, to build an interactive web application for collaborative filtering-based movie recommendations.

- ## 2. Collaborative Filtering Implementation:

Utilized the Surprise library in Python to implement collaborative filtering using the SVD algorithm, offering personalized movie recommendations based on user preferences inferred from historical interactions.

- ### 3. Customized Display Functionality:

Learned to create a custom function to display movie recommendations with enriched details within the Streamlit interface, enhancing user experience through organized and

242 informative movie suggestions.
243
244 4. Enhanced User Engagement:
245 Through the combination of Streamlit's interactive interface and collaborative filtering
246 techniques, we gained insights into optimizing user engagement.
247

248 **Output:**

Collaborative Filtering Search

Enter your user ID:

11

Show Collaborative Filtering Recommendations

Collaborative Filtering Process:

Step 1: Preparing to show recommendations...

Step 2: Collaborative Filtering Recommendations...

Top Collaborative Filtering Recommendations:



Rating: 3.67

Title: 300

Rating: 3.67

Summary (Tagline): Spartans, prepare for glory!

Overview: Based on Frank Miller's graphic novel, "300" is very loosely based the 480 B.C. Battle of Thermopylae, where the King of Sparta led his army against the advancing Persians; the battle is said to have inspired all of Greece to band together against the Persians, and helped usher in the world's first democracy.

Runtime: 117 minutes

Genre: Action, Adventure, War



Rating: Random

Title: 2010

Rating: Random

Summary (Tagline): "Something wonderful is going to happen"

Overview: This is a sequel to 2001 A Space Odyssey. It is now 2010 and both the Americans and the Russians are racing to get to Jupiter to investigate the black monolith (similar to the one found in Lunar Crater Clavius) which was found by the U.S.S. Discovery in orbit around Jupiter's moons. The U.S.S. Discovery's orbit is rapidly decaying and it will crash into IO but the Americans cannot get there in time to save U.S.S. Discovery. The Russians can get to Jupiter in time but only the Americans have the knowledge to access and awaken the U.S.S. Discovery's H.A.L.9000 sentient computer. This forces a joint American-Soviet space expedition against a backdrop of growing global tensions. The combined expedition is seeking answers to several mysteries. What is the significance of the black monolith? Why did H.A.L.9000 act so bizarrely and terminate 4 of 5 of the U.S.S. Discovery's crew? What happened to David Bowman? Along the way, curious data is detected ...

Runtime: 116 minutes

249
250

4.2 Content Based Filtering App:

Our Streamlit app leverages content-based filtering to provide personalized movie recommendations based on user-entered genres. The app is designed to cater to users seeking movie suggestions within specific genres of interest. Upon entering a genre of their choice, users can click the 'Get Recommendations' button to receive a curated list of the top 10 movie suggestions aligned with the provided genre.

Behind the scenes, the app uses a dataset containing movie information, including genres and plot overviews. Utilizing Natural Language Processing techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) vectorization and cosine similarity, the app processes user-entered genres to identify movies within the specified category. It then employs similarity scores to find related movies within the same genre, ultimately presenting a tailored list of recommended movies to the user.

With its intuitive interface and efficient content-based recommendation system, our Streamlit app aims to enhance users' movie discovery experiences, offering personalized movie suggestions aligned with their genre preferences

Content-Based Filtering Search


Enter your user ID:

Select genres:

T x + -

Show Content-Based Filtering Recommendations

Content-Based Filtering Process:




Rating: 7.2

Title: 2081

Overview: 2081 depicts a dystopian future in which, thanks to the 212th Amendment to the Constitution and the unceasing vigilance of the United States Handicapper General, everyone is "finally equal...." The strong wear weights, the beautiful wear masks and the intelligent wear earpieces that fire off loud noises to keep them from taking unfair advantage.

Rating: 7.2



Rating: 5.6

Title: 2012

Overview: Dr. Adrian Helmsley, part of a worldwide geophysical team investigating the effect on the earth of radiation from unprecedented solar storms, learns that the earth's core is heating up. He warns U.S. President Thomas Wilson that the crust of the earth is becoming unstable and that without proper preparations for saving a fraction of the world's population, the entire race is doomed. Meanwhile, writer Jackson Curtis stumbles on the same information. While the world's leaders race to build "arks" to escape

269

5 Conclusions

Throughout the exploration of various recommendation system methodologies—content-based, collaborative, and hybrid filtering—alongside the development of an application using Streamlit for collaborative filtering, several valuable insights were gained into the realm of recommendation systems.

Content-based filtering allowed us to understand the significance of leveraging textual data and feature extraction techniques like TF-IDF to recommend items based on inherent item characteristics. Collaborative filtering, implemented through Surprise's SVD algorithm, unveiled the power of using historical interactions to personalize recommendations, emphasizing the importance of user-item interactions in enhancing recommendation accuracy. Additionally, the hybrid approach highlighted the benefits of combining multiple methods to leverage the strengths of each, thereby broadening the scope and accuracy of recommendations.

The creation of the Streamlit application for collaborative filtering shed light on the integration of user-friendly interfaces and powerful backend algorithms. This integration highlighted the pivotal role of seamless design in delivering engaging and personalized user experiences, where users could actively participate in generating tailored movie recommendations based on their preferences.

Collectively, this journey has underscored the essence of understanding diverse recommendation techniques—from content-based analysis to collaborative filtering and their hybridization—in providing more accurate, diverse, and personalized recommendations. It reinforced the importance of user engagement, data-driven personalization, and the seamless fusion of interface design with robust backend algorithms in shaping a user-centric recommendation system. Ultimately, this endeavor enriched our comprehension of recommendation systems, emphasizing their role in delivering tailored content and enhancing user satisfaction by aligning recommendations more closely with individual preferences and tastes.

6 Future Enhancements

1. Integration of Contextual Information:
Incorporate contextual information such as time, location, or user behavior patterns into the recommendation system. This inclusion could enhance the system's ability to adapt recommendations to varying user contexts and preferences.
2. Enhanced User Interaction in Streamlit App:
Improve user interaction within the Streamlit application by incorporating feedback mechanisms. Implement features like user ratings, preference selection, or feedback loops to refine recommendations continually and enhance user engagement.
3. Scalability and Real-time Updates:
Focus on scalability and real-time updates by exploring distributed computing techniques or implementing streaming algorithms. This will enable the system to handle larger datasets efficiently and provide real-time recommendations as new data arrives.
4. Dynamic Hybrid Approaches:
Develop dynamic hybrid models that dynamically adjust the weightage or combination of content-based and collaborative filtering based on user behavior or contextual information. This could involve reinforcement learning techniques to optimize the hybrid model's performance based on changing user preferences.

7 Work Contribution

Hamzah	Data Preparation & Feature Engineering, Collaborative and Hybrid Filtering, App For Collaborative Filtering
Shreya	Data Preparation, Content Based Filtering, App For Content Based Filtering

8 References

1. <https://labelyourdata.com/articles/movie-recommendation-with-machine-learning>
2. <https://gritglobal.io/blog/product-recommendation-system/>
3. https://youtu.be/n3RKsY2H-NE?si=LLGaG9IukGj6_yhM
4. <https://towardsdatascience.com/creating-a-hybrid-content-collaborative-movie-recommender-using-deep-learning-cc8b431618af>
5. https://medium.com/@tonydaing_78432/comparing-collaborative-filtering-based-recommender-and-hybrid-recommender-system-cb785efdb15a
6. <https://medium.com/codex/hybrid-recommender-system-netflix-prize-dataset-e9f6b4a875aa>