

Movie Recommendation System



Group 23

Venkat Ratnam Sabbavarapu	1222291070
Shahil Mohammed	1225369223
Raj Kishan Cherukuru	1225507153
Hari Krishna Chinta	1225302910
Bharat Sagar Gurugubelli	1225421197
Vidya Sagar Killamsetty	1226214641

Problem Definition



With a plethora of choices in movies, online shopping, and social networks, users face decision fatigue.

Need?

A system that curates content based on individual tastes, simplifying choices, and enhancing user experience.

Solution?



Introduce a movie recommendation engine that observes viewing habits to suggest personalized options

How It Works??

Utilize the K-Nearest-Neighbor algorithm with cosine similarity to analyze and match user preferences.

Approach?

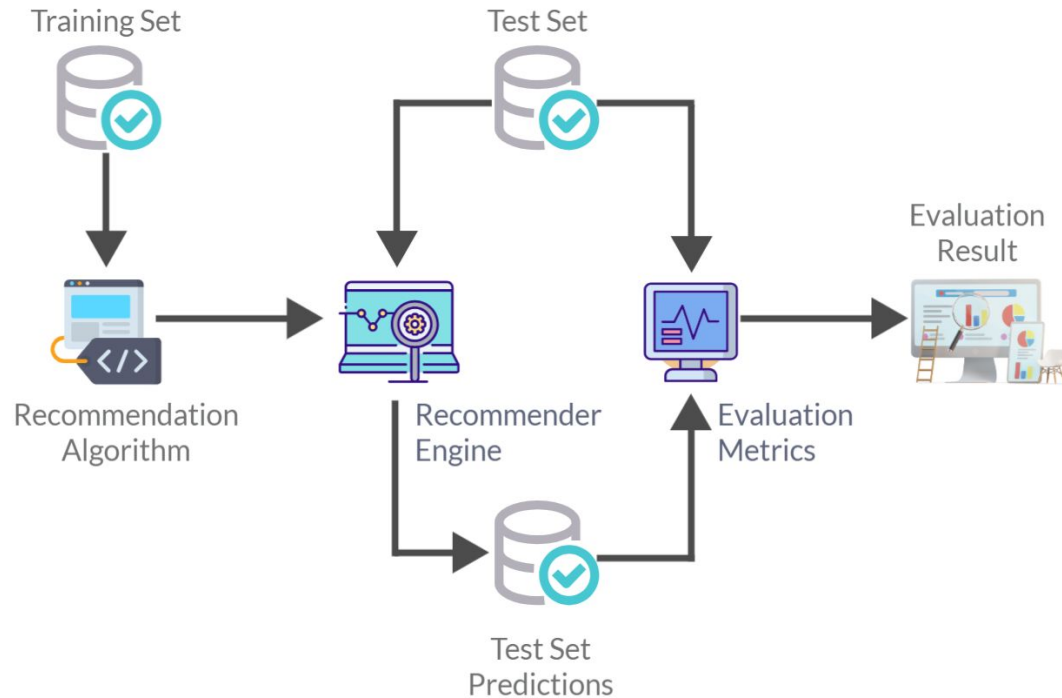


Adopt collaborative filtering to generate suggestions reflecting the likes of similar users.

Foundation?

Use Matrix Factorization for a robust and scalable recommendation system infrastructure.

System Architecture



Dataset Description

MovieLens 20M

- 25 million ratings and one million tag applications.
- Tag genome dataset with 15 million relevance scores.
- Ratings are in the range [0.5, 5]
- At least 20 movies rated by user who is represented by a userID

The Movies Database TMDb 5000

- Contains movies and credits CSV files.
- Both CSV files have 4803 rows each.
- Movies.csv holds movie metadata like genre, language, tagline, and keywords.
- Credits.csv includes movie ID, title, cast, and crew details

Research Plan

Movie Recommendation System Elements

- Users and Items
- Users receive movie recommendations, movies are items

Datasets

- MovieLens 25M
- TMDB 5000

Content-Based Recommendation:

- Nearest Neighbors Algorithm
- Utilizes movie features
- Requires domain knowledge

Research Plan

Collaborative Filtering:

- Learns from other user preferences
- Domain knowledge-free

Algorithms Used:

- Matrix Factorization-Based Recommendation
- KNN (K-Nearest Neighbors) Based Recommendation

State-of-Art Methods & Algorithms



Content-based filtering

Recommends items similar to what a user
has liked in the past



Collaborative filtering

Recommends items based on similarity
between users and items

Content-based and collaborative filtering are two key
techniques for building recommendation systems.

Evaluation Metrics



We are partitioning our dataset into training and testing sets with an **80:20** ratio for evaluation purposes.

Our foundational models include:

Matrix Factorization: This collaborative filtering technique uncovers the relationships among items and specified entities. It reveals hidden attributes and connections between users and item matrices. By leveraging this information, we can ascertain similarities and generate predictions concerning both items and entities.

Item-based KNN (k-nearest neighbors): This model hinges on the similarity of item features. It distributes data points across multiple clusters to deduce predictions for new instances.

To assess the performance of our models, we'll use the following metrics:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)
- Precision and Recall
- Accuracy
- F1 Score

Evaluation Metrics



Evaluating Prediction Accuracy

Mean Absolute Error (MAE): Average absolute difference between predicted and actual ratings. A lower MAE represents better accuracy.

Root Mean Squared Error (RMSE): Square root of the average squared differences between predictions and actual outcomes. It gives a higher weight to larger errors.

Accuracy & F1 Score: Accuracy reflects the overall correct predictions. The F1 Score is the harmonic mean of precision and recall, ensuring a balanced evaluation of the recommendation system's performance.

Precision: The proportion of movies that the system recommended and were actually worth watching.

Recall: The proportion of movies that were worth watching that the system successfully recommended.

Evaluation Metrics



Precision and Recall in Movie Recommendations

Context: Rating Threshold = 3.5


Relevance Definition:

- Relevant Item: True rating ≥ 3.5
- Irrelevant Item: True rating < 3.5

Recommendation Algorithm Output:

- Recommended Item: Predicted rating ≥ 3.5
- Not Recommended Item: Predicted rating < 3.5

Preliminary Findings



	Split	Precision	Recall	RMSE	F1 Score	MAE
0	1	0.667596	0.274070	0.950828	0.388605	0.729382
1	2	0.676355	0.267268	0.946588	0.383136	0.722245
2	3	0.657718	0.254623	0.939984	0.367122	0.716903
3	4	0.660301	0.270438	0.956019	0.383717	0.733120
4	5	0.677732	0.276924	0.943801	0.393189	0.725030

	Split	Precision	Recall	RMSE	F1 Score	MAE
0	1	0.671940	0.276930	0.947999	0.392215	0.725111
1	2	0.672350	0.261450	0.947533	0.376496	0.725553
2	3	0.675369	0.272897	0.946729	0.388722	0.722828
3	4	0.672195	0.273666	0.948169	0.388973	0.728530
4	5	0.665654	0.261422	0.943976	0.375409	0.725491

KNNBase Collaborative Filtering:

Utilizes the KNNBase model from the Surprise library to predict ratings for movies based on a scale of 0.5 to 5, employing k-fold dataset splits to mitigate overfitting.

- **Anti-Testset Generation:** Constructs a complement of the training dataset using `build_anti_testset`, containing entries for movies a user hasn't rated, allowing the model to predict ratings for these unrated movies.
- **Rating Prediction:** Predicts movie ratings within the anti-testset using the KNNBase model and either cosine or Pearson similarity.
- **Top-N Movie Recommendations:** Ranks movies for each user based on predicted ratings, presenting the top N recommended movies accordingly.

Preliminary Findings



	Split	Precision	Recall	RMSE	F1 Score	MAE
0	1	0.681500	0.261927	0.975192	0.378415	0.753310
1	2	0.674945	0.258220	0.978945	0.373534	0.754511
2	3	0.654098	0.251986	0.982151	0.363815	0.758051
3	4	0.648960	0.251235	0.974332	0.362236	0.750020
4	5	0.642857	0.245741	0.964915	0.355563	0.744201

	Split	Precision	Recall	RMSE	F1 Score	MAE
0	1	0.673852	0.266666	0.950745	0.382116	0.729525
1	2	0.652244	0.263808	0.954977	0.375671	0.729906
2	3	0.693634	0.266490	0.940647	0.385048	0.719891
3	4	0.660902	0.263820	0.951512	0.377106	0.727735
4	5	0.669235	0.273755	0.942311	0.388564	0.722014

Matrix Factorization Collaborative Filtering:

Utilizes SVD model for predicting ratings, ranging from 0.5 to 5, employing k-fold data splits to avoid overfitting.

- **Anti-Testset Creation:** Constructs a complement of the training dataset using `build_anti_testset`, providing entries for unrated movies by users.
- **Rating Prediction:** Utilizes the model to predict ratings for the unrated movies in the anti-testset, using Mean Squared Difference, cosine, or Pearson similarity metrics.
- **Top-N Movie Recommendations:** Ranks movies for each user based on predicted ratings, offering the top N movies as recommendations.

UI Landing Page



Movie Recommendation System

Get movie recommendations

1

Get Recommendations

Search by movie name

Inception

Get Recommendations

Results: Based on USER ID

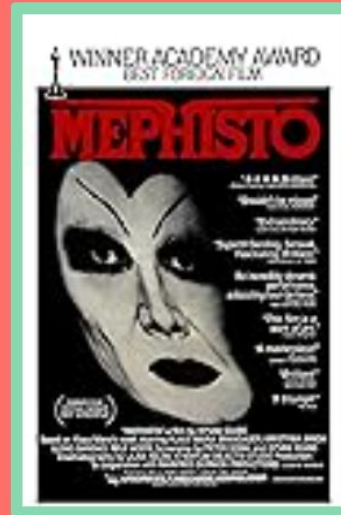
Recommended Movies

Results using Matrix Factorization



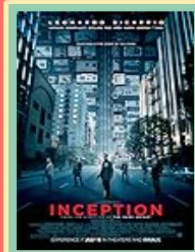
Results (Continued...):

Results using KNN



Results (Continued...): Based on Movie Name

Recommendations based on the movie Inception



Inception



The Prestige



The Dark Knight Rises



Interstellar



Man Of Steel



The Dark Knight



Memento



Batman Begins



Transcendence



The Helix... Loaded

Project Plan



Task	Description	Deadline
Initial Research - Papers, Blogs	Initiate preliminary investigation into the defined problem statement and familiarize yourself with the latest state-of-the-art methodologies applied.	Done
Understand and Prepare the Dataset	Comprehend the dataset and ready it for modeling by engaging in data preprocessing activities.	Done
Algorithms Survey - Evaluate and Finalize	Explore different state-of-the-art algorithms through a survey and assess their performance on small subsets of the 25 million movie dataset.	Done
Modeling - Implementation	Implement models on the entire dataset and check model performance	Done
Project Presentation	Develop a presentation to articulate the problem statement and present proposed solutions.	Done
Demo of the Project	Present the working model and the created presentations	Done
Final Project Report	Capture and summarize the project details in a comprehensive project report.	12/1

References



1. Lu, Jie, et al. "Recommender system application developments: a survey." Decision Support Systems 74 (2015): 12-32.
2. Yehuda Koren, Robert M. Bell, Chris Volinsky: Matrix Factorization Techniques for Recommender Systems. IEEE Computer 42(8): 30-37 (2009)
3. Tahsin Mayeesha. (2018, January 29). Recommending Animes Using Nearest Neighbors. Retrieved from: <https://medium.com/learningmachine-learning/recommending-animes-using-nearestneighbors61320a1a5934>.
4. <https://grouplens.org/datasets/movielens/>
5. <https://medium.com/learning-machine-learning/recommending-animesusing-nearest-neighbors-61320a1a5934>
6. S. Zhang, L. Yao, and X. Xu, "AutoSVD++ An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders," in Proc. of ACM SIGIR, 2017.
7. Takacs, Gabor, et al. "Matrix factorization and neighbor-based algorithms for the Netflix prize problem." Proceedings of the 2008 ACM conference on Recommender systems.ACM, 2008.
8. Yuan Yao, Hanghang Tong, Guo Yan, Feng Xu, Xiang Zhang, Boleslaw K. Szymanski, Jian Lu: Dual-Regularized One-Class Collaborative Filtering. CIKM 2014: 759-768.

Github Repository Link



https://github.com/ChintaHari/CSE-573_MovieRecommendationSystem

THANK
YOU!
