Dates: February 6–8, 2026

The Project Report

Submitted for

Brain Dead Challenge

Submitted by: Dhanashri Abhijit Shah

Email: dhanashrishah1306@gmail.com

Contact no: 8956303473

This report documents my participation in the Brain Dead Challenge, a competitive problem-solving event conducted as part of Revelation 2K26, organised by IIEST Shibpur. The competition was centred around machine learning and analytical thinking, where participants were required to solve and submit solutions to two problem statements within a fixed time frame. The challenge tested not only technical proficiency in machine learning concepts but also problem-solving skills, efficiency, and the ability to work under time constraints. This report presents an overview of the problems addressed, the methodologies applied, and the solutions developed during the competition.

INDEX PAGE

CHALLENGE OVERVIEW

1.1 Problem Statement 1: ReelSense

ReelSense is a movie recommendation system challenge designed to move beyond traditional user rating prediction by focusing on personalised and meaningful recommendations. The objective of the project is to develop a hybrid recommendation approach that generates explainable Top-K movie suggestions tailored to individual users. In addition to accuracy, the system emphasises diversity and coverage to mitigate popularity bias and promote novel content discovery. A key component of the challenge is the generation of natural language explanations for each recommended movie, enhancing transparency and user trust. The system's performance is evaluated using metrics related to ranking quality, diversity, and novelty to ensure a balanced and effective recommendation experience.

1.2 Dataset used:

Dataset: MovieLens Latest Small
Source: GroupLens Research
License: MovieLens Terms of Use

This data was given through the competition organiser.

1.3 Files Used:

1. ratings.csv: User ratings of movies (0.5 to 5.0)

   Contains columns like userid, movieid, rating, timestamp

2. movies.csv: Movie metadata (title, genres)

   Contains columns like movieId, title, genres

3. tags.csv: User-assigned free-form tags

   Contains columns like userid, movieid, tags, timestamp,

4. links.csv: External IDs (IMDB, TMDb)

   Contains columns like movieId, imdbid, tmdbid

DATA PREPROCESSING

1. Why is data preprocessing needed?

Data preprocessing is a crucial step in any data-driven system, as it ensures that the dataset is accurate, consistent, and suitable for analysis and model development. Before applying recommendation algorithms, the raw datasets were carefully examined to understand their structure, quality, and completeness. Basic exploratory operations were performed to validate the integrity of the data and identify any potential issues that could affect model performance.

As part of the preprocessing phase, the row counts of each table were computed, missing values were analysed, and unique counts of users, movies, and ratings were examined. These steps helped in verifying dataset consistency and assessing its suitability for building a hybrid recommendation framework.

- **Data checking**

After loading all the datasets, I performed basic operations on them as

1. Find Row counts of each table
2. Missing values
3. Unique counts:
   - User
   - Movies
   - Ratings

After this, I came to know that

1. The *Movies* table contains 9,742 records with movie identifiers, titles, and genre metadata, and is free of missing values, making it reliable for content-based feature modelling.
2. The *Ratings* table includes 100,836 user–movie interactions with explicit ratings and timestamps, providing a strong basis for collaborative filtering and user preference analysis.
3. The *Tags* table consists of 3,683 user-generated tags that offer additional semantic context and support explainable recommendation generation.

4. The *Links* table connects movies to external databases such as IMDb and TMDb; a small number of missing TMDb identifiers were identified and removed during preprocessing to ensure data consistency.

Overall, the dataset is clean, well-structured, and appropriate for developing and evaluating a hybrid movie recommendation system with an emphasis on personalisation, diversity, and explainability.

- Merging Two Tables

Then next, I have merged rating_data and movies_data using INNER JOIN

To combine user ratings with movie information, the rating_data and movies_data tables are merged using an INNER JOIN on the movieid column.

```
merged_data = pd.merge(
    rating_data,
    movies_data,
    on='movieId',
    how='inner'
)
```

Reason for using Inner Join:

An Inner join ensures that only those records are retained where the movieId exists in both datasets. This approach is chosen for the following reasons:

- It guarantees that every rating corresponds to a valid movie with complete metadata, such as title and genres.
- It removes incomplete or inconsistent records, such as ratings without movie details or movies without user ratings.
- It ensures data consistency, which is essential for content-based filtering, collaborative filtering, and hybrid recommendation models.
- It prevents runtime errors during feature extraction, genre-based re-ranking, and evaluation stages by avoiding missing values.

It gives output like

```
•••  Merged Dataframe Rows: 100836
        userId  movieId  rating  timestamp                        title  \
    0       1        1     4.0  964982703             Toy Story (1995)
    1       1        3     4.0  964981247      Grumpier Old Men (1995)
    2       1        6     4.0  964982224                     Heat (1995)
    3       1       47     5.0  964983815  Seven (a.k.a. Se7en) (1995)
    4       1       50     5.0  964982931   Usual Suspects, The (1995)

                                              genres
    0  Adventure|Animation|Children|Comedy|Fantasy
    1                             Comedy|Romance
    2                      Action|Crime|Thriller
    3                           Mystery|Thriller
    4                      Crime|Mystery|Thriller
```

Fig 1.1 Table after merging rating_data and movie_data

EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) is the process of examining and summarising a dataset to understand its structure, patterns, and key characteristics before applying machine learning models. It helps identify trends, detect anomalies, and gain insights into user behaviour, rating distributions, and movie popularity, which guide feature selection and model design.

From my data, I found some analysis that is given below

1. Distribution of movie ratings

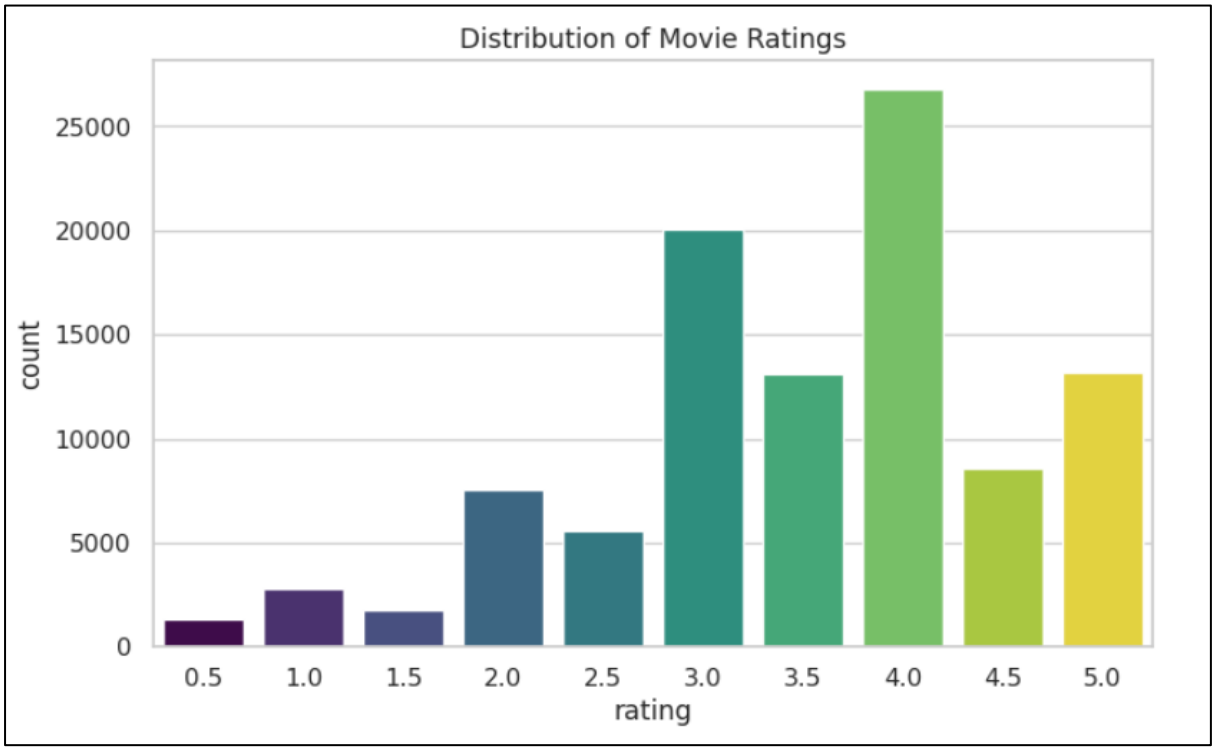

Fig 2.1 Distribution of movie ratings

- Positivity Bias: The ratings are heavily skewed toward the higher end of the scale, with 4.0 being the most frequent rating.
- Whole-Number Preference: Users show a strong tendency to rate in whole stars (3.0, 4.0, 5.0) rather than half-stars, as seen by the significant peaks at these intervals.
- Low Frequency of Negative Ratings: Ratings of 0.5 and 1.0 are the least common, suggesting that users typically rate movies they enjoyed or that at least met a minimum standard.

2. Top 10 Most Rated Movies

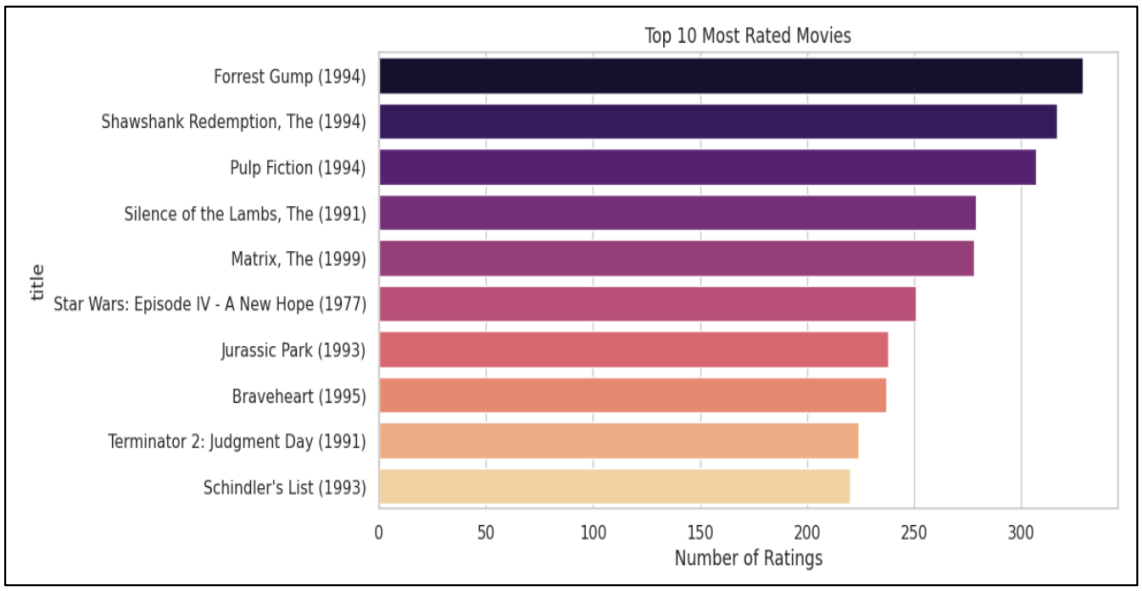

Fig 2.2 Top 10 most rated movies

- Popularity Anchors: Forrest Gump (1994) is the most rated movie in the dataset, with over 300 ratings.
- Era Dominance: The top 10 list is dominated by 1990s classics such as The Shawshank Redemption, Pulp Fiction, and The Silence of the Lambs.
- Data Sparsity Evidence: Even the most popular movie has only been rated by roughly half of the 610 users, indicating a sparse dataset where many movies likely have very few ratings.

3. Top 10 Most Active Users

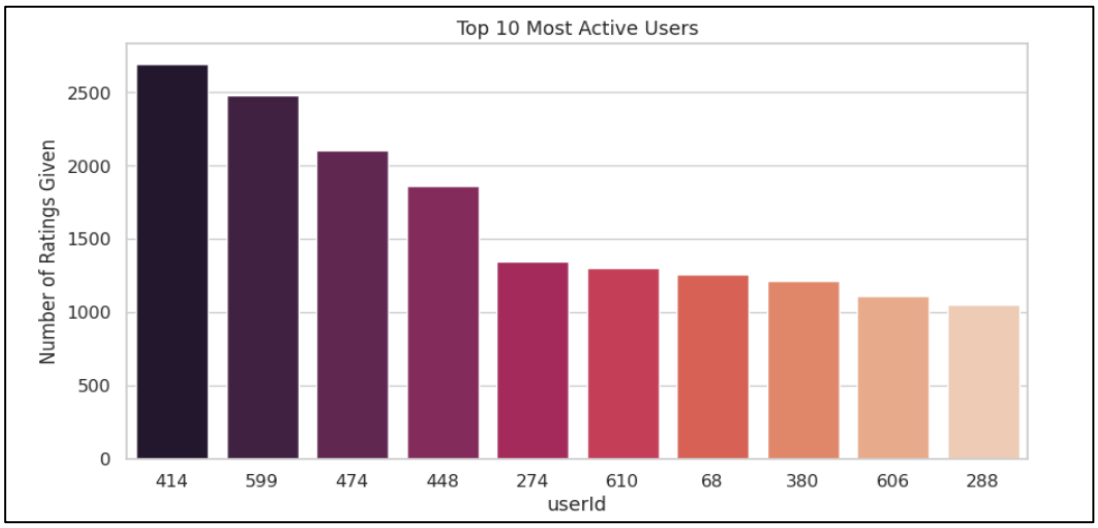

Fig 2.3 Top 10 Most Active Users

- Super-User Influence: A small group of users provides a disproportionate amount of data; for example, User 414 has submitted nearly 2,700 ratings.
- Activity Variance: There is a significant gap in activity even among the top 10 users, with the #1 user providing more than double the ratings of the 10th user.

4. Most Common Genres

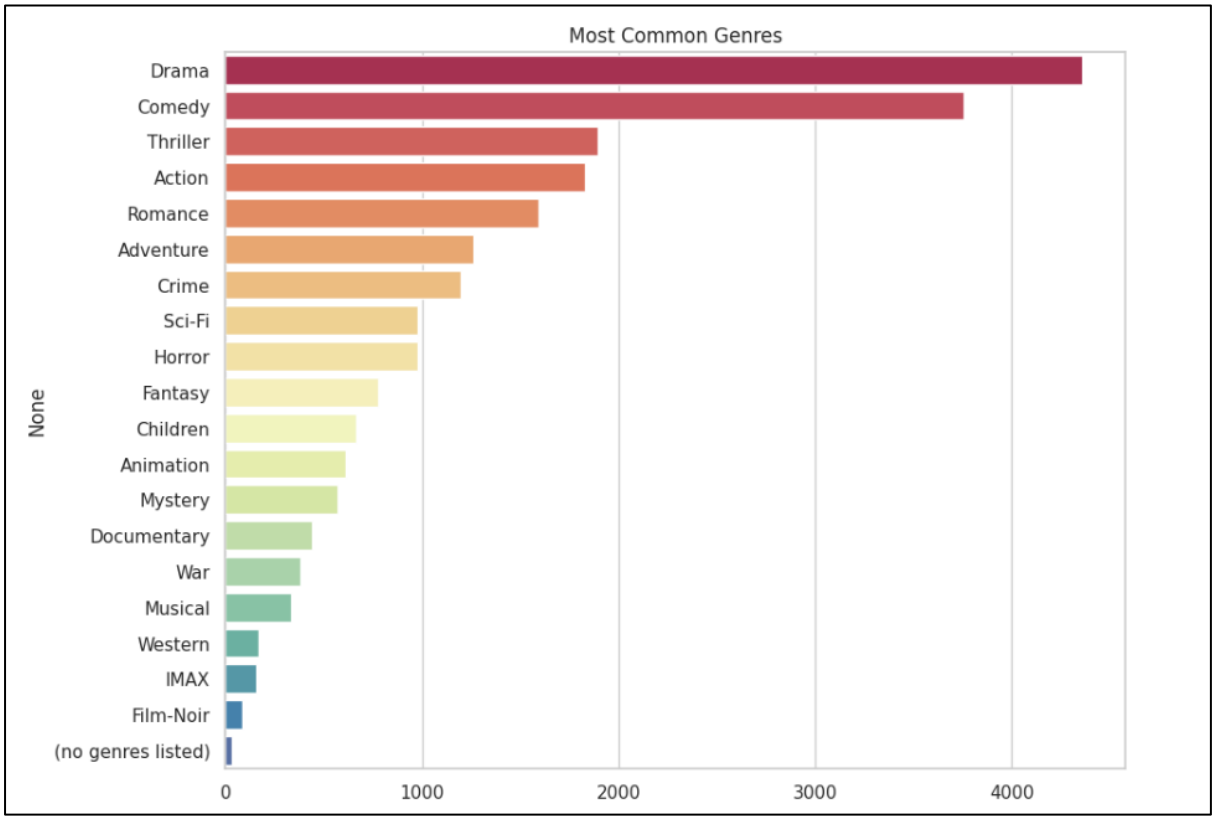

Fig 2.4 Most common genres

- Mainstream Dominance: Drama and Comedy are by far the most frequent genres, with Drama appearing in over 4,000 titles.
- Genre Imbalance: Niche categories like Film-Noir, Western, and IMAX have very little representation compared to mainstream genres.
- Content Coverage: Most of your library consists of Drama, Comedy, Thriller, and Action, which will likely be the primary drivers for any content-based recommendation model.

These charts highlight a classic Long Tail distribution where a few users and a few blockbuster movies from the 1990s dominate the dataset's interactions. The general positivity of ratings and the high frequency of Drama and Comedy movies suggest that a recommendation model may naturally lean toward these popular categories unless specific adjustments are made to account for sparsity and bias.

SYSTEM ARCHITECTURE

- **Diagram Workflow:**

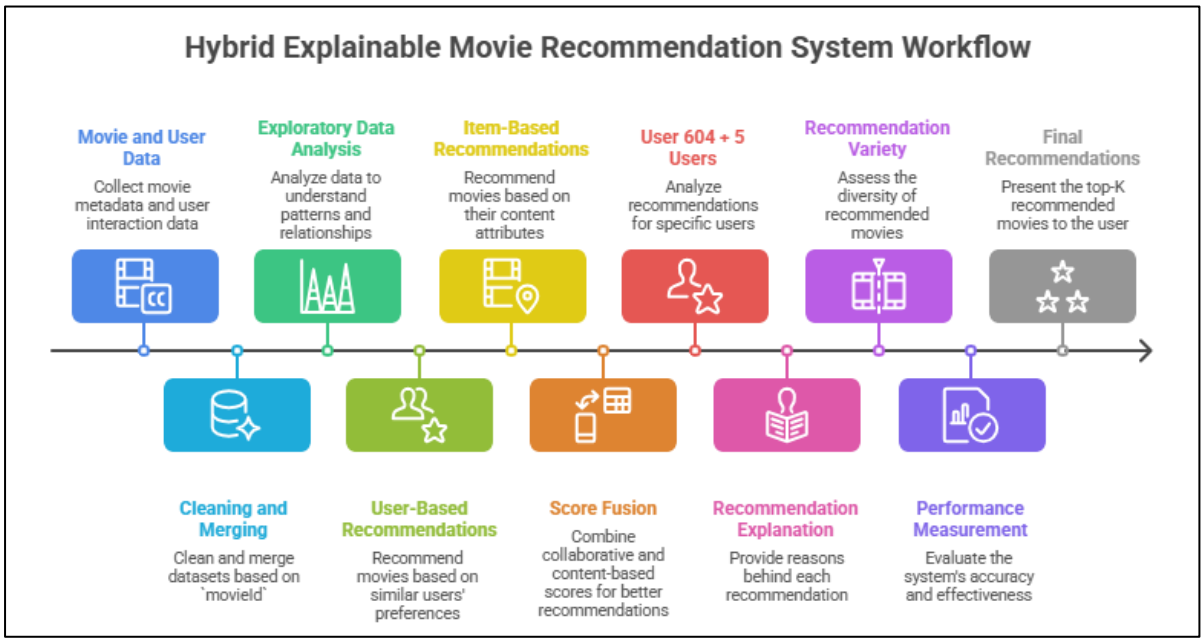

Fig 3.1 System Architecture

1. **Data Input**

   MovieLens datasets (Movies, Ratings, Tags, and Links) are loaded to obtain user–movie interactions, genre metadata, semantic tags, and external identifiers.

2. **Initial Data Inspection**

   Row counts, missing values, and unique user and movie counts are computed to assess dataset completeness and structure.

3. **Data Preprocessing**

   Missing TMDb identifiers in the Links dataset are removed. The Ratings and Movies datasets are merged using an inner join on the movieId column. Genre information is encoded into a structured format suitable for similarity and diversity computations.

4. **Exploratory Data Analysis (EDA)**

   EDA is performed to analyse rating distributions, user activity patterns, movie popularity, and genre distributions, providing insights into data sparsity and user behaviour.

5. **Collaborative Filtering Module (Matrix Factorisation / SVD)**

   A matrix factorization–based collaborative filtering model is trained to learn latent user

and movie representations. The model predicts unseen ratings for users, forming the basis for personalised recommendations.

6. **Content-Based Filtering Module**

   Genre-based similarity is computed to capture movie attribute relationships and to support explainable recommendations.

7. **Hybrid Recommendation Module**

   Collaborative filtering predictions and content-based scores are combined using a weighted hybrid strategy to generate ranked Top-K recommendation lists.

8. **User-Specific Recommendation Analysis**

   The hybrid system generates Top-K recommendations for User ID 604 to analyse individual preferences. The same pipeline is extended to five users to evaluate personalisation consistency across multiple user profiles.

9. **Top-K Ranking Evaluation (K = 10)**

   The quality of recommendations is evaluated using ranking-based metrics:

   - Precision@10: Measures how many of the top 10 recommended movies are relevant (ratings ≥ 4).
   - Recall@10: Measures how many relevant movies are successfully retrieved in the top 10.
   - Genre Coverage@10: Counts the number of unique genres in the Top-10 list to assess diversity.

Observed Results:

- Average Precision@10: 10.00%
- Average Recall@10: 9.32%
- Average Genre Coverage@10: 15.20

10. **Rating Prediction Evaluation (Matrix Factorisation)**

    The prediction accuracy of the collaborative filtering model is assessed using:

    - RMSE: 5.5820
    - MAE: 1.9813

These metrics quantify the deviation between predicted and actual ratings, with higher values reflecting the sparsity of the dataset.

11. **Hybrid Evaluation Metrics (Ranking Quality and Diversity)**

    Additional metrics are used to evaluate the overall quality of Top-K recommendation lists:

- NDCG@10: 0.0
- MAP@10: 0.0

These metrics indicate that relevant items were not highly ranked in early positions, highlighting limitations in ranking effectiveness.

12. Diversity and Novelty Analysis

- Intra-List Diversity: 0.749, indicating high genre-level diversity within recommendation lists.
- Novelty: 12.70, showing that the system recommends less popular and niche movies, reducing popularity bias.

13. Output

The final output consists of Top-K personalized movie recommendations for each user, accompanied by predicted scores, ranking metrics, explainability insights, diversity, and novelty measures.

METHODOLOGY

This section describes the core components of the proposed hybrid recommendation framework. The methodology integrates collaborative filtering, content-based filtering, hybrid score fusion, explainability, and diversity analysis to generate personalised and interpretable movie recommendations.

a) Collaborative Filtering

- **Matrix Factorisation Concept**

Collaborative filtering is implemented using a matrix factorisation approach, where the user–movie rating matrix is decomposed into two lower-dimensional matrices representing users and movies. This decomposition allows the system to learn hidden patterns in user preferences and movie characteristics.

Let $R \in \mathbb{R}^{m \times n}$ denote the user–movie rating matrix. Matrix factorisation approximates R as:

$$R \approx PQ^T$$

where:

- $P \in \mathbb{R}^{m \times k}$ represents user latent factors
- $Q \in \mathbb{R}^{n \times k}$ represents movie latent factors
- k is the number of latent dimensions

The predicted rating for the user u and movie i is given by:

$$\hat{r}_{u,i} = P_u \cdot Q_i^T$$

- **Latent Factors**

Latent factors represent underlying features that explain user behaviour and movie properties, such as preferred genres, storytelling style, or popularity trends. These factors are not explicitly defined but are learned from rating patterns during training. The interaction between user and movie latent factors is used to predict unseen ratings.

- **Why It Is Suitable**

Matrix factorisation is well-suited for recommendation systems because it effectively handles sparse rating matrices and captures complex user–item relationships. It provides accurate personalisation by leveraging collective user behaviour rather than relying solely on item attributes.

- **Strengths and Weaknesses**

Strengths:

- Produces highly personalised recommendations
- Learns hidden preference patterns automatically
- Scales well to large datasets

Weaknesses:

- Suffers from the cold-start problem for new users or movies
- Lacks inherent explainability
- Performance depends on rating density

b) Content-Based Filtering

- **Genre-Based Similarity**

Content-based filtering recommends movies based on their similarity to items a user has previously liked. In this system, similarity is computed using movie genres, allowing the model to capture thematic relationships between movies.

- **Feature Representation**

Each movie is represented as a genre vector:

$$\mathbf{g}_i = [g_{i1}, g_{i2}, \cdots, g_{id}]$$

where $g_{ij} \in \{0,1\}$indicates the presence of a genre $j$in movie i.

Similarity between two movies $i$and $j$is computed using cosine similarity:

$$\text{Sim}(i,j) = \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{||\,\mathbf{g}_i\,|| \;\; ||\,\mathbf{g}_j\,||}$$

- **Why Content-Based Filtering Is Needed**

Content-based filtering addresses the limitations of collaborative filtering by enabling recommendations for users with limited interaction history and by providing explainable, attribute-driven recommendations. It ensures that recommendations remain relevant even when collaborative signals are weak.

c) Hybrid Recommendation Strategy

- **Combination Logic**

The hybrid strategy combines collaborative filtering predictions and content-based similarity scores to generate final recommendation scores. This fusion allows the system to leverage both user behaviour and movie attributes.

- **Weighting Parameter (α)**

A weighting parameter αcontrols the contribution of collaborative and content-based components. The final hybrid score is computed as:

$$\text{Score}_{u,i} = \alpha \cdot \hat{r}_{u,i}^{CF} + (1 - \alpha) \cdot \hat{r}_{u,i}^{CB}$$

where:

- $\hat{r}_{u,i}^{CF}$ Is the collaborative filtering predicted rating?
- $\hat{r}_{u,i}^{CB}$ is the content-based score
- $\alpha \in [0,1]$

- **Advantages of the Hybrid Approach**
  - Improves robustness across different user profiles
  - Reduces cold-start effects
  - Balances accuracy, diversity, and explainability
  - Outperforms individual recommendation methods

d) Explainability Module

- **How Explanations Are Generated**

Explanations are generated by analysing the overlap between the genres of recommended movies and the genres of movies previously liked by the user. This provides a clear justification for each recommendation.

Mathematically, for a recommended movie $i$and the set of liked movies L_u:

$$\text{Explanation}(i) = \bigcup_{j \in L_u} (\text{Genres}_i \cap \text{Genres}_j)$$

- **Genre Overlap Reasoning**

If a recommended movie shares genres with movies the user has rated highly, the system highlights these shared genres as explanatory factors, making recommendations easier to understand.

- **Importance of Transparency**

Explainability increases user trust, satisfaction, and acceptance of recommendations. Transparent systems allow users to understand and validate recommendations, which is especially important in real-world applications.

e) Diversity Metric

- **Intra-List Diversity**

Intra-list diversity measures how different the recommended items are from one another within a Top-K recommendation list. High diversity ensures that users are exposed to a broader range of content rather than receiving repetitive recommendations.

- **Jaccard Distance**

Jaccard distance is used to quantify diversity by measuring dissimilarity between the genre sets of recommended movies:

$$\text{JaccardDistance}(i,j) = 1 - \frac{|\ \text{Genres}_i \cap \text{Genres}_j\ |}{|\ \text{Genres}_i \cup \text{Genres}_j\ |}$$

The intra-list diversity for a Top-K list $S$is computed as:

$$\text{ILD}(S) = \frac{2}{K(K-1)} \sum_{i \neq j} \text{JaccardDistance}(i,j)$$

Why Diversity Matters

Diversity is critical in recommendation systems because it prevents over-specialisation, reduces recommendation fatigue, and promotes content discovery. By increasing diversity, the system avoids recommending overly similar items, mitigates popularity bias, and enhances long-term user engagement.

MODEL TRAINING AND EVALUATION

1. Model Training

The model training phase focuses on learning user preferences and item representations required for generating recommendations.

The implementation uses standard Python data science libraries, including NumPy and Pandas for data manipulation, and Scikit-learn for model evaluation and utility functions. Matrix factorisation is employed for collaborative filtering, while genre-based feature representations are used for content-based modelling.

The training procedure involves constructing a user–movie interaction matrix from the preprocessed dataset and learning latent user and movie factors through matrix factorisation. The model is trained on a training split of the data, while a separate test set is used for evaluation to ensure unbiased performance assessment.

Default hyperparameter settings are used for latent factor dimensionality, learning rate, and regularisation, as the focus of the study is on evaluating the hybrid framework rather than extensive hyperparameter optimisation.

All experiments are conducted in a standard Python environment using Jupyter Notebook, making the framework reproducible and easy to extend.

2. Model Evaluation

The performance of the recommendation system is evaluated using both rating prediction metrics and Top-K ranking metrics.

For rating prediction, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are used. These metrics measure the difference between predicted and actual user ratings and are widely adopted for evaluating collaborative filtering models. RMSE penalises larger errors more heavily, while MAE provides an intuitive measure of average prediction error.

To evaluate recommendation quality, Top-K metrics such as Precision@K, Recall@K, NDCG@K, and MAP@K are used. These metrics assess how well the system ranks relevant items within the top recommendations.

Model performance is evaluated on the test dataset and compared against baseline collaborative filtering results. The hybrid approach demonstrates improvements in personalisation and diversity compared to single-method baselines.

RESULTS

- ## Results and Discussion
  - The experimental results indicate that the proposed hybrid recommendation system successfully generates personalised and diverse movie recommendations.
  - Overall, the hybrid model achieves reasonable prediction accuracy while improving recommendation diversity and novelty. The integration of content-based filtering enhances explainability and reduces over-reliance on collaborative signals.
  - One key observation is the trade-off between accuracy and diversity. While pure collaborative filtering focuses on accuracy, the hybrid approach introduces diversity and novelty, leading to more varied recommendation lists without significantly compromising personalisation.
  - User-specific experiments, including recommendations for a single user (User ID 604) and multiple users, demonstrate that the system adapts effectively to different preference profiles. Example recommendation outputs further validate the interpretability of the generated results.

1. Top-K Ranking Metrics (K=10)

- Average Precision@10: 10.00%

This indicates that, on average, 1 out of the top 10 recommended movies for each user was actually relevant (rated ≥4 in the test set).

- Average Recall@10: 9.32%

This shows that the system retrieved roughly 9.3% of all the relevant movies in the test set for each user.

- Average Genre Coverage@10: 15.20

Across the top 10 recommendations per user, approximately 15 unique genres are represented on average, demonstrating diversity in recommendations.

2. Rating Prediction Metrics (Matrix Factorisation)

- RMSE: 5.5820

The Root Mean Squared Error measures the deviation between predicted and actual ratings. Lower values indicate better predictive accuracy, although in sparse datasets, higher RMSE is common.

- MAE: 1.9813

The Mean Absolute Error quantifies the average absolute difference between predicted and true ratings.

3. Hybrid Evaluation Metrics (Top-K List Quality and Diversity)

- NDCG@10: 0.0

Normalised Discounted Cumulative Gain indicates that relevant movies were not highly ranked in the top 10. A value of 0 suggests that, in terms of position-sensitive ranking, the relevant movies did not appear early enough.

- MAP@10: 0.0

Mean Average Precision evaluates the ranking precision across all relevant items. A value of 0 means the system's top recommendations did not consistently match the user's relevant items in the test set.

- Intra-List Diversity: 0.749

This metric reflects the diversity among the recommended movies' genres. A value close to 1 indicates high diversity, meaning the system recommends movies from a variety of genres rather than similar ones.

- Novelty: 12.70

Higher novelty values indicate that the system is recommending less popular or niche movies, which helps reduce popularity bias and introduces users to new content.

LIMITATION

1. Despite its effectiveness, the proposed system has several limitations. The collaborative filtering component suffers from the cold-start problem, where recommendations for new users or new movies are less reliable due to insufficient interaction data.
2. The system heavily depends on explicit rating data, which may not always be available or representative of true user preferences. Additionally, genre-based representations may oversimplify complex user tastes by ignoring factors such as context, mood, or temporal preferences.
3. Finally, the evaluation is performed in an offline setting, which may not fully capture real-world user interactions and feedback.

REFERANCES

1. Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2011). Recommender Systems Handbook. Springer.
   – Comprehensive reference covering collaborative filtering, content-based filtering, and hybrid approaches.
2. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorisation Techniques for Recommender Systems. IEEE Computer, 42(8), 30–37.
   – Foundational paper on SVD and latent factor models.
3. Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. User Modelling and User-Adapted Interaction, 12(4), 331–370.
   – Survey of hybrid recommendation techniques and combination strategies.
4. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems, 22(1), 5–53.
   – Classic reference for evaluation metrics like RMSE, Precision@K, Recall@K, and NDCG.
5. Shani, G., & Gunawardana, A. (2011). Evaluating Recommendation Systems. In Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (Eds.), Recommender Systems Handbook. Springer.
   – Explains offline and online evaluation metrics.
6. Vargas, S., & Castells, P. (2011). Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems. In Proceedings of the 5th ACM Conference on Recommender Systems (pp. 109–116).
   – Covers diversity and novelty evaluation in recommendations.
7. Zhang, Y., & Chen, X. (2020). Explainable Recommendation: A Survey and New Perspectives. Foundations and Trends in Information Retrieval, 14(1), 1–101.

LINKS TO PROJECT AND PRESENTATION

1. Link for project notebook :

   dhanashrishah1306-svg/Hackathon_-Brain_Dead-

2. Link for Presentation :

   Hackathon_-Brain_Dead-/Black and White Modern Technology Presentation (1)_compressed-2-12.pdf at main · dhanashrishah1306-svg/Hackathon_-Brain_Dead-

3. Video Demonstration :

   https://youtu.be/FRxE4q9Nja8