PROJECT

MOVIE RATING

Presented By: NARMATHA.S PAAVAI College of Engineering-
ELECTRICAL AND ELECTRONICS ENGINEERING

edunet foundation

OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

PROBLEM STATEMENT

- Design and implement a movie rating project that allows users to rate and review movies, as well as discover new films based on their preferences.

PROPOSED SOLUTION

- The output is a dynamic and interactive platform that enhances users' movie-watching experience by providing a comprehensive database, personalized recommendations, and social interaction features.

SYSTEM APPROACH

1. Requirement Analysis: - Understand the project goals, features, and user expectations. - Gather detailed requirements through discussions with stakeholders.
2. Database Design: - Design a relational database schema to store movie details, user information, ratings, reviews, and other relevant data. - Normalize the database to minimize redundancy and ensure data integrity.
3. Backend Development: - Choose a suitable backend technology stack (e.g., Node.js, Python/Django, Ruby on Rails) based on project requirements and team expertise. - Develop REST ful APIs to handle user authentication, movie CRUD operations, rating submission, review posting, and recommendation generation. - Implement business logic to calculate average ratings, generate recommendations, and handle user interactions.
4. Testing and Quality Assurance: - Conduct unit tests, integration tests, and end-to-end tests to ensure the functionality, performance, and security of the application. - Perform usability testing to gather feedback from users and identify areas for improvement.

ALGORITHM & DEPLOYMENT

- 1. User Preferences Initialization: - When a new user signs up, initialize their preferences by assigning weights to different movie genres, directors, actors, etc. These weights will determine the user's initial preference profile.
- 2. User Ratings Collection: - Collect ratings from users for movies they have watched. Ratings can be on a scale of 1 to 5 stars.
- 3. Rating Normalization: - Normalize user ratings to adjust for individual rating scales and biases. For example, if a user tends to rate movies higher or lower than average, normalize their ratings to a common scale.
- 4. Similarity Calculation: - Calculate the similarity between users based on their rating patterns. Use similarity metrics such as cosine similarity, Pearson correlation coefficient, or Jaccard similarity. - For each pair of users, compute their similarity score by comparing their normalized rating vectors.
- 5. Neighborhood Selection: - Select a neighborhood of similar users for each target user. Define a threshold or fixed number of nearest neighbors to consider. - Identify the users whose rating patterns are most similar to the target user's pattern.

ALGORITHM & DEPLOYMENT

- 6. Rating Prediction: - Predict the target user's ratings for movies they have not yet rated based on the ratings of their neighbors. - Weight the ratings of neighbors based on their similarity to the target user. Closer neighbors should have more influence on the prediction.
- 7. Top-N Recommendations: - Generate a list of top-N recommended movies for the target user based on the predicted ratings. - Sort the unrated movies by their predicted ratings in descending order. - Recommend the top-N movies with the highest predicted ratings to the user.
- 8. Cold Start Handling: - Handle the "cold start" problem for new users or movies with limited ratings by using alternative approaches such as content-based recommendations or popularity-based recommendations.
- 9. Feedback Incorporation: - Continuously update user preferences and refine recommendations based on user feedback. Allow users to provide explicit feedback (e.g., like/dislike) on recommended movies to improve future recommendations.
- 10. Re-ranking and Personalization: - Re-rank recommended movies based on additional factors such as release date, genre diversity, or popularity to provide a diverse selection to the user. - Personalize recommendations further by considering contextual information such as time of day, user location, or recent user activity.
- 11. Evaluation and Optimization:*- Evaluate the performance of the recommendation algorithm using metrics such as precision, recall, and mean average precision. - Optimize the algorithm parameters, similarity metrics, and neighborhood selection criteria to improve recommendation quality and user satisfaction.

RESULT

1. Structured Dataset: - A well-organized dataset containing information about movies, user ratings, and user profiles. This dataset would include attributes such as movie titles, genres, release years, directors, cast members, user IDs, and corresponding ratings
2. Data Cleaning and Preprocessing: - Processed and cleaned dataset free from inconsistencies, missing values, and outliers. Preprocessing steps may include data normalization, handling null values, and resolving duplicate entries.

CONCLUSION

- User Engagement: The project witnessed high user engagement, as evidenced by the number of active users, ratings, reviews, and interactions within the platform.
- Personalized Recommendations: The recommendation engine effectively generated personalized movie recommendations for users, enhancing their movie-watching experience and satisfaction.

FUTURE SCOPE

1. Advanced Recommendation Algorithms.
2. User Interaction Features
3. Social Integration
4. Internationalization
5. Monetization Strategies
6. Data Analytics and Insights:
7. Accessibility Improvements.
8. Virtual Reality (VR) Integration

REFERENCES

- Movie Lens:
- IMDb
- TMDB
- Kaggle

THANK YOU

edunet foundation