MOVIELENS RECOMMENDATION SYSTEM.

NON-TECHNICAL PRESENTATION.

* 1. BUSINESS UNDERSTANDING.

Recommendation systems play a crucial role in the success of streaming platforms like Netflix, Amazon Prime and Hulu. By analyzing users' viewing histories and preferences, these systems facilitate the discovery of new content, which in turn boosts user engagement and satisfaction. This project leverages the MovieLens dataset, created by the GroupLens Research Lab at the University of Minnesota, which is one of the most extensively used datasets for developing and testing recommendation systems in the movie industry. It contains millions of movie ratings from users, along with detailed information about the movies, such as titles, genres, and release years.

There are, however, major challenges in building a recommendation system that require strategic solutions. One major challenge is dealing with the sparsity and cold start problem. The MovieLens dataset and other similar datasets typically suffer from sparsity, where most users have rated only a small fraction of all available movies. This sparsity makes it challenging to accurately predict user preferences, especially for new or less active users who have rated very few movies (cold start problem). A solution to address data sparsity is to implement item-item collaborative filtering whereas a way to address the cold start problem is to integrate content-based filtering techniques. The metric of success of the model should be an accuracy score of at least 80%.

In conclusion, by leveraging collaborative filtering and hybrid methods, we are looking to obtain a comprehensive recommendation system that provides personalized movie suggestions that not only suggests movies based on user ratings and preferences but also incorporates real-time feedback to continually improve the accuracy and relevance of recommendations.

Business problem.

We have been tasked to develop a personalized movie recommendation system that maximizes user satisfaction and engagement, taking into account their ratings on other movies. For this analysis, we’re leveraging the MovieLens dataset that contains information on movies and explicit ratings by the users. The goal is to increase user satisfaction, encourage longer engagement on the platform, and potentially increase revenue through improved user retention and targeted content promotion.

Objectives.

The MAIN objective is to build a model that provides top 5 movie recommendations to a user, based on their ratings of other movies.

The specific objectives are:

1. To implement a collaborative filtering algorithm to analyze user ratings and identify similarities between users or movies.
2. To integrate content-based filtering techniques to enhance recommendation quality by addressing the cold start problem.
3. To evaluate the model's performance using appropriate metrics such as RMSE and MAE.
   1. DATA UNDERSTANDING.

This project analysis uses the MovieLens dataset, created by the GroupLens Research Lab at the University of Minnesota, which contains information on movies and explicit ratings by the users.

The dataset is a folder with csv files (movies, ratings, links and tags) downloaded from: <https://grouplens.org/datasets/movielens/latest/>

This dataset(movielens\_dataset) describes 5-star rating and free-text tagging activity from MovieLens (<http://movielens.org/>), a movie recommendation service.

It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018. This dataset was generated on September 26, 2018.

Users were selected at random for inclusion. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id, and no other information is provided.

The relevant files in our analysis are:

1. Movies dataset

Movie information is contained in the file “movies.csv”. Each line of this file after the header row represents one movie, and has the following format: movieId, title, genres

It contains 9742 rows and 3 columns.

Genres are a pipe-separated list, and are selected from the following:

* Action
* Adventure
* Animation
* Children's
* Comedy
* Crime
* Documentary
* Drama
* Fantasy
* Film-Noir
* Horror
* Musical
* Mystery
* Romance
* Sci-Fi
* Thriller
* War
* Western

1. Ratings dataset

All ratings are contained in the file “ratings.csv”. Each line of this file after the header row represents one rating of one movie by one user, and has the following format: userId, movieId, rating, timestamp

It contains 100836 rows and 4 columns.

Ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars)

* 1. DATA PREPARATION.
  2. Movies dataset.

The shape; 9742 rows and 3 columns.

The info; movieId has 9742 non-null integers.

title and genres have 9742 non-null objects (typically strings).

This dataset contains no duplicates and no missing values.

* 1. Ratings dataset.

The shape; 100836 rows and 4 columns.

The info; userId, movieId and timestamp have 100836 non-null integers.

rating has 100836 non-null floats.

This dataset contains no duplicates and no missing values.

After using boxplots to display whether there are outliers or not in the numerical columns (rating and timestamp), the analysis indicates the absence of significant outliers due to the inherent characteristics of these columns.

* 1. DATA ANALYSIS.

First, we defined a function **clean\_title** to remove all non-alphanumeric characters from movie titles and then applied this function to every title in the 'title' column.

For feature engineering, a vectorizer is applied to the cleaned movie titles, transforming them into a TF-IDF feature matrix, which is a numerical representation of the text data based on term frequency and inverse document frequency.

For EDA, we found the distribution of movie ratings and from the boxplot, it is evident that most movies were rated 4.0 and the least number of movies were rated 0.5.

The Mean global rating is 3.5 and the Mean rating per user is 3.66

* 1. MODELLING.

1. Content-based filtering.

We created a function **clean\_title** to remove special characters from movie titles before applying TF-IDF (Term Frequency-Inverse Document Frequency) which is crucial for accurate and effective content-based filtering. It ensures that the text data is clean and consistent, leading to more reliable TF-IDF scores and, consequently, better recommendations based on movie titles.

1. Collaborative filtering.

We created a function **collaborative\_filter** that recommends movies through identifying users who have highly rated a given movie and then suggesting other movies highly rated by those similar users, while also considering the overall popularity of these movies.

Then there’s a test function for finding movie recommendations based on a given movie title. As an observation, the recommendations for the movie "Jumanji" suggest similar films based on user preferences and ratings, providing a list of movies with high ratings and relevant genres such as “Dragonheart”.

We created an interactive widget for movie recommendations and then made a function **show\_movie\_suggestions** that processes the users input and updates the recommendations.

* 1. EVALUATION.

We prepare the dataset for a collaborative filtering model by;

* defining the rating scale.
* converting the data into a suitable format.
* splitting it into training and testing sets.

We set up a user-based collaborative filtering model, train it, make predictions and evaluate accuracy with RMSE. As a result, an RMSE of 0.98 is obtained which is relatively high, suggesting that the model's predictions may not be very accurate. Typically, an RMSE closer to 0 is preferred.

Tuning

We used GridSearchCV for the hyper parameter tuning.

We then get the best RMSE score from the grid search which was 0.9048 and Best Parameters: k: 70, Similarity Metric: 'msd' (Mean Squared Difference), User-Based: False

**Cross-validate** for further improvement and refinement of the mode. First, Extract the best model from grid search and then perform cross-validation with the best model. The results were;

* RMSE (Root Mean Square Error): 0.9053 (average across the 5 folds)
* MAE (Mean Absolute Error): 0.6964 (average across the 5 folds)

These values suggest that the model has a reasonable level of accuracy in predicting the ratings, with relatively low errors.

* 1. DEPLOYMENT.

First, use the best hyper parameters obtained from tuning to optimize the recommendation accuracy. Then, make **find\_similar\_movies** function to find and return movies similar to a given movie, identified by movie\_id. It will use the trained KNNBasic collaborative filtering model to find similar movies based on user ratings. By entering the movie id, you get similar movies (the title, genre and rating).

* 1. CONCLUSION.

The completion of the personalized movie recommendation system, utilizing the MovieLens dataset, has achieved its primary objective of delivering tailored movie suggestions based on user ratings. By integrating both collaborative filtering and content-based filtering techniques, the system effectively addresses various challenges including user preference modeling and the cold start problem.

* *Accuracy*: The model’s performance was evaluated using RMSE as an accuracy metric. The system demonstrated a strong ability to predict user ratings accurately, reflecting close alignment with user preferences.
* *Diversity*: The 5 recommendations were varied and avoided excessive similarity, contributing to a more engaging user experience.
* *Novelty*: The system introduced users to new or less-known movies, enhancing the freshness of recommendations.

The successful implementation of these features and metrics validates the effectiveness of the recommendation system in enhancing user experience and driving platform engagement.

* 1. RECOMMENDATIONS.

1. *Ongoing Optimization*: Continue to refine and optimize the collaborative filtering and content-based algorithms based on ongoing user feedback and evolving data to maintain high recommendation quality.
2. *Regular Updating*: Regularly update the model with new user ratings and movie data to ensure that recommendations stay relevant and reflect current user preferences.
3. *User Feedback Integration*: Systematically collect and analyze user feedback to identify areas for improvement and make data-driven adjustments to continue enhancing satisfaction.
4. *Explore Advanced Techniques*: Investigate the integration of additional advanced recommendation techniques, such as deep learning models or reinforcement learning, to further improve recommendation accuracy and personalization.
5. *Enhance Diversity and Novelty*: Continue to focus on increasing the diversity and novelty of recommendations to keep the user experience engaging and prevent content stagnation.
   1. FUTURE WORK.
6. *Ongoing Optimization*: Implement adaptive learning and dynamic parameter tuning to refine recommendation algorithms based on real-time user feedback and evolving data patterns.
7. *Regular Updating*: Integrate real-time data processing for instant model updates and establish version control to manage and stabilize recommendations.
8. *User Feedback Integration*: Create robust feedback loops and use sentiment analysis to systematically incorporate user interactions and refine recommendations.
9. *Explore Advanced Techniques*: Investigate deep learning, reinforcement learning, and hybrid models to enhance recommendation accuracy and personalization.
10. *Enhance Diversity and Novelty*: Track diversity metrics, balance exploration and exploitation, and use novelty algorithms to keep recommendations fresh and engaging.