GbGibson7 Updating the presentations 668c8aa · 1 minute ago

data	Updating the key findings and READ...	2 hours ago
.gitignore	Initial commit	3 days ago
Movie-Recommendation-Syst...	Updating the notebook and READM...	1 hour ago
Presentations.pdf	Updating the presentations	1 minute ago
README.md	Resolved merge conflicts in README...	1 hour ago

README

Phase-4-Project-Movie-Recommendation-System

This project delves into the classic MovieLens dataset to build a robust movie recommendation system. Leveraging various machine learning techniques such as Linear Regression, K-Nearest Neighbors Regresson and User-Based Collaborative Filtering as well as following the CRISP-DM methodology, the aim is to predict user preferences and deliver personalized top-5 movie recommendations. This repository showcases the end-to-end process from data understanding to model evaluation, emphasizing model training best practices with pipelines and hyperparameter tuning.

1. Business Understanding

The core objective is to develop a system that provides users with tailored movie recommendations based on their past ratings. Imagine this system enhancing a streaming platform's user experience, increasing engagement by surfacing films users are most likely to enjoy.

2. Data Understanding

Initial exploration of the MovieLens "small" dataset (movies.csv, ratings.csv, links.csv, tags.csv) revealed key characteristics:

- Movies: Over 9,700 titles with genres.
- Ratings: Over 100,000 ratings from 610 users, showcasing a clear positive bias towards higher scores (3.0, 4.0, 5.0 are most common).
- Popularity Skew: A few blockbusters receive a disproportionately high number of ratings, while many films have very few. Visual analysis (histograms and joint plots) confirmed that average ratings for scarcely reviewed movies can be misleadingly extreme, necessitating data filtering.

Data Challenges

- Filtering is needed to exclude movies with few ratings(<20) to mitigate noice in recommendation mode
- Potential bias with popular movies dominating ratings --> niche titles maybe underrepresented.

3. Data Preparation

This phase focused on transforming raw data into a usable format for machine learning models. Key steps included:

- Merging: Combining movie metadata with user ratings to create a comprehensive working DataFrame.
- Matrix Creation: Constructing sparse user-item matrices (users as rows, movies as columns) for collaborative filtering, with NaN values representing unrated movies.
- Feature/Target Split: Separating userId and movieId as features (X) from rating as the target (y) for sklearn models.
- Data Splitting: Dividing the dataset into training and testing sets to ensure robust model evaluation on unseen data.

4. Exploratory Data Analysis(EDA)

Visual and statistical analysis provided critical insights:

- The distribution of ratings heavily favors positive scores, confirming a common user behavior.
- The number of ratings per movie is heavily right-skewed, showing that a few popular movies accumulate most reviews, while many remain obscure.
- Average ratings per movie (filtered for reliability) reveal a more stable distribution, concentrating around higher scores for widely-rated films.
- A joint plot graphically demonstrated that movies with fewer ratings often have extreme, less reliable average scores, reinforcing the need for rating thresholds.

5. Modelling

This project implemented and compared multiple recommendation models:

- Linear Regression (with Pipeline): A baseline sklearn regression model predicting ratings, incorporating StandardScaler in a pipeline for robust feature scaling.
- K-Nearest Neighbors Regressor (with Pipeline and GridSearchCV): A more sophisticated sklearn regression model, also using a pipeline for scaling. Its n_neighbors hyperparameter was optimized using GridSearchCV to find the most effective number of neighbors for prediction. The optimal n_neighbors was determined to be 15, yielding a significantly better performance than Linear Regression.
- User-based Collaborative Filtering: Recommends movies by finding users with similar taste profiles (using Cosine Similarity) and suggesting what they liked.

6. Evaluation

The models' performance was rigorously assessed using Root Mean Squared Error (RMSE), which quantifies the average magnitude of prediction errors in terms of rating points. RMSE for sklearn models was evaluated on clipped predictions (0.5-5.0) for realistic comparison. Key Findings (RMSE):

- User-based Collaborative Filtering: Achieved an RMSE of approximately 0.9690.
- K-Nearest Neighbors Regressor (Tuned): Demonstrated strong performance with an RMSE of approximately 0.9809 (clipped).
- Linear Regression: Served as a baseline with an RMSE of approximately 1.0466 (clipped).

The lower RMSE values indicated better predictive accuracy, with User-based CF and tuned KNN generally outperforming simple Linear Regression for this dataset.

Contributors:

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2


GbGibson7


RGitau-coder RGitau-RG

Languages

- Jupyter Notebook 100.0%