SteveNgatia /
MovieLens-Recomender-System

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Movies Recomender System With an App

0 stars 0 forks 1 watching 1 Branch 0 Tags Activity

Public repository

main 1 Branch 0 Tags Go to file t Go to file + Add file Code ···

SteveNgatia Initial Commit 1 minute ago

Phase4 Recomender System.ipynb	Initial Commit	1 minute ago
Presentation.pdf	Initial Commit	1 minute ago
README.md	Initial Commit	1 minute ago
app.py	Initial Commit	1 minute ago
model	Initial Commit	1 minute ago
movie_ratings_pivot_df	Initial Commit	1 minute ago
movies.csv	Initial Commit	1 minute ago
notebook.pdf	Initial Commit	1 minute ago
ratings.csv	Initial Commit	1 minute ago

README

Movie-Recommendation-System-Project

Overview

This project aims to develop a recommendation system for movies using the MovieLens dataset. The system leverages collaborative filtering, content-based filtering, and hybrid approaches to provide personalized movie recommendations to users based on their past ratings and preferences.

Table of Contents

- Overview
- Data
- Data Preprocessing
- Exploratory Data Analysis
- Building Models
- Evaluation
- Conclusion and Recommendations

Data

- MovieLens : Contains the MovieLens dataset.
- This dataset was obtained from the GroupLens research lab at the University of Minnesota.

Data Preprocessing

The dataset is preprocessed to handle duplicates, missing values, and format the data for modeling.

Exploratory Data Analysis

Exploratory data analysis is conducted to understand user behavior, movie characteristics, and distribution of ratings.

Building Models

Collaborative Filtering

- Implemented k-Nearest Neighbors algorithm for collaborative filtering.
- Utilized Surprise library for model development and evaluation.
- Explored various similarity metrics and hyperparameters to optimize model performance.

Content-Based Filtering

- Leveraged movie metadata (genres) to develop content-based filtering.
- Utilized TF-IDF vectorization and cosine similarity for content-based recommendations.
- Addressed the cold-start problem by recommending popular movies for new users.

Hybrid Approach

- Combined collaborative filtering and content-based filtering for improved recommendation accuracy.
- Developed a hybrid recommendation system that adapts to user preferences and item characteristics.

Reproducibility

To access the data used in this project, simply click on the files provided under the repository and the files will be downloaded to your local computer.

Evaluation

- Evaluated model performance using metrics such as RMSE and MAE for collaborative filtering.
- Utilized cross-validation and grid search for hyperparameter tuning.
- Analyzed the strengths and limitations of each recommendation technique.

Conclusion and Recommendations

Conclusion

In this project, we developed a movie recommendation system using collaborative filtering and content-based filtering techniques on the MovieLens dataset. Through analysis, we gained insights into user preferences and movie characteristics, identifying popular and high-quality movies. Leveraging movie metadata and user ratings, our content-based filtering provided personalized recommendations based on genre similarities. Concurrently, collaborative filtering utilized k-Nearest Neighbors to recommend movies liked by similar users, overcoming challenges like the cold-start problem for new users. Evaluation metrics confirmed the accuracy and robustness of our models, laying the groundwork for a versatile recommendation engine adaptable to various movie recommendation scenarios.

In summary, our project successfully implemented recommendation systems that combine collaborative and content-based filtering for personalized movie recommendations. By optimizing parameters and addressing challenges like the cold-start problem, we've developed a robust framework for accurate movie suggestions.

Recommendations

1. Continue exploring and fine-tuning the hybrid model, as it leverages the strengths of both collaborative filtering and content-based filtering, potentially leading to more accurate and diverse recommendations.
2. Investigate additional movie metadata features (e.g., directors, actors, plot summaries) and incorporate them into the content-based filtering component to enhance the recommendation quality.
3. Explore other recommendation algorithms and techniques, such as matrix factorization, deep learning-based approaches, or graph-based methods, to potentially improve the recommendation accuracy further.

Overall, the project successfully developed a recommendation system for movies, demonstrating the potential of combining collaborative filtering and content-based filtering techniques. By continuously refining and enhancing the

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- Jupyter Notebook 97.3%
- Python 2.7%

Suggested workflows

Based on your tech stack

Python Package using Anaconda Configure

Create and test a Python package on multiple Python versions using Anaconda for package management.

Publish Python Package Configure

Publish a Python Package to PyPI on release.

Pylint

Configure

Lint a Python application with pylint.

More workflows

Dismiss suggestions