

yesimcebeci / movie-recommender

Public

0 stars

0 forks

Star

Unwatch

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

main

yesimcebeci

Final Notebook

6 minutes ago

34

View code

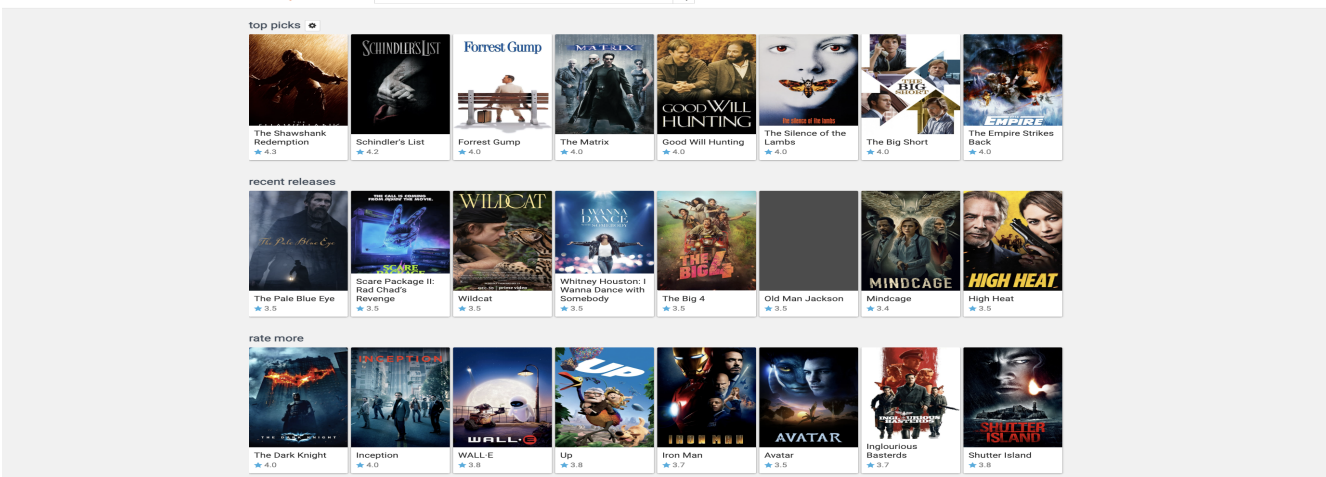
README.md

Movie-Recommender

In our daily life when we are shopping online, or looking for a movie to watch, we normally ask our friends or search for it. So what about if there is a system that can understand you, and recommend for you based on your interests, that would be soo cool isn't it. Well, that exactly what recommender systems are made for.

This notebook is going to explain how I worked throughout the entire life cycle of recommender system, and provide my solutions to some technical issues.

For the recommender system I will use Content-based and Model Based Collaborative filtering . Recommender systems are built on MovieLens dataset with 100,000 movie ratings. These Recommender systems were built using Pandas operations and by fitting Machine Learning models to suggest movies for the users based on similar users and for queries specific to genre, user, movie, rating.



The screenshot shows the MovieLens website interface. At the top, there's a search bar and navigation links. Below, there are three sections: 'top picks', 'recent releases', and 'rate more'. Each section displays a grid of movie posters with their titles and ratings. For example, in 'top picks', movies like 'The Shawshank Redemption' (4.3), 'Schindler's List' (4.2), and 'Forrest Gump' (4.0) are shown. 'recent releases' includes 'The Pale Blue Eye' (3.5), 'Scare Package II: Red Chad's Revenge' (3.5), and 'Wildcat' (3.5). 'rate more' features 'The Dark Knight' (4.0), 'Inception' (4.0), 'WALL-E' (3.9), and 'Up' (3.9).

Table of Content

- Business Understanding
- Data Understanding

- [Data Analysis](#)
- [Recommendation Engine](#)
- [Conclusion and Future Consideration](#)
- [References](#)
- [For More Information](#)

Business Understanding

MovieLens is a website and virtual community that recommends movies for its users to watch movies based on users' movie ratings and movie reviews

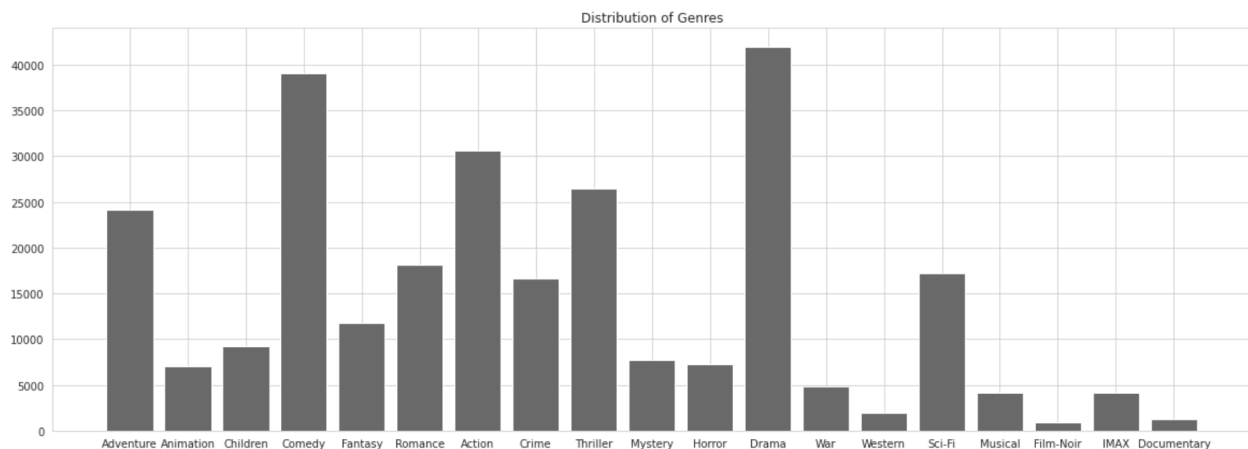
In this project, I will build recommendation engines and improve upon predictions iteratively so that the end user can be provided with better movie suggestions on their homepages.

Data Understanding

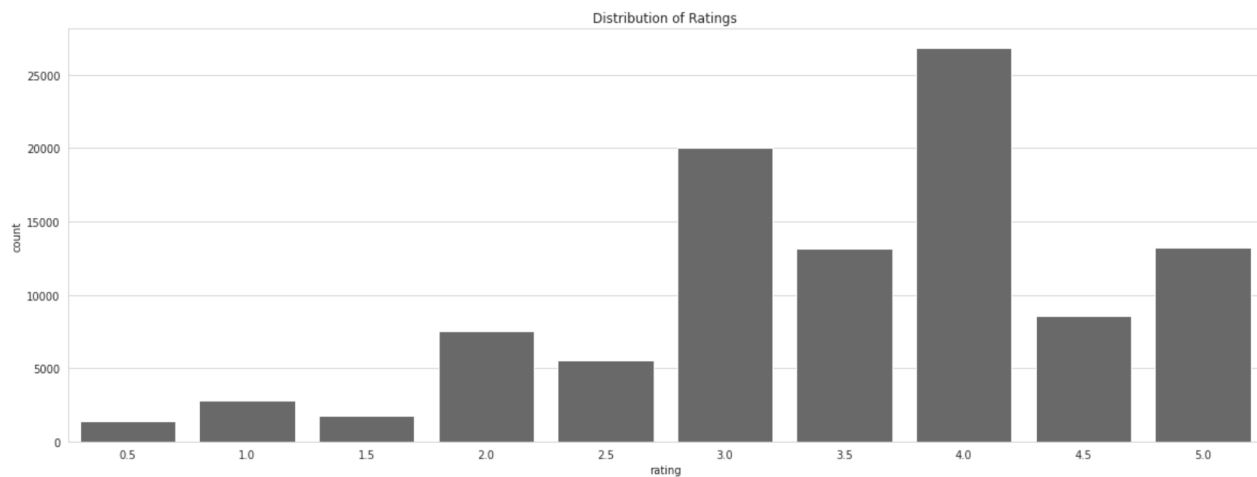
The dataset that I'm working with is MovieLens, one of the most common datasets that is available on the internet for building a Recommender System. The version of the dataset that I'm working with (ml-latest-small) contains 100836 ratings of 9742 movies made by 610 MovieLens users who joined MovieLens in 2018.

Data Analysis

After processing the data and doing some exploratory analysis, here are the most interesting features of this dataset:



- The top 5 genres are, in that respect order: Drama, Comedy, Action, Thriller, and Adventure



- It appears that ratings are not normally distributed.
- The mean rating is 3.50 on a scale of 5.
- Almost half the movies have a rating of 4 and 5

Recommendation Engine

Recommendations based on Popularity to a New User

As the name suggests it recommends based on what is currently trending/ popular across the site. This is particularly useful when you don't have past data as a reference to recommend product to the user. It is not tailor fit for any particular group of audience or movie.

These are the most popular movies which can be recommended to a new user. Movies are sorted by both the number of ratings and their average rating in our dataset.

	title	mean_rating
277	Shawshank Redemption, The (1994)	4.429022
2226	Fight Club (1999)	4.272936
46	Usual Suspects, The (1995)	4.237745
224	Star Wars: Episode IV - A New Hope (1977)	4.231076
461	Schindler's List (1993)	4.225000
898	Star Wars: Episode V - The Empire Strikes Back...	4.215640
257	Pulp Fiction (1994)	4.197068
1939	Matrix, The (1999)	4.192446
314	Forrest Gump (1994)	4.164134
510	Silence of the Lambs, The (1991)	4.161290

Recommendations based on Genres to a New User

Content based recommendation system is certainly good if we want to recommend suggestions based on features like genres, actors, overview etc..

```
get_recommendations_based_on_genres('Pulp Fiction (1994)')
```

	title	genres
520	Fargo (1996)	Comedy Crime Drama Thriller
791	Freeway (1996)	Comedy Crime Drama Thriller
2453	Man Bites Dog (C'est arrivé près de chez vous)...	Comedy Crime Drama Thriller
3155	Beautiful Creatures (2000)	Comedy Crime Drama Thriller
4169	Confessions of a Dangerous Mind (2002)	Comedy Crime Drama Thriller
4523	Party Monster (2003)	Comedy Crime Drama Thriller
6676	In Bruges (2008)	Comedy Crime Drama Thriller
7129	Informant!, The (2009)	Comedy Crime Drama Thriller
7293	Leaves of Grass (2009)	Comedy Crime Drama Thriller
19	Money Train (1995)	Action Comedy Crime Drama Thriller

Some limitations of content based filtering are that it can only make recommendations based on existing interests of the user, it does not consider the fact that what do other users think of an item

Collaborative Filtering

Item-Based Collaborative Filtering

The main idea behind these methods is to use other users' preferences and taste to recommend new items to a user. The usual procedure is to find similar users (or items) to recommend new items which were liked by those users, and which presumably will also be liked by the user being recommended.

```
movie_similarity('Edward Scissorhands (1990)')
```

	title	genre	similarity	rated_movie
movieId				
1086	Hamlet (1996)	Crime Drama Romance	0.880000	25
236	Man of the House (1995)	Comedy	0.804984	45
2959	Billy Elliot (2000)	Drama	0.800000	218
5618	Dark Portals: The Chronicles of Vidocq (Vidocq...	Action Crime Fantasy	0.800000	87
4262	Bend It Like Beckham (2002)	Comedy Drama Romance	0.800000	67
2302	Dogma (1999)	Adventure Comedy Fantasy	0.800000	59
1276	Amistad (1997)	Drama Mystery	0.800000	57
2804	Butterfly (La lengua de las mariposas) (1999)	Drama	0.800000	55
1921	Wing Commander (1999)	Action Sci-Fi	0.800000	47
1562	Song of the South (1946)	Adventure Animation Children Musical	0.800000	42

Collaborative Filtering Model Based on User Ratings

In this section, I decided to apply Dimensionality Reduction technique to derive the tastes and preferences from the raw data, otherwise known as doing low-rank matrix factorization. Why reduce dimensions?

- I can discover hidden correlations / features in the raw data.
- I can remove redundant and noisy features that are not useful.
- I can interpret and visualize the data easier.
- I can also access easier data storage and processing.

	movieId	Expected Rating	title	genres	rated_movie
93	318	5.0	Shawshank Redemption, The (1994)	CrimeDrama	317
87	296	4.9	Pulp Fiction (1994)	ComedyCrimeDramaThriller	307
167	593	4.9	Silence of the Lambs, The (1991)	CrimeHorrorThriller	279
279	1196	4.9	Star Wars: Episode V - The Empire Strikes Back...	ActionAdventureSci-Fi	211
23	50	4.9	Usual Suspects, The (1995)	CrimeMysteryThriller	204
39	4226	4.9	Memento (2000)	MysteryThriller	159
280	1197	4.9	Princess Bride, The (1987)	ActionAdventureComedyFantasyRomance	142
278	1193	4.9	One Flew Over the Cuckoo's Nest (1975)	Drama	133
294	7361	4.9	Eternal Sunshine of the Spotless Mind (2004)	DramaRomanceSci-Fi	131
82	2329	4.9	American History X (1998)	CrimeDrama	129

These look like pretty good recommendations. It's good to see that, although I didn't actually use the genre of the movie as a feature, the truncated matrix factorization features "picked up" on the underlying tastes and preferences of the user. I've recommended some comedy, drama, and romance movies — all of which were genres of some of this user's top rated movies.

Conclusion and Future Consideration

In this notebook different recommendation approaches of content and collaborative filtering has been discussed.

First, I did exploratory data analysis then I started with content based filtering to recommend movie to new user based upon genre and movie popularity or the average ratings given by other users in the database.

I then progressed collaborative filtering based engines which try to find similar movies or users to make their predictions. After assessing models on RMSE metric, I found SVD++ to be the most accurate model but since SVD++ hyperparameters tuning time consuming I decided to go with SVD model tuned hyperparameters by using GridSearchCV

Finally, I made a recommendation engine which recommends 10 movies to specific user by using SVD model. And I added filtering options for genre and minimum number of ratings to make recommendations more accurate

There is a lot of potential to do but in the future, deep learning based recommender system can be built to enhance the performance and provide better recommendations to user.

For More Information

For any additional questions, please contact Yesim Cebeci at ysm.cebeci@gmail.com or <https://www.linkedin.com/in/yesim-cebeci/>

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

● Jupyter Notebook 100.0%