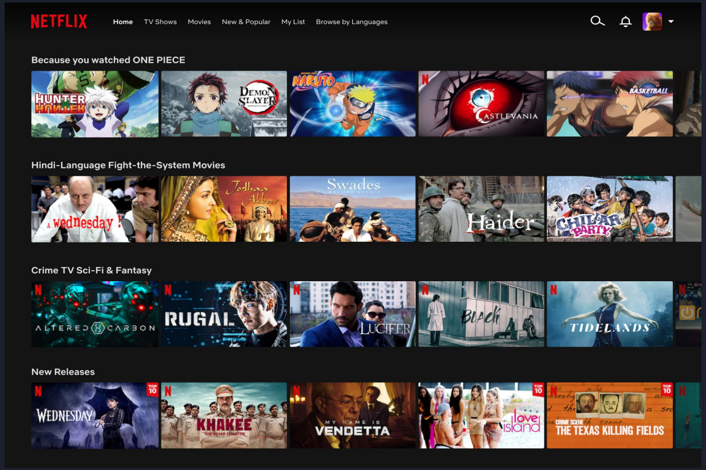
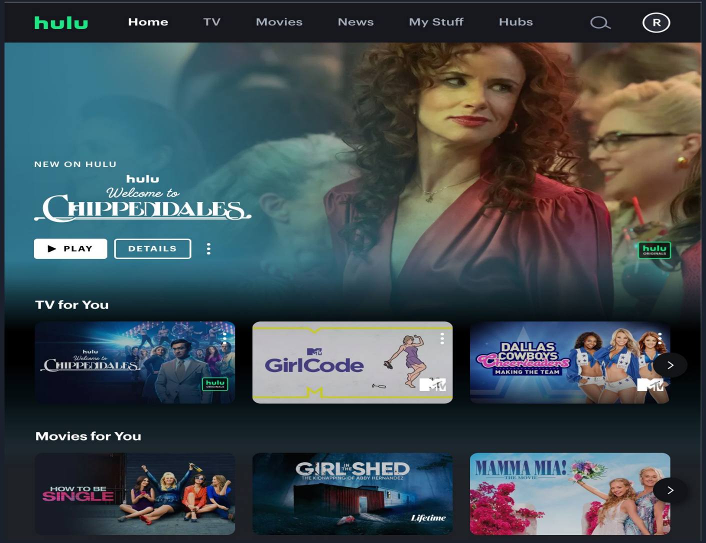




Hybrid Recommender System

By: Ruchina Shakya

Yingfan Wang

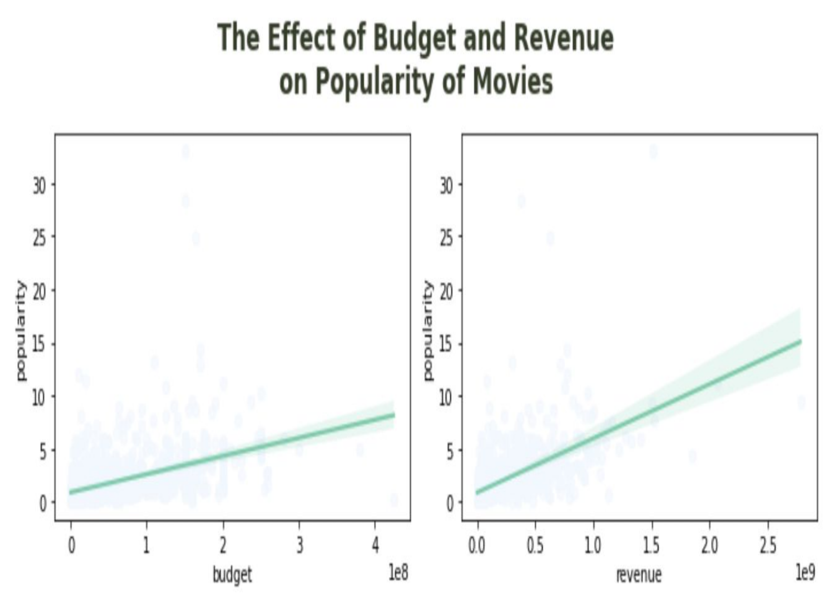
Preprocessing

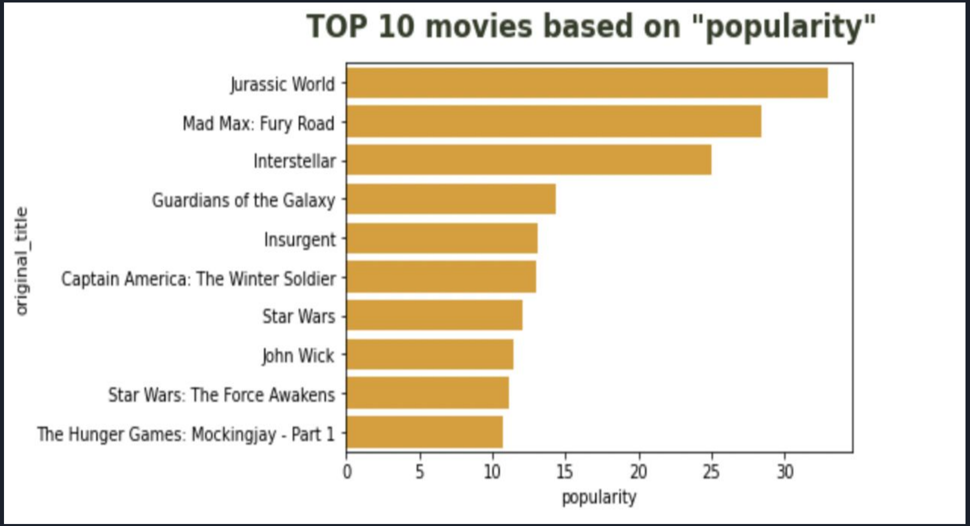
- Import dataset(We used MovieLens dataset and tmdb dataset)
- Remove the category we don’t need to simplify data
- Define function to remove symbols like “[” “{” from the title
- Removing movies with less watch count(<1500)

Vectorization

- Extract the movieId,title of movies and vectorize them
- Count Vectorization
- TF-IDF Vectorizer

Distribution of data





Compute cosine distance between vectores

Using cosine_similarity to compute cosine distance, so system can recommend similar movies to user. In our dataset, the top 5 most similar movies to ‘Terminator Genisys’ are:

```
#test
recommend('Terminator Genisys')
```

```
Back in Time
Narcopolis
Prince of Persia: The Sands of Time
Detention
Justice League: The Flashpoint Paradox
```

Hybrid similarity and weighted-average

To get a more accurate result, we also calculated weighted rating using formula as follows:

W=(R*v+C*m)/(v+m)

W: weighted-rating

R: average(mean) of the ratings of movie from number 1 to number 10

v: number of votes for the movie

m: minimum votes required to be listed in Top 250

C: mean vote across the whole report

To combine , we plus 40% of weighted rating score and 60% of similarity score, take summation as the final score.

Correlation and Weighted Average

- Finding users and an item similar to the randomly chosen user.
- Collaborative filtering(user-based and item-based)
- We created a correlation dataframe.
- The function `final_df.T.corr()`
- We selected users with correlation greater than **0.65.**
- **We used the weighted average approach.**
- `top_users_ratings["weighted_rating"] = top_users_ratings["correlation"] * top_users_ratings["rating"]`
- `agg({"weighted_rating": "mean"})`
- ***For item-based filtering, the random user which contains movie rating equal to 4.***
- ***We pick the movie names that the random user has rated 4.***
- ***We pick the first movie. Check the users which rated the same movie a 4.***
- ***Correlation :*** `user_movie_df.corrwith(movie_name)`

Sample data: user-users and item-user

	userId	correlation
19	2567.0	0.992065
18	63674.0	0.902522
17	45889.0	0.841233
16	34586.0	0.828798
15	59537.0	0.824942
14	55284.0	0.821542
13	24474.0	0.803947
12	110643.0	0.764303
11	115720.0	0.743210
10	103328.0	0.735279

User-based filtering

	movieId	title	genres
166	168	First Knight (1995)	Action|Drama|Romance

Item-based filtering

The Result: Top 10 movies recommendation

Movie recommendation for the random user using other users.

```
0                      Other, The (1972)
1                 Dawn of the Dead (2004)
2                       Ring, The (2002)
3           Mothman Prophecies, The (2002)
4     Once Upon a Time in America (1984)
Name: title, dtype: object
```

Movie recommendation for the random user using the item movie name.

```
title
First Knight (1995)
Jack (1996)
Kid in King Arthur's Court, A (1995)
Up Close and Personal (1996)
Highlander III: The Sorcerer (a.k.a. Highlander: The Final Dimension) (1994)
dtype: float64
```

Result: Using tmdb dataset and applying cosine similarity

Recommendation for the user number 342 using his rating data :

```
#test
predict_movie(342, 5)
```

```
Top 5 recommendations for user 342:

1. Lucky You
2. Escape from New York
3. Ghost Rider
4. The Prestige
5. Mission: Impossible
```

References

#https://#https://www.kaggle.com/code/mustafayurtcan/movielens-hybridrecommendersystem

#https://www.kaggle.com/code/mfaaris/hybrid-and-tensorflow-recommender-system

#https://www.kaggle.com/code/bhu1111/movies-recommendation-system