FINAL YEAR PROJECT

FIRST EVALUATION (SEM 8)

DEVAL MUDIA	BT16CSE060
AYUSH SINGH	BT16CSE098
CHAITYA CHHEDA	BT16CSE016
SADNEYA PUSALKAR	BT16CSE076

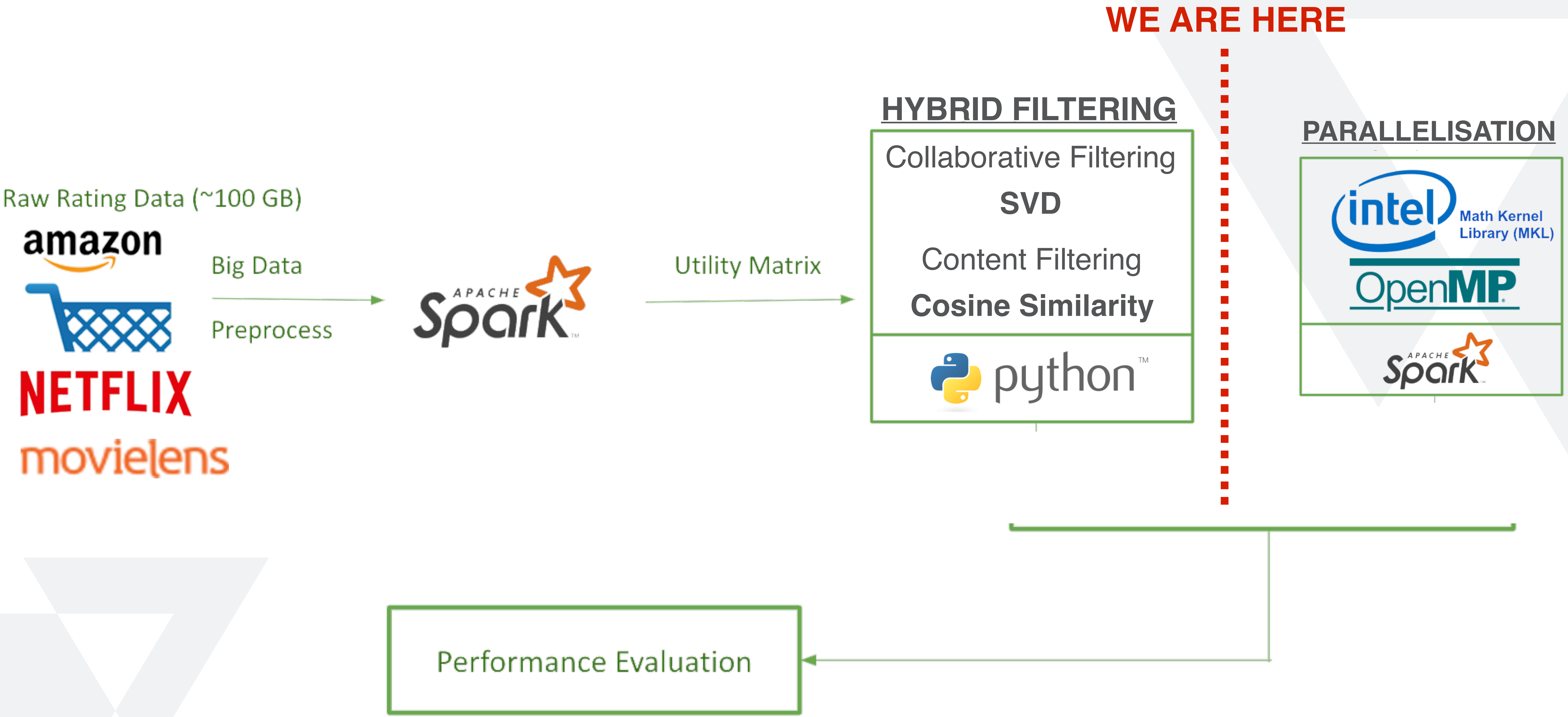
UNDER THE GUIDANCE OF
Dr. S.R. SATHE

PROBLEM STATEMENT

Parallelised Recommendation System for Amazon and Netflix

USING BIG DATA AND BIG COMPUTE

INFRASTRUCTURE



LAST EVALUATION

MovieLens

DATASET

Used MovieLens 1M movie ratings with 1 million ratings from 6000 users on 4000 movies.

Collaborative Filtering

FILTERING METHOD

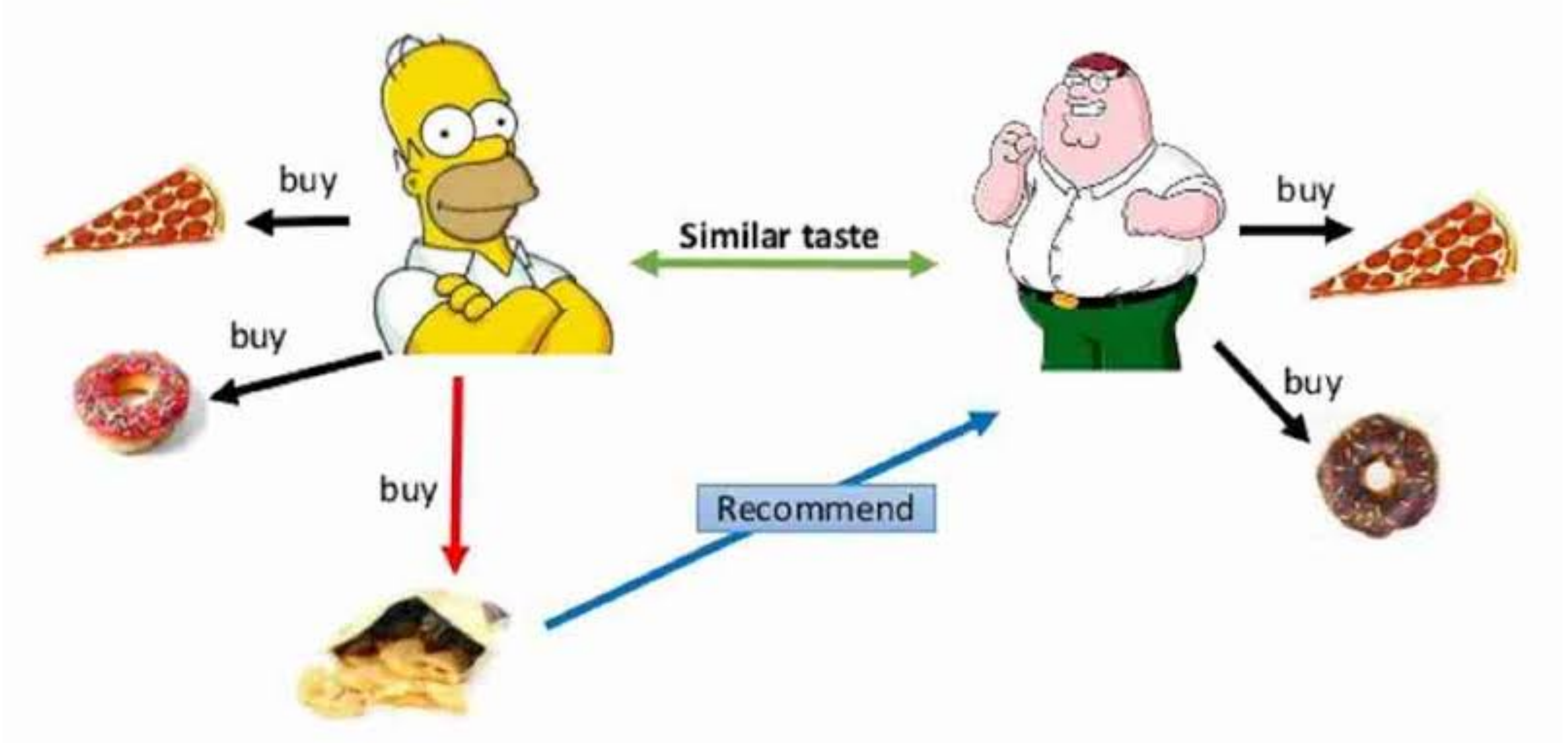
People like things that are liked by other people with **similar taste.**

Matrix Factorisation via SVD

RESULT

A method that can derive the tastes and preference vectors from the raw data to predict movies according to similarity between the users.

Collaborative Filtering



COLLABORATIVE FILTERING

PROBLEMS

Sparsity: If the movie is not popular or just released, then it will have few or no ratings all all.

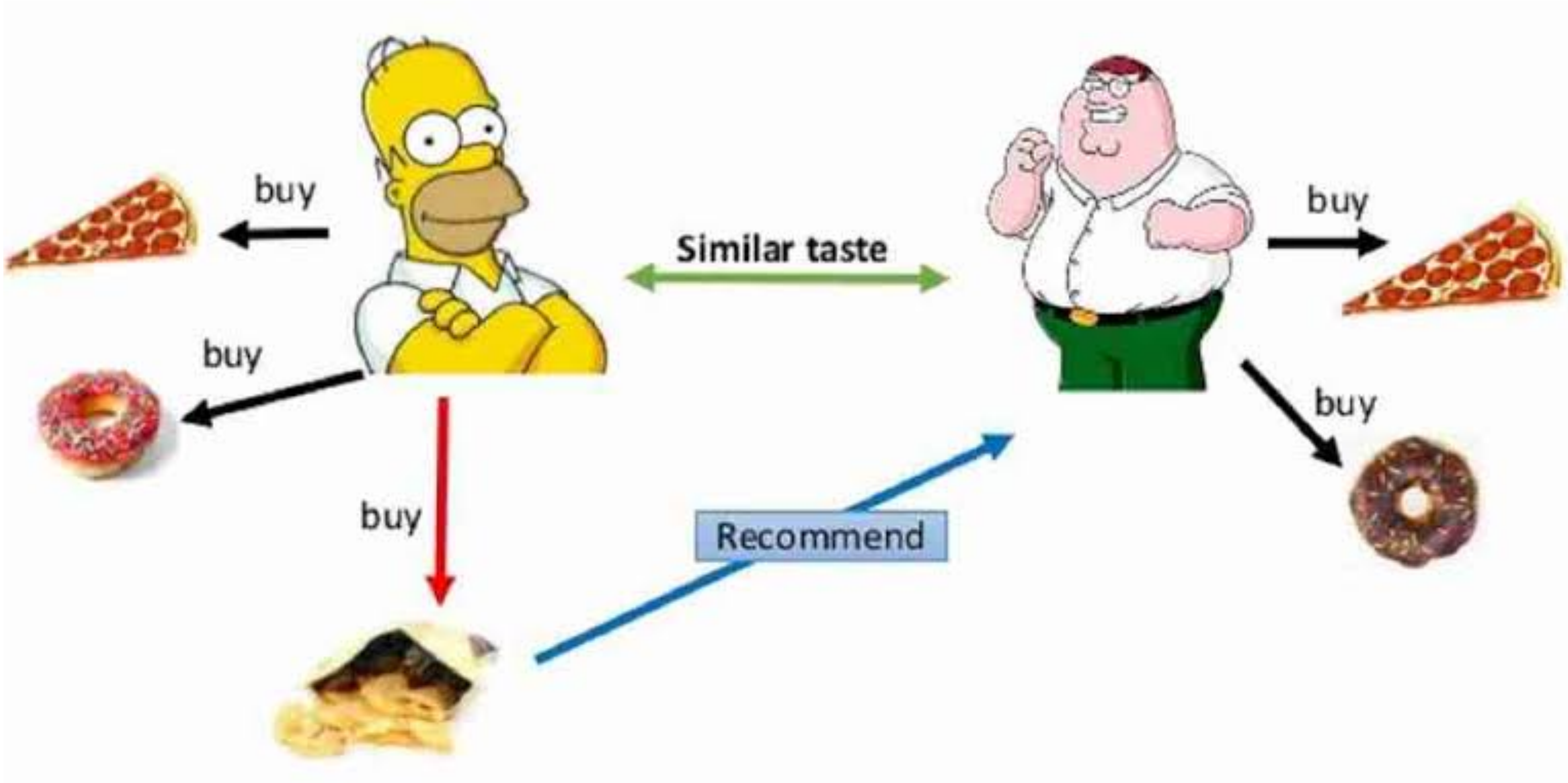
Cold Start Problem: If a user just joins the platform and hasn’t reviewed any movies, there is no way to create a taste of the user to recommend movies to him/her.

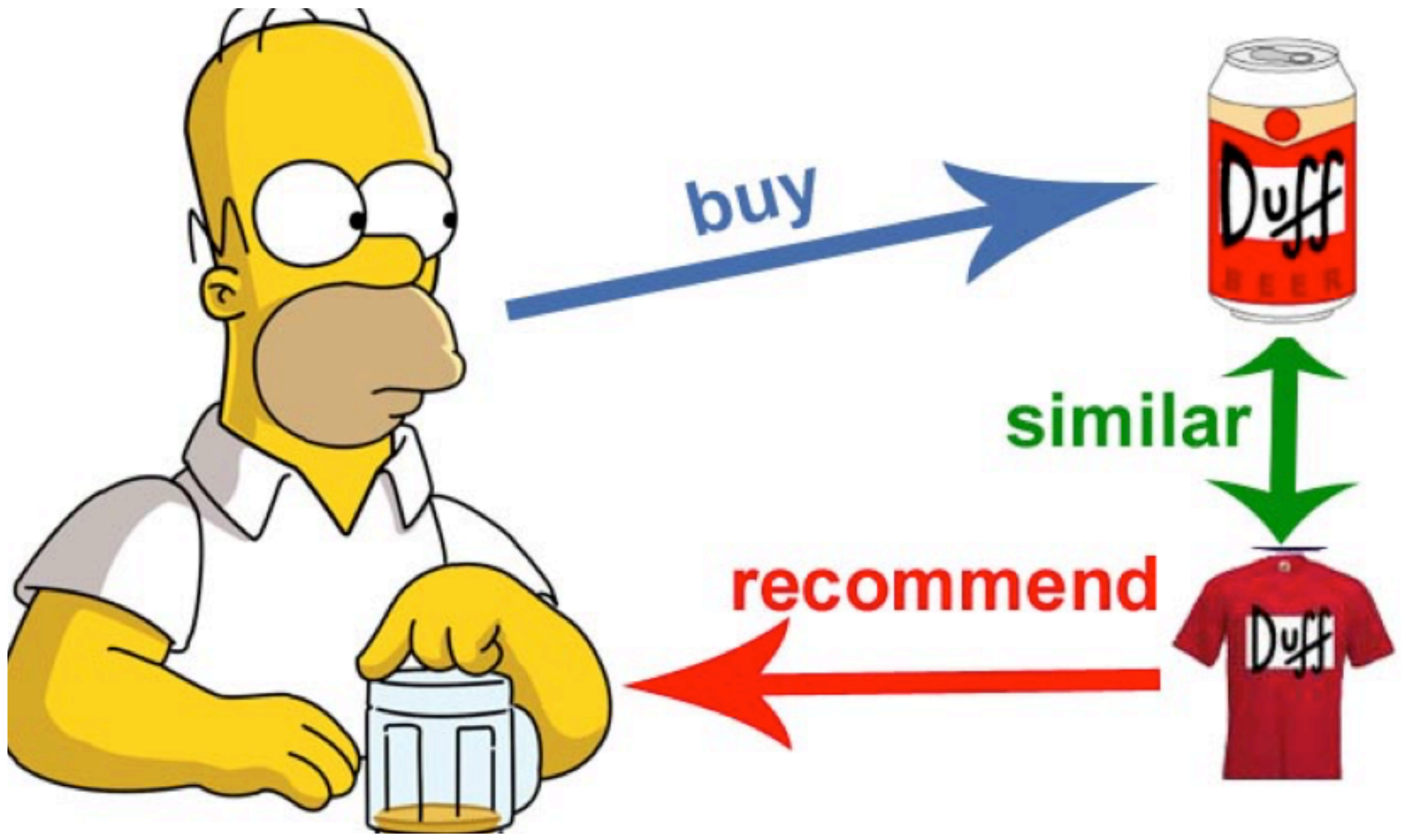
Popularity Bias Problem: The above method ends up recommending the most popular movies, which does not add an extra value to all the users.

SOLUTION

Hybrid Filtering

Collaborative Filtering + Content Filtering





Content Filtering

Compute pairwise similarity scores for all movies based on on the following metadata: **the 3 top actors, the director, related genres and the movie plot keywords** and recommend movies based on that similarity score.

	title	cast	director	keywords	genres
0	Toy Story	[Tom Hanks, Tim Allen, Don Rickles]	John Lasseter	[jealousy, toy, boy]	[Animation, Comedy, Family]
1	Jumanji	[Robin Williams, Jonathan Hyde, Kirsten Dunst]	Joe Johnston	[board game, disappearance, based on children'...	[Adventure, Fantasy, Family]
2	Grumpier Old Men	[Walter Matthau, Jack Lemmon, Ann-Margret]	Howard Deutch	[fishing, best friend, duringcreditsstinger]	[Romance, Comedy]

Used the **CountVectorizer()** instead of TF-IDF. This is because we do not want to down-weight the presence of an actor/director if he or she has acted or directed in relatively more movies.

Finally, using the cosine similarity to calculate a numeric quantity that denotes the similarity between two movies.

Used sklearn's **linear_kernel()**

```
get_recommendations('The Dark Knight Rises', cosine_sim2)
```

```
12589      The Dark Knight
10210        Batman Begins
9311                Shiner
9874      Amongst Friends
7772              Mitchell
516     Romeo Is Bleeding
11463         The Prestige
24090            Quicksand
25038             Deadfall
41063                 Sara
```

Hybridisation

The hybrid model builds a person's movie taste-profile using **collaborative filtering** and generates lists of similar movies to develop the recommendations using **content filtering.**

We combine the ratings from Content based filtering and Collaborative filtering to get more accurate results. It gives the predicted rating as weighted combination of the above described methods.

DEMO

Next Step

Parallelisation approach to reduce computation time of Collaborative Filtering and Content Filtering.

Movielens 1M	RMSE	MAE	Time
SVD	0.873	0.686	0:02:13
SVD++	0.862	0.673	2:54:19
NMF	0.916	0.724	0:02:31
Slope One	0.907	0.715	0:02:31
k-NN	0.923	0.727	0:05:27
Centered k-NN	0.929	0.738	0:05:43
k-NN Baseline	0.895	0.706	0:05:55
Co-Clustering	0.915	0.717	0:00:31
Baseline	0.909	0.719	0:00:19
Random	1.504	1.206	0:00:19

Next Step

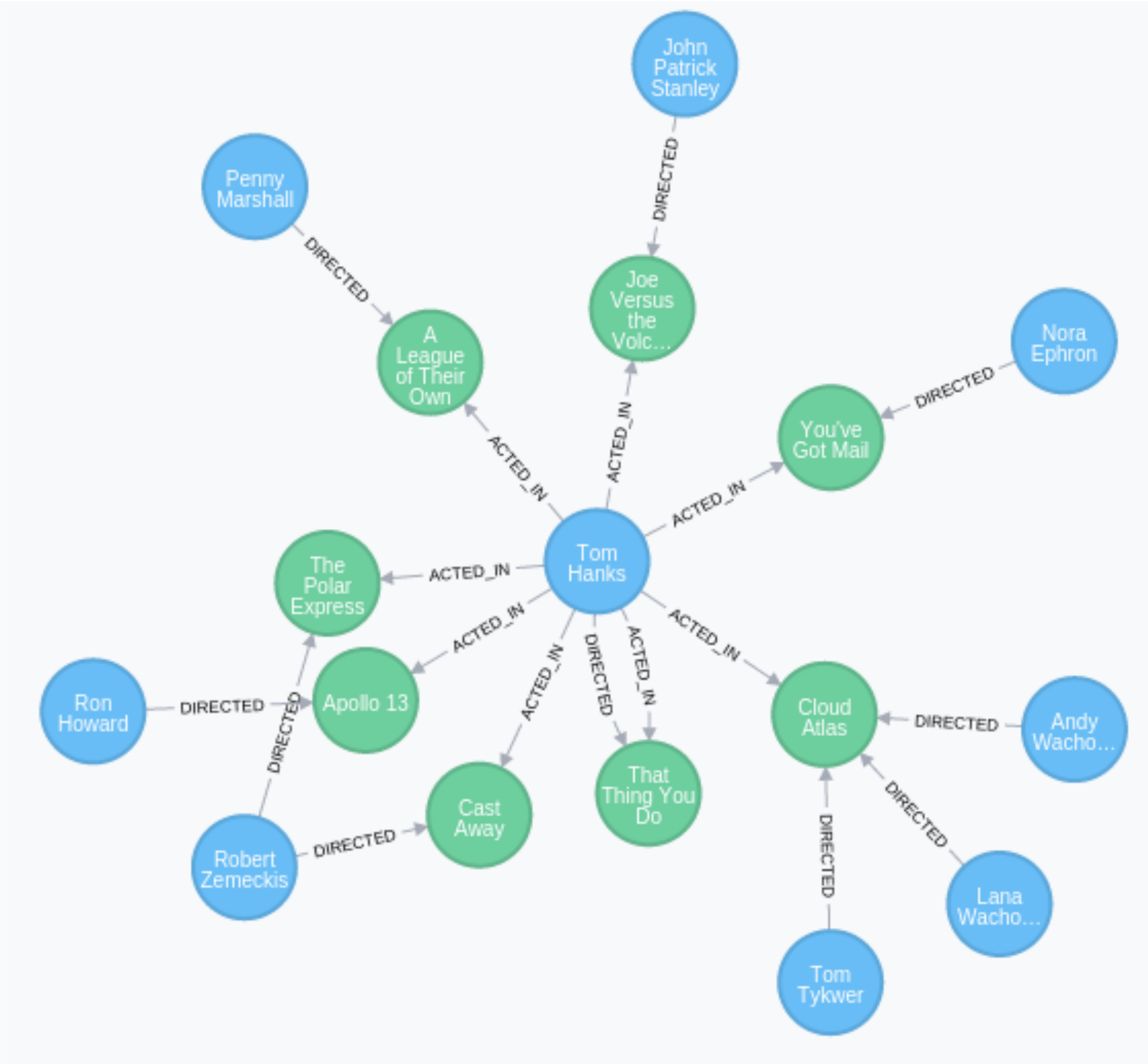
Graph Filtering

Created a network of all 45,000 films connected by the 500,000+ actors in the database.

Before creating a recommendation, chose a user's top three rated films. Then, once we have a set of recommendations from the Collaborative Filter and Content Filter (**CF_recs1**, **CF_recs2**), utilise the Graph Filter:

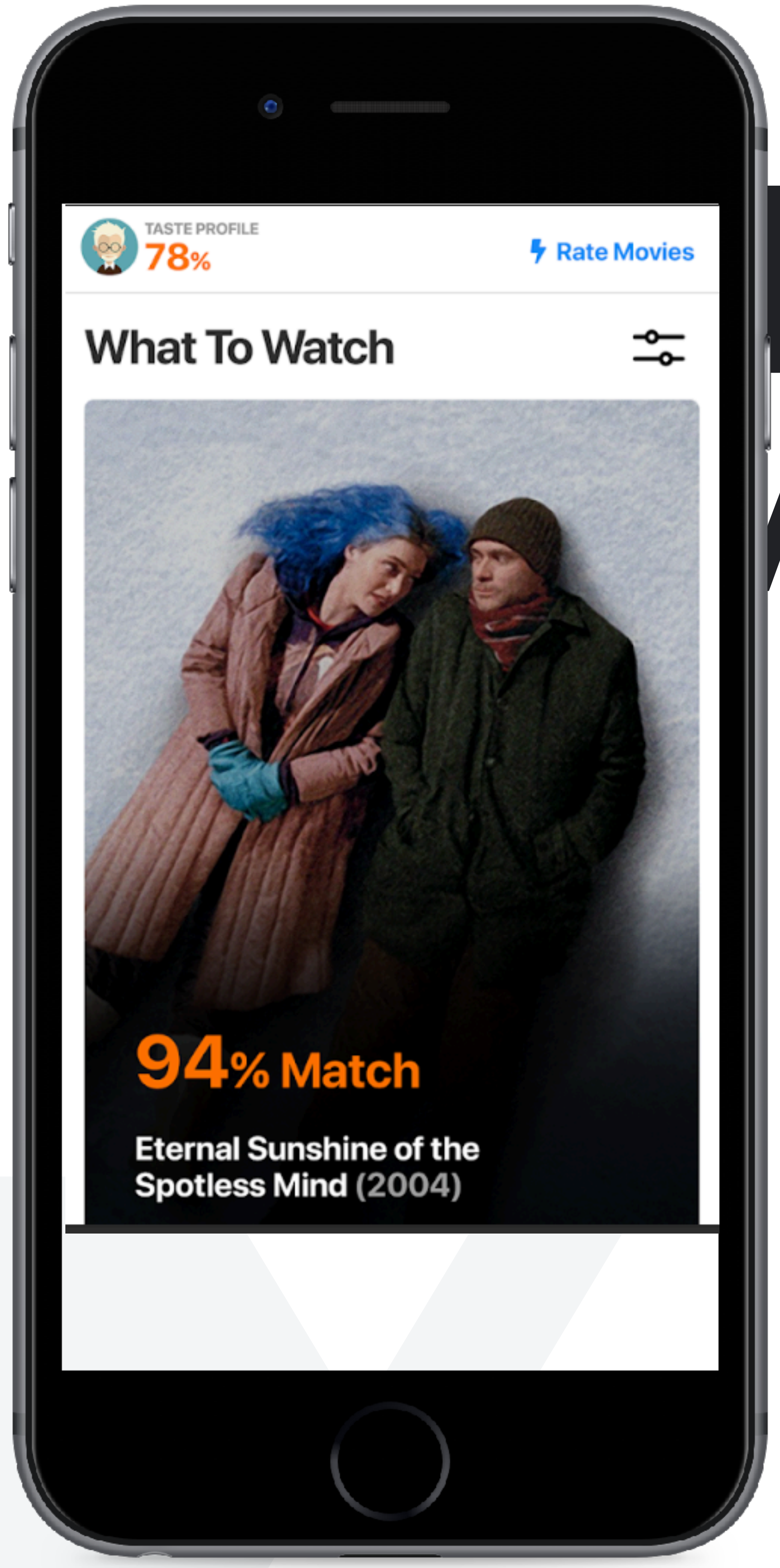
1. check to verify that CF_recs1 is connected on the graph to the top-3 films by a degree of two.
2. check to verify that CF_recs2 is connected on the graph to the top-3 films by a degree of two.
3. Films that are connected on the graph are recommended.

Implementation: Python neo4j API, and neo4j Cypher GraphQL


John Patrick Stanley
Penny Marshall
Joe Versus the Volc…
Nora Ephron
A League of Their Own
You've Got Mail
Tom Hanks
The Polar Express
Apollo 13
Ron Howard
Cast Away
That Thing You Do
Cloud Atlas
Andy Wacho…
Robert Zemeckis
Lana Wacho…
Tom Tykwer
DIRECTED
ACTED_IN

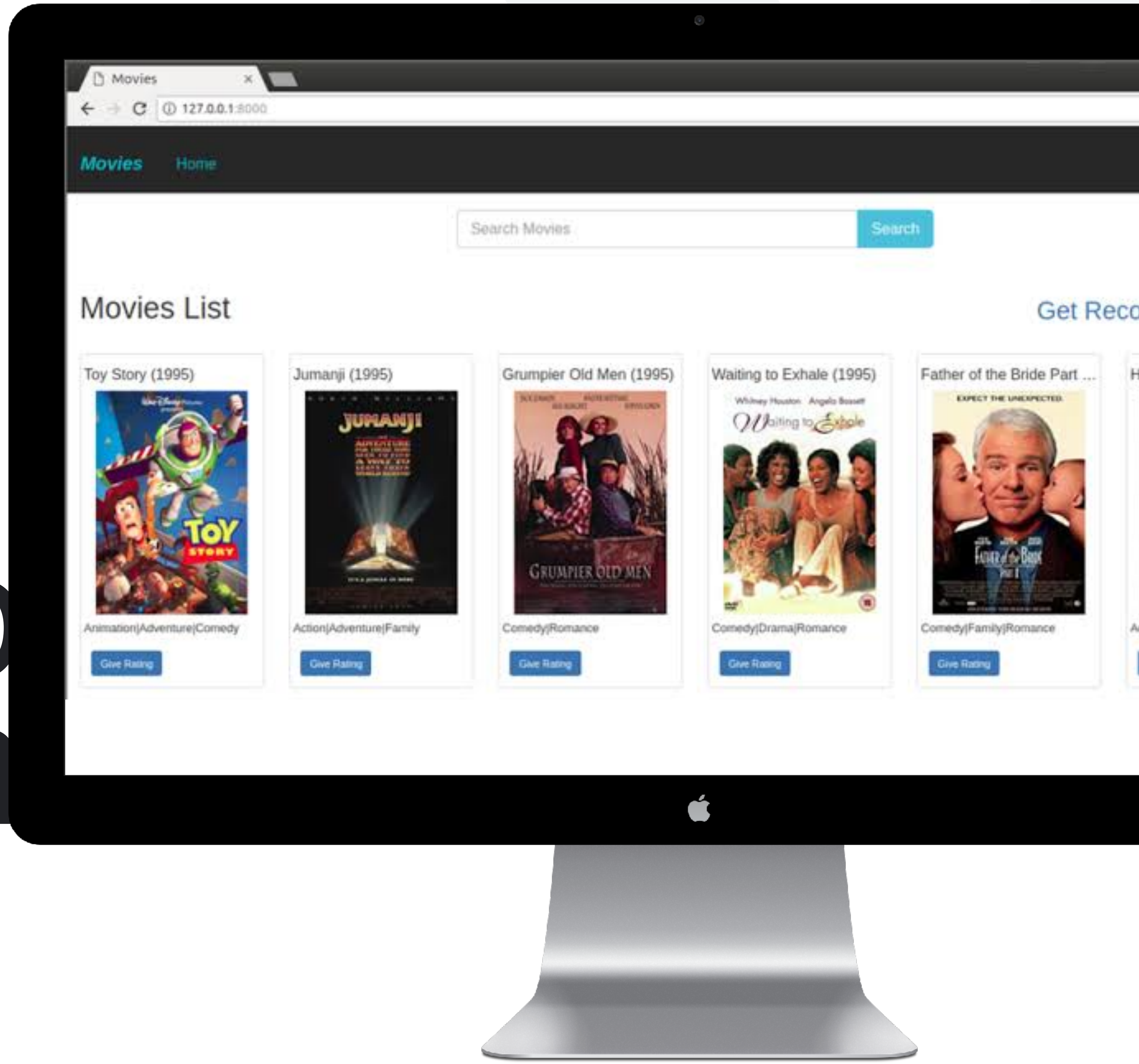
Next Step

Working Platform



Mobile App

Web Platform



References

Course : Recommender Systems (University of Minnesota)
https://www.coursera.org/specializations/recommender-systems#courses

https://www.datacamp.com/community/tutorials/recommender-systems-python

https://github.com/beckernick/matrix_factorization_recommenders

https://hackernoon.com/introduction-to-recommender-system-part-1-collaborative-filtering-singular-value-decomposition-44c9659c5e75