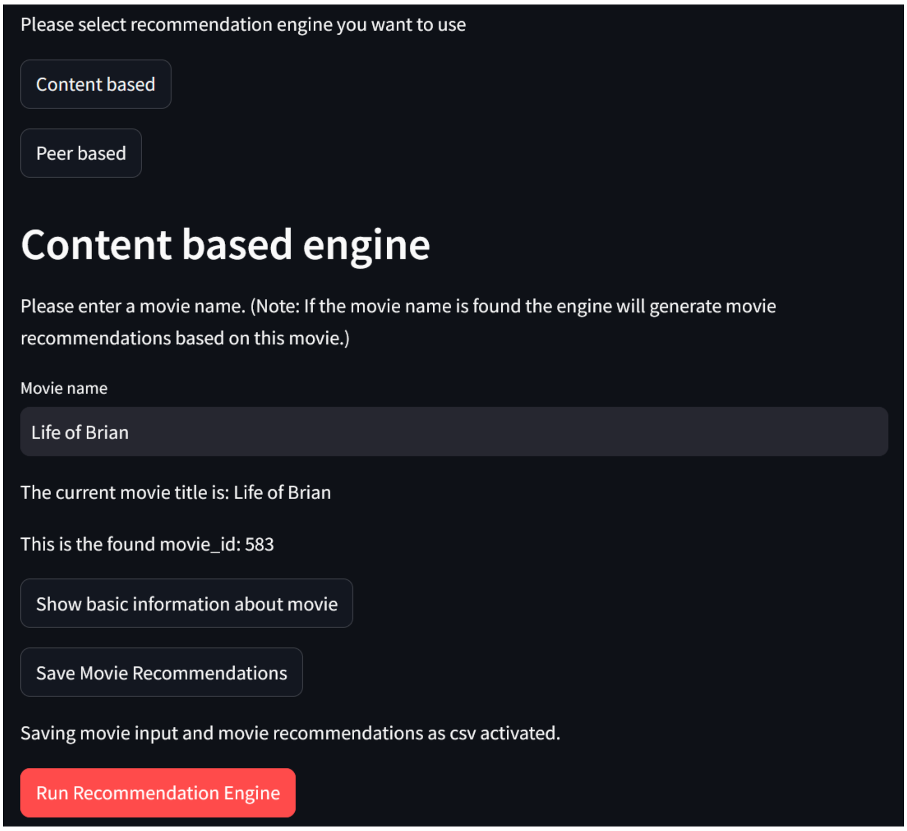
Movie Recommendation Engine
Report (First version)

Jens Kobler

March 21, 2025



Please select recommendation engine you want to use

Content based engine

Please enter a movie name. (Note: If the movie name is found the engine will generate movie recommendations based on this movie.)

Movie name: Life of Brian

The current movie title is: Life of Brian

This is the found movie_id: 583

Show basic information about movie
Save Movie Recommendations

Saving movie input and movie recommendations as csv activated.

Run Recommendation Engine

Contents

1 Idea of Project

The goal of this project is to program a movie recommendation engine. Content- and Peer-based recommendations are considered. Additionally a web-app is programmed to serve as a user interface. Streamlit was considered for this purpose. This project version lays the foundation for future development.

2 Dataset Information

- Name: The Movies Dataset
- Link: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset/
- 7 different csv files

Description of files:

- movies_metadata.csv: The main Movies Metadata file. Contains information on 45,000 movies featured in the Full MovieLens dataset. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies.
- keywords.csv: Contains the movie plot keywords for our MovieLens movies. Available in the form of a stringified JSON Object.
- credits.csv: Consists of Cast and Crew Information for all our movies. Available in the form of a stringified JSON Object.
- links.csv: The file that contains the TMDB and IMDB IDs of all the movies featured in the Full MovieLens dataset.
- links_small.csv: Contains the TMDB and IMDB IDs of a small subset of 9,000 movies of the Full Dataset.
- ratings.csv: The 100,000 ratings from 700 users.
- ratings_small.csv: The subset of 100,000 ratings from 700 users on 9,000 movies.

3 Data Exploration (DE)

- around 44.000 movies are in the database
  - movies only till 2013(?) are in the database
- around 26.000.000 User Ratings are in the database
- around 270.000 users rated movies.

4 Engine Concept

4.1 Content-based

A content-based recommendation engine, recommends movies based on the movie name the user wants to get recommendations for.

Since the movie dataset has genre labels for each movie. For example "Action", "Drama" and so on. Each movie is labeled with their respective genre. The idea to compare different movies for similarity is to, first, generate for each movie a binary vector based on the labels (decode the movies in the data base). Secondly, compute the cosine similarity between the movie input and the movies in the database. Select those movies which have the highest similarity and recommend them to the user.

With this method, the algorithm is able to narrow down the movies from 44000 to roughly 10 to 1000 movies. In some cases to much movies are recommended. This is why, the actor information is considered additionally. The idea here is to select only those movies with the same main actor. With this filter in place only a few movies (around 0 to 10) are recommended.

4.1.1 Implementation Details

- only the main actor is considered
  - one could also add all other actors (weighted with ordering of actors; program similar similarity measurments bewtween actors), idea: User can decide which filters he wants to use. For example: genre, actors, title, ... and combinations of them or all of them... and Priorities which filter first (use only the (by actor, title, genre, ...) narrowed down movie space).

4.2 Peer-based

A peer-based recommendation engine, recommends movies based on the user group the user belongs to.

As backbone the genre vector is considered. The user ratings are decoded in such a way that the result is a weighted and scaled genre vector. The genre vector is computed by multiplying the rating with the genre vector and summing all weighted rating vectors of one user up. This way, a single point in the feature space represents one user. This vector is min-max scaled to achieve values between 0 and 1 for each feature.

In the next step, those user genre vectors are clustered using the unsupervised machine learning algorithm KMeans. As a first k value k=20 is used.

For each cluster a weighted genre vector is again computed. The user clusters are represented by this weighted genre vector. With this weighted genre vector, movies are recommended using the method described in the content-based engine section. To not compute those movie recommendations everytime the engine is run, the solutions are saved in a database.

4.2.1 Choice of ML unsupervised algorithm

DBSCAN: does not work well.
KMeans: does work.
Not tried: Decision Tree Classifier. Sounds promising at the first glance.

4.2.2 Visualizing ML results

Visualizing cluster results: nothing can be seen.

4.2.3 Implementation Details

- only 1000 users were considered for the ml supervised clustering algorithm input (260.000 till 261.000), ordered by number of ratings

5 Web-App

5.1 Content-based recommendation engine

1. input moviename
2. engine runs
   - compute similarity with all 44.000 movies
   - select only movies with highest similarity
   - further movie filtering via main actor
3. output recommendations

5.2 Peer-based recommendation engine

App Design idea:

1. input moviename
2. input rating
3. save in user ratings database (loop 1.,2.,3.)
4. compute peer-based movie recommendations
   - compute from user ratings database, the weighted genre vector
   - classify user with ml algorithm
   - recommend movies based on the respective cluster

6 Flaws of version and Next Action Suggestions

Regarding the recommendation engine:

- the current peer-based engine version only serves as a proof of concept
  - to many movies are recommended for each cluster, more than 10
  - one could improve the quality of the user ratings
    - include more users
- improve quality of ml algorithm used
  - change hyperparameter (cluster number)
  - check quality of clustering
  - try Decision tree classifer? (right now only kmeans considered - works. DBSCAN does not work well.)
- include movie titles with bag of words functionality
- one could also include date of movie

Regarding the code:

- the movieIds do not have the same type. 3 different types are found (int, float, str).
  - clean those ids
- column names are not consistent (but it works)
- The program probably loads data several times.
- improve code quality
  - write comments to each function
  - clean the code
  - automaticly generate a code documentation (using sphinx?)

Regarding the Report:

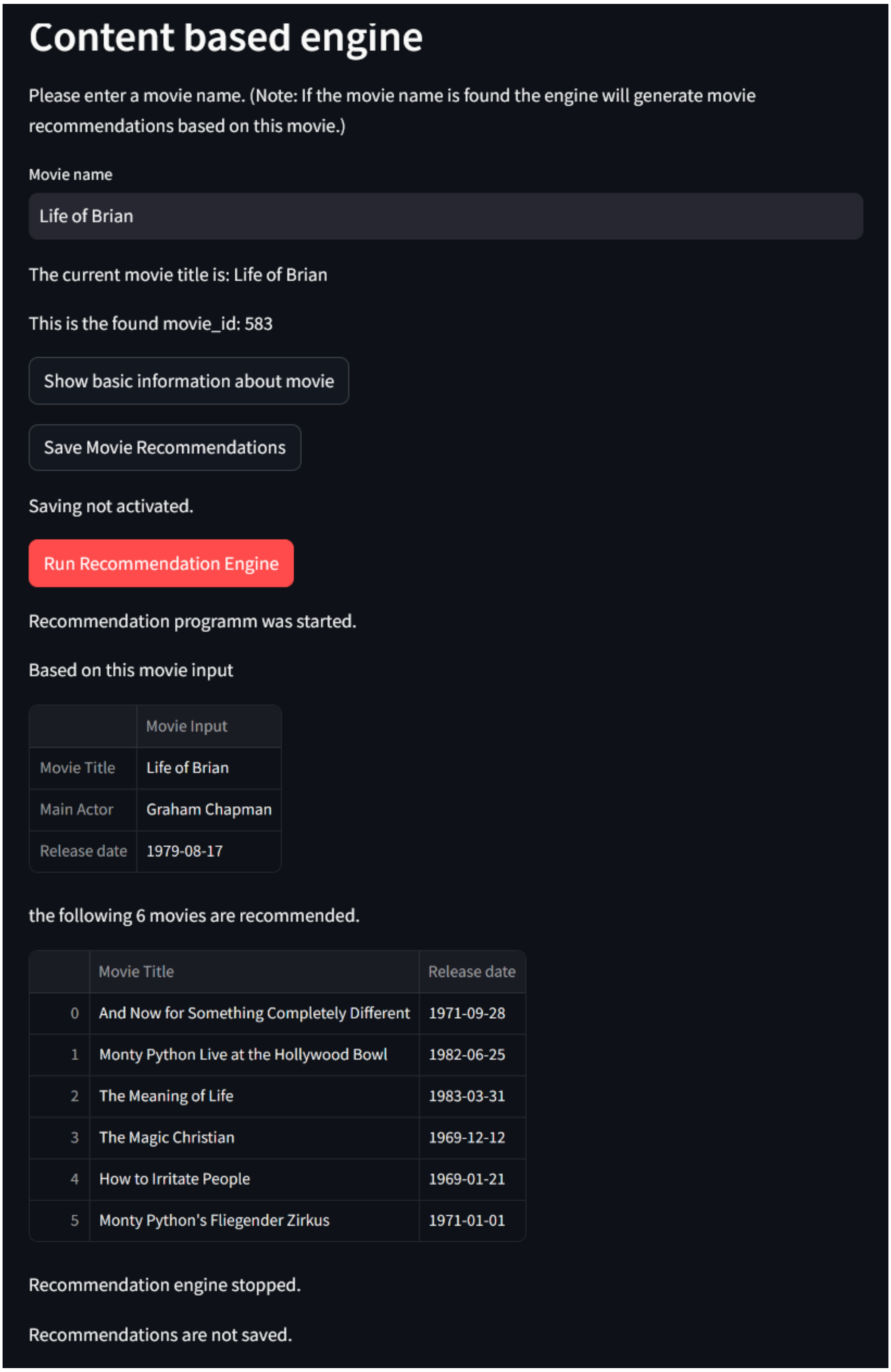
- improve grammer and spelling mistakes.
- increase depth of report
- increase visual appearance quality of report
- in general improve quality of report!

Next steps:

- Improve Peer Based - ml algorithm (optimize this algorithm)
  - inlcude more users in user rec data base
    - but first clean movie ids and filter all 26.000.000 user recommendations
    - by making a list of all correct movie ids
    - and selecting only those recommendations which belong to such an id
    - after that compute user ratings vectors including more users
    - choose number of k which results in different recommendations for each cluster.
    - increase k and check for similar recommendations between clusters. Stop at highest k without equal recommendations!.

A Examples

A.1 Content-based recommendation engine



Content based engine

Please enter a movie name. (Note: If the movie name is found the engine will generate movie recommendations based on this movie.)

Movie name: Life of Brian

The current movie title is: Life of Brian

This is the found movie_id: 583

Show basic information about movie
Save Movie Recommendations

Saving not activated.

Run Recommendation Engine

Recommendation programm was started.

Based on this movie input

	Movie Input
Movie Title	Life of Brian
Main Actor	Graham Chapman
Release date	1979-08-17

the following 6 movies are recommended.

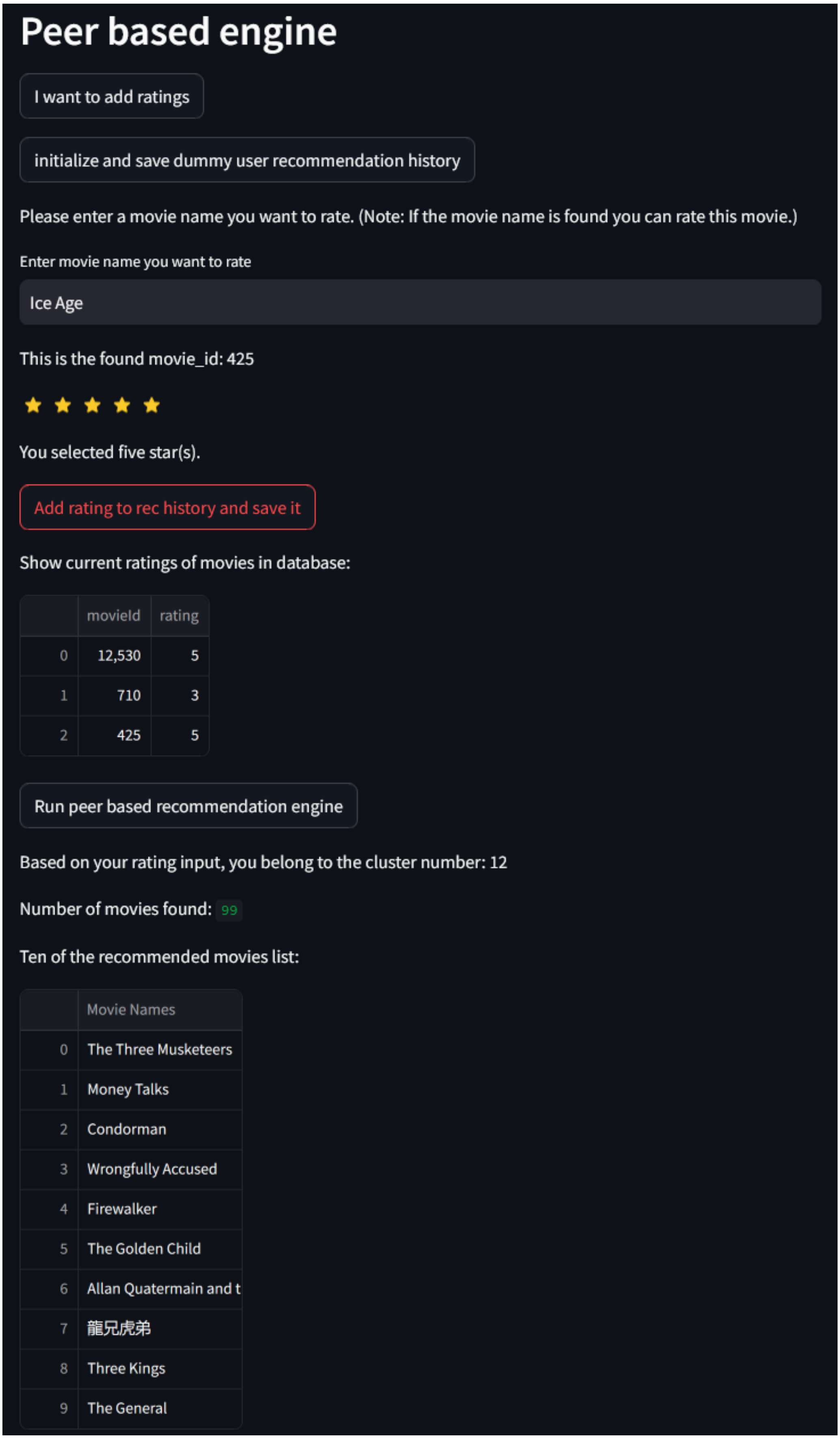
	Movie Title	Release date
0	And Now for Something Completely Different	1971-09-28
1	Monty Python Live at the Hollywood Bowl	1982-06-25
2	The Meaning of Life	1983-03-31
3	The Magic Christian	1969-12-12
4	How to Irritate People	1969-01-21
5	Monty Python's Fliegender Zirkus	1971-01-01

Recommendation engine stopped.

Recommendations are not saved.

Figure 1: Example input and output for content based engine.

A.2 Peer-based recommendation engine



Peer based engine
I want to add ratings
initialize and save dummy user recommendation history

Please enter a movie name you want to rate. (Note: If the movie name is found you can rate this movie.)

Enter movie name you want to rate

Ice Age

This is the found movie_id: 425

★ ★ ★ ★ ★

You selected five star(s).

Add rating to rec history and save it

Show current ratings of movies in database:

	movieId	rating
0	12,530	5
1	710	3
2	425	5

Run peer based recommendation engine

Based on your rating input, you belong to the cluster number: 12

Number of movies found: 99

Ten of the recommended movies list:

	Movie Names
0	The Three Musketeers
1	Money Talks
2	Condorman
3	Wrongfully Accused
4	Firewalker
5	The Golden Child
6	Allan Quatermain and t
7	龍兄虎弟
8	Three Kings
9	The General

Figure 2: Example input and output for peer based engine.