

Movie Recommender

System

By Group 9

Pranjal Bhardwaj 210741

Harshvardhan Pulipati 210792

Nitin Vedwal 200650

Nikita Singh 200638

|  |
| --- |
| cONTENTS |

* Introduction
* Dataset
* Methodology

1. Collaborative filtering methods
2. Content based filtering methods
3. Hybrid recommendation system

* Exploratory data analysis
* Model
  1. Score calculation
  2. Cosine similarity
  3. User-User comparison
  4. Item -Item comparison
* Model in action
* Conclusion

|  |
| --- |
| Introduction |

## *More choices, less happiness*

During the last few decades, with the rise of Youtube, Amazon, Netflix and many other such web services, recommender systems have taken more and more place in our lives. From e-commerce (suggest to buyers articles that could interest them) to online advertisement (suggest to users the right contents, matching their preferences), recommender systems are today unavoidable in our daily online journeys.

In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy or anything else depending on industries).

Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors. As a proof of the importance of recommender systems, we can mention that, a few years ago, Netflix organised a challenges (the “Netflix prize”) where the goal was to produce a recommender system that performs better than its own algorithm with a prize of 1 million dollars to win.

|  |
| --- |
| Dataset |

[The Movies Dataset](https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset/?select=ratings.csv)

The dataset can be found above.

## About Dataset

The dataset contains metadata on over 45,000 movies. 26 million ratings from over 270,000 users.

* The dataset consists of movies released on or before July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages.
* Ratings are on a scale of 1-5 and have been obtained from the official GroupLens website.

# Content

This dataset consists of the following 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\_small.csv:** The subset of 100,000 ratings from 700 users on 9,000 movies.

|  |
| --- |
| Methodology |

We are going to overview the two major paradigms of recommender systems : collaborative and content based methods.

**Collaborative filtering methods**

Collaborative methods for recommender systems are methods that are based solely on the past interactions recorded between users and items in order to produce new recommendations. These interactions are stored in the so-called “user-item interactions matrix”.

A screenshot of a black screen

Description automatically generated

Then, the main idea that rules collaborative methods is that these past user-item interactions are sufficient to detect similar users and/or similar items and make predictions based on these estimated proximities.  
  
The class of collaborative filtering algorithms is divided into two sub-categories that are generally called memory based and model based approaches. Memory based approaches directly works with values of recorded interactions, assuming no model, and are essentially based on nearest neighbours search (for example, find the closest users from a user of interest and suggest the most popular items among these neighbours). Model based approaches assume an underlying “generative” model that explains the user-item interactions and try to discover it in order to make new predictions.

A screenshot of a computer

Description automatically generated

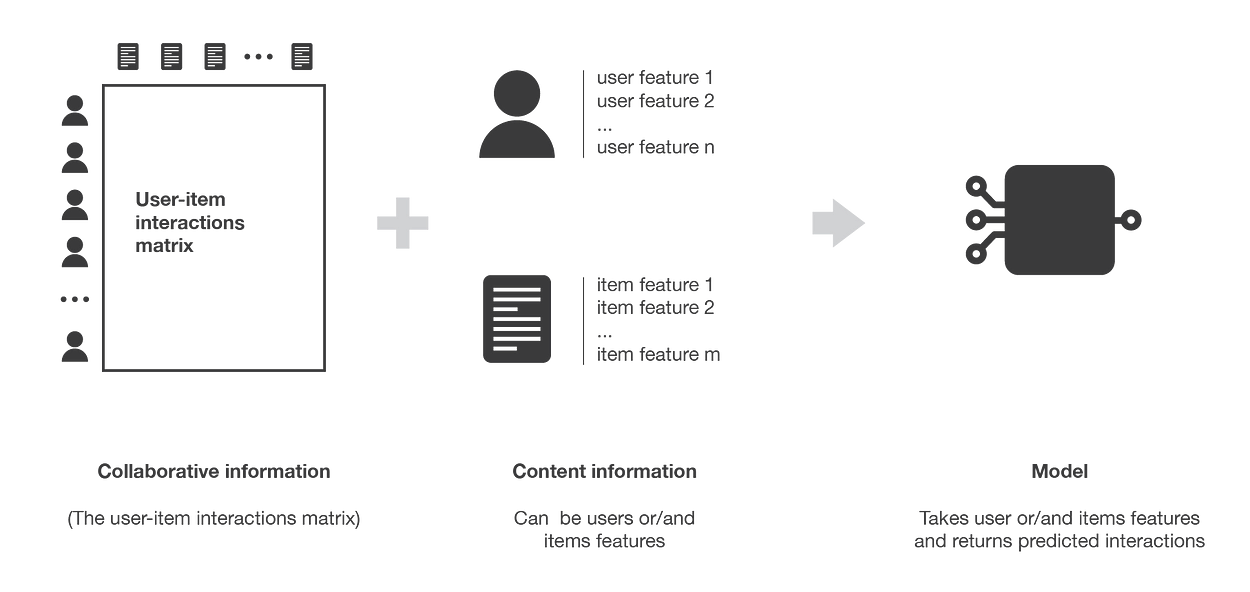
The main advantage of collaborative approaches is that they require no information about users or items and, so, they can be used in many situations. Moreover, the more users interact with items the more new recommendations become accurate: for a fixed set of users and items, new interactions recorded over time bring new information and make the system more and more effective.

However, as it only consider past interactions to make recommendations, collaborative filtering suffer from the “cold start problem”: it is impossible to recommend anything to new users or to recommend a new item to any users and many users or items have too few interactions to be efficiently handled. This drawback can be addressed in different way: recommending random items to new users or new items to random users (random strategy), recommending popular items to new users or new items to most active users.

## Content based methods

Unlike collaborative methods that only rely on the user-item interactions, content based approaches use additional information about users and/or items. If we consider the example of a movies recommender system, this additional information can be, for example, the age, the sex, the job or any other personal information for users as well as the category, the main actors, the duration or other characteristics for the movies (items).

Then, the idea of content based methods is to try to build a model, based on the available “features”, that explain the observed user-item interactions. Still considering users and movies, we will try, for example, to model the fact that young women tend to rate better some movies, that young men tend to rate better some other movies and so on. If we manage to get such model, then, making new predictions for a user is pretty easy: we just need to look at the profile (age, sex, …) of this user and, based on this information, to determine relevant movies to suggest.



Content based methods suffer far less from the cold start problem than collaborative approaches: new users or items can be described by their characteristics (content) and so relevant suggestions can be done for these new entities. Only new users or items with previously unseen features will logically suffer from this drawback, but once the system old enough, this has few to no chance to happen.

**Hybrid Recommendation System**

A recommendation engine that uses hybrid filtering utilizes both collaborative and content-based data. Not surprisingly, its superior filtering produces the best information.

Netflix is a good example of a company using hybrid recommender filtering on its website. It accounts for users’ interests (collaborative filtering) and also for movie descriptions and features (content-based filtering).

A black background with a black square

Description automatically generated with medium confidence

Summary of the different types of recommender systems algorithms

We initially introduced a movie recommendation approach that relies on user ratings and the movies' popularity among them. Following that, we developed a hybrid recommender system that considers both user preferences and ratings. This system takes into account the movies a user has rated highly and the ratings provided by similar users for movies not yet rated by the original user. Using this information, the model predicts the likely rating the user would assign to a movie and suggests the top 'k' movies accordingly. The value of 'k' can range from 1 to 10.

|  |
| --- |
| Exploratory data analysis |

First thing first, there's always an EDA to give us a sense of what data we are dealing with. It's also useful to acquire some insights, information’s, and even mistakes from data.

## **Insights**

**A screen shot of a chart

Description automatically generated**

There is a huge difference between those 2 types of movies in terms of presence for this particular dataset.

A comparison of a graph

Description automatically generated with medium confidence

**Budget and Revenue just slighly influence the popularity of the movies.**

**A graph with a line and a line

Description automatically generated with medium confidence**

Most of the movies lay on top of the yellow line, indicate that those movies make a profit

A close up of words

Description automatically generated

Word "life", "one", "find", "love" apparently appear in many occasions.

A graph of a movie genre

Description automatically generated

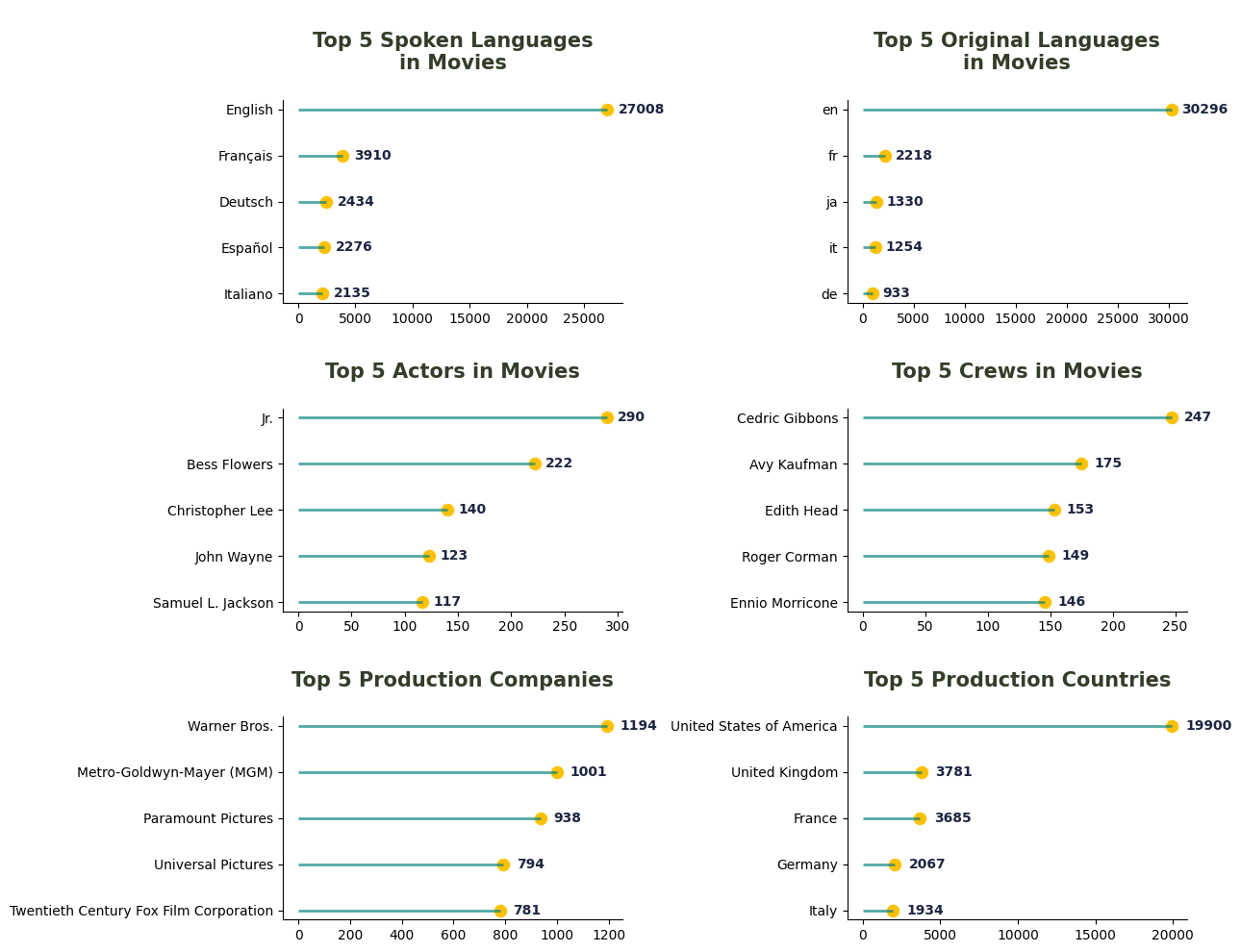
Drama is the most dominant genre with over 18000 movies.

A graph showing a number of movies

Description automatically generated

Started from 1930, movies industry had grown significantly from 50 years ago

A drop in total released movies around 2020 is because of the Covid outbreak impacting the industry

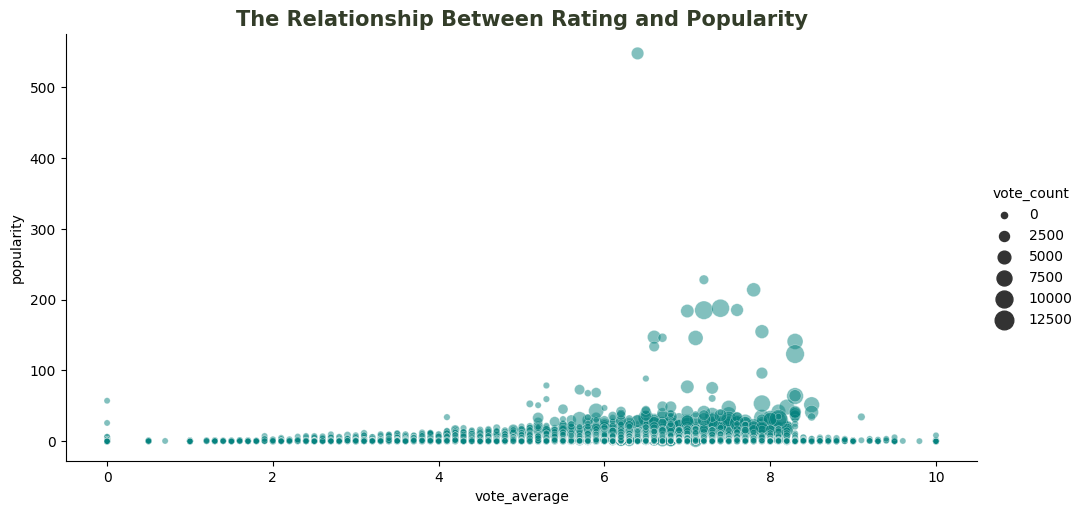


For this particular dataset, english is on top of the list for the original and spoken language in the movies.

Jr. and Cedric Gibbons are actor and crew involved in the most movies in the list respectively.

Warner Bros. with 1194 movies make it become top 1 production company in the list.

Many great production companies come from USA. So, it's not a surprise if USA is become our top 1 for production country.



Movies that either got rating 0 or 10 are basically caused by small number of voter. As the vote count increase, the rating is most likely around 5 to 8.5.

It's clear that popular movies will get more vote count as shown from above plot.

A graph of data distribution

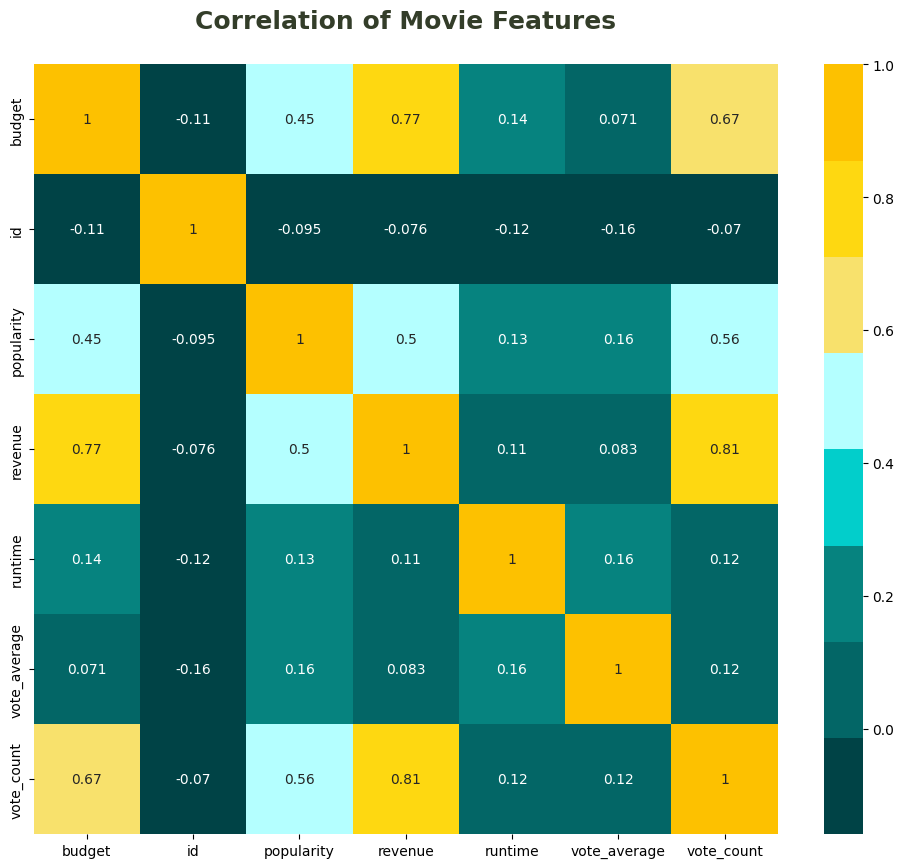
Description automatically generated with medium confidence

The movie genre that has the longest runtime is drama.

The least popular genre in the top 5 is romance.

Action movies spent more money than the rest of the movies.

One of the action movies got a vast profit compared to the others.

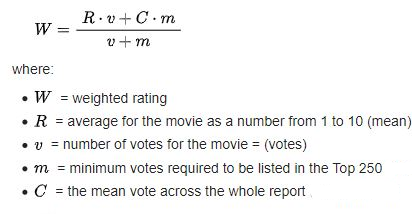


Vote count, budget, and popularity are 3 dominant features that will determined the revenue of the movies

This was the EDA of the data now we shall proceed to towards the preprocessing and Model making .

|  |
| --- |
| Model |

We initially make a score parameter to find out of the movies based upon rating given to the movie and overall mean rating based on the formula given below.



People watch a movie not just because they see a good rating for that movie, but also because of the hype of certain movie. So, in this case, put popularity into a consideration is a wise choice.

Let's take 40% weight for weighted average and 60% weight for popularity considering people don't want to miss a hype movie even the reviews and ratings are poor, to find the score. We can play around with the number.

Now we create a bag of words from the movie’s genre overview product\_companies , tagline , keywords, crew, characters,actors. Now A common method to find similarity between 2 movies is a method called cosine similarity.

Using cosine similarity we will predict movies similar to a given movie. Cosine similarity will be applied on the bag of words and other features.

Now we train a TensorFlow recommender model. Tensorflow comes with a library called TensorFlow Recommenders (TFRS) for building a recommender system. It's built on Keras and aims to have a gentle learning curve while still giving you the flexibility to build complex models.

This time, we use multi-objective approach that applies both implicit (movie watches) and explicit signals (ratings). In the end, we can predict what movies should the user watch along with the given rating corresponds to historical data.

**User-user**

In order to make a new recommendation to a user, user-user method roughly tries to identify users with the most similar “interactions profile” (nearest neighbours) in order to suggest items that are the most popular among these neighbours (and that are “new” to our user). This method is said to be “user-centred” as it represent users based on their interactions with items and evaluate distances between users.

Assume that we want to make a recommendation for a given user. First, every user can be represented by its vector of interactions with the different items (“its line” in the interaction matrix). Then, we can compute some kind of “similarity” between our user of interest and every other users. That similarity measure is such that two users with similar interactions on the same items should be considered as being close. Once similarities to every users have been computed, we can keep the k-nearest-neighbours to our user and then suggest the most popular items among them (only looking at the items that our reference user has not interacted with yet).

Notice that, when computing similarity between users, the number of “common interactions” (how much items have already been considered by both users?) should be considered carefully! Indeed, most of the time, we want to avoid that someone that only have one interaction in common with our reference user could have a 100% match and be considered as being “closer” than someone having 100 common interactions and agreeing “only” on 98% of them. So, we consider that two users are similar if they have interacted with a lot of common items in the same way (similar rating, similar time hovering…).

A screenshot of a video game

Description automatically generated

**Item-item**

To make a new recommendation to a user, the idea of item-item method is to find items similar to the ones the user already “positively” interacted with. Two items are considered to be similar if most of the users that have interacted with both of them did it in a similar way. This method is said to be “item-centred” as it represent items based on interactions users had with them and evaluate distances between those items.

Assume that we want to make a recommendation for a given user. First, we consider the item this user liked the most and represent it (as all the other items) by its vector of interaction with every users (“its column” in the interaction matrix). Then, we can compute similarities between the “best item” and all the other items. Once the similarities have been computed, we can then keep the k-nearest-neighbours to the selected “best item” that are new to our user of interest and recommend these items.

Notice that in order to get more relevant recommendations, we can do this job for more than only the user’s favourite item and consider the n preferred items instead. In this case, we can recommend items that are close to several of these preferred items.

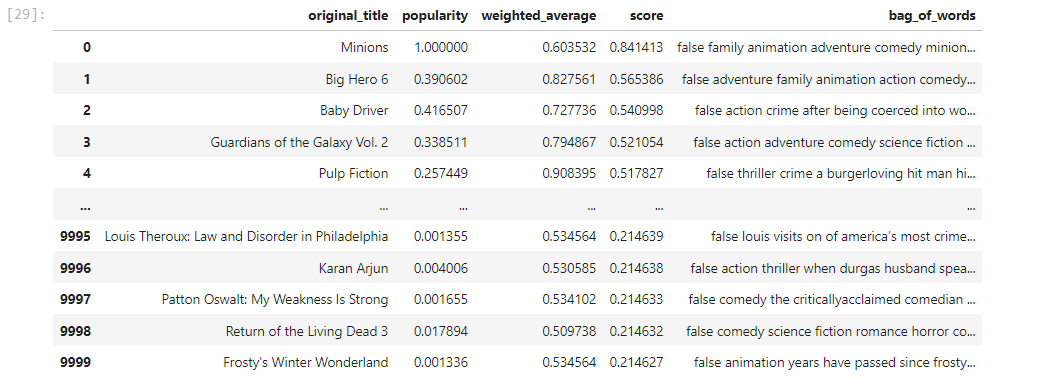
A screen shot of a black background

Description automatically generated

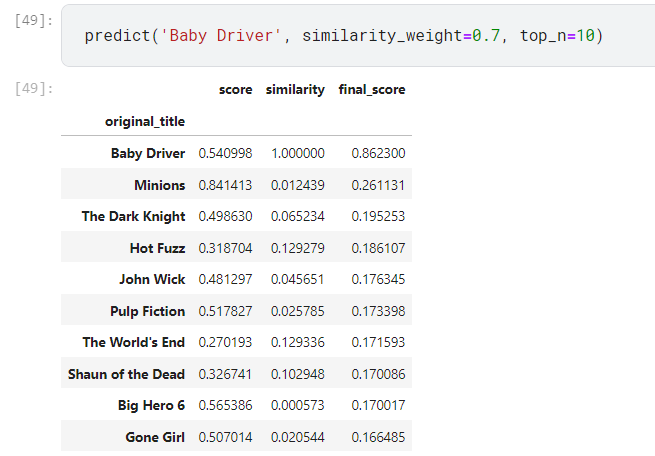
**Evaluation of a recommender system**

As for any machine learning algorithm, we need to be able to evaluate the performances of our recommender systems in order to decide which algorithm fit the best our situation. Evaluation methods for recommender systems can mainly be divided in two sets: evaluation based on well defined metrics (Root Mean square error of ratings) and evaluation mainly based on human judgment and satisfaction estimation.

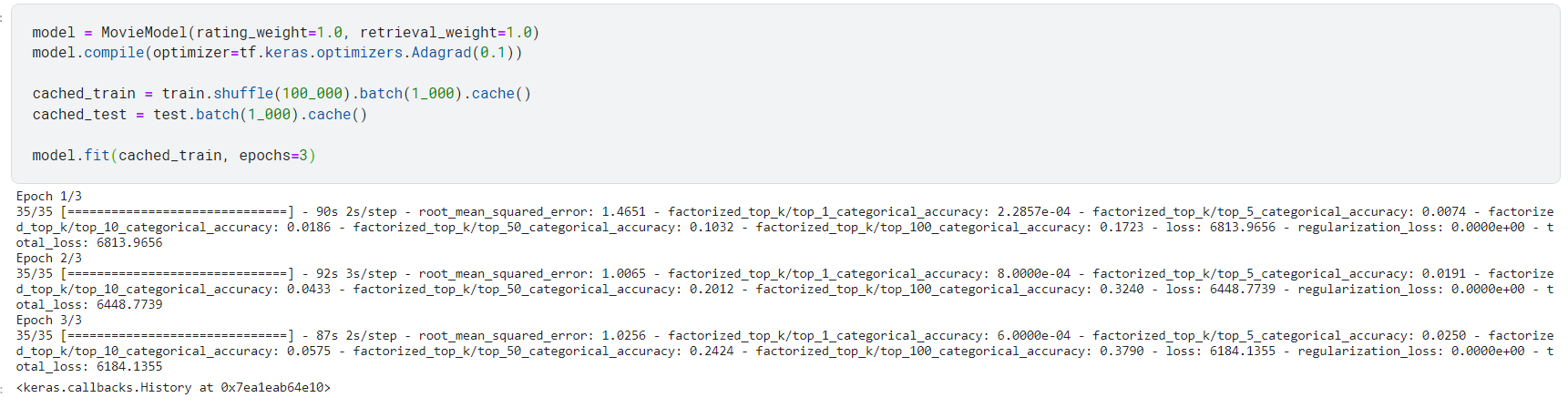
# Model in action



Bag of words of the movie features



Prediction of top 10 similar movies to 'Baby Driver'



Training the model

A screenshot of a computer code

Description automatically generated

A screenshot of a computer screen

Description automatically generated

Prediction of user 123 , his top 5 movies that he will like according to his rating to other movies

A screenshot of a computer

Description automatically generated

Prediction of rating of Minions of User 123

If we examine the rating given by the User 123

A screenshot of a computer

Description automatically generated

If we examine the movies that is recommended as per model

A screenshot of a computer

Description automatically generated

At a glance, we can see if **User 123** love watching Drama movies most of the time. He/She also gives a good rating for that genre. In our recommendation, We give 5 more Drama movies that we expect him/her to love the movies in a similar way with the previous watched movies.

In our dataset, we don't see any Animation movies that have been watched by **User 123**. So, it's not a surprise if the estimated rating for Minions is quite low.

# Conclusion

In this project, we developed a movie recommender system using a hybrid approach that combines collaborative filtering and content-based filtering. The system was able to generate personalized recommendations for users based on their past ratings and the ratings of similar users, as well as the movies' genres, keywords, and other features.

We evaluated the performance of our system using the root mean square error (RMSE) metric. The RMSE score for our system was 0.9.

Overall, our movie recommender system is a feasible and effective way to generate personalized recommendations for users. The system is able to take into account both the users' past preferences and the movies' features to generate recommendations that are likely to be of interest to each user.

Here are some specific findings from our experiments:

* Collaborative filtering was more effective than content-based filtering for predicting users' ratings of movies they had not yet rated. This suggests that users' past ratings of similar movies are a good predictor of their ratings of new movies.
* The hybrid approach that combines collaborative filtering and content-based filtering outperformed both individual approaches. This suggests that the two approaches complement each other and can be used to generate more accurate recommendations.
* The system was able to generate personalized recommendations for users with different preferences. For example, the system was able to recommend different movies to users who preferred different genres.

Personalized recommendations have become an integral part of numerous online platforms and services, such as Amazon.com, Netflix, and Pandora, enhancing user experiences and driving engagement. As we journey further into this digital age, recommender systems will continue to evolve, ushering in a future where the burden of choice is lightened, and our capacity to make informed decisions remains intact. In this ever-expanding landscape of choice, recommender systems are the compass that keeps us moving forward.