# Project Report

## Movie Recommendation System

Prepared By: **Team Outliers**

Harika Nalam

Rajvi Shah

Sharad Nataraj

Shreya Nimbhorkar

**Abstract**:

With OTT platforms being the new way of watching content, a lot of content is hosted on a single streaming service. The user would have to search for specific content among a large collection of movies. Because of its ability to provide improved entertainment, a movie recommendation is vital in our social lives. Users can be recommended a set of movies based on their interests or the popularity of the films.

To solve this problem, a recommendation system can be used based on the user profile, recent views and the content of the movie, we will be able to recommend movies that are more towards the interest of the user. The system will work based on a model trained using already existing data

A recommendation system is used to make suggestions for things to buy or see. They employ a big collection of information to steer consumers to the things that will best match their needs. A recommender system, also known as a recommendation engine or platform, is a type of information filtering system that attempts to anticipate a user's "rating" or "preference" for an item. They're mostly employed for commercial purposes. It also assists users in efficiently and effectively locating movies of their choosing based on the movie experiences of other users, without wasting time in pointless searching.

## **Introduction** :

When it comes to streaming show websites, understanding user behaviour is always a challenge. Everyone loves watching movies at home, regardless of gender, age, or geographical region. Many people prefer romance, action, or comedy films, while others enjoy the visions of the leading performers and filmmakers. Through this beautiful medium, we are all connected. What's most intriguing, though, is how unique our combinations and choices are in terms of the show's tastes. However, given everything that's been said, it is incredibly difficult to generalize a film and declare that everyone will enjoy it. As a result, recommender systems shine as unique assistants for the interests of the user.

## **Related Work** :

Following are the papers that we referred -

- f0d1e3f5683da81c9018ff3308495420.A Movie Recommender System MOVREC using Machine Learning Techniques.pdf (ijesc.org)

The above paper discusses collaborative filtering and content based filtering. Our approach of content filtering is based on the research of this paper.

- An Efficient movie recommendation algorithm based on improved k-clique | Human-centric Computing and Information Sciences | Full Text (springeropen.com)

In this research, they offered an efficient movie recommendation algorithm based on modified k-clique algorithms, which have the highest accuracy.

However, the MovieLens data is evaluated using the k nearest neighbors, the maximal clique methods, the k-clique methods, and the proposed ways to monitor the success of collaborative filtering methods.

The results reveal that the recommended strategies improve the movie recommendation system's accuracy more than any other strategy employed in this study.

- (PDF) Design and Implementation of Movie Recommendation System Based on Knn Collaborative Filtering Algorithm (researchgate.net)

The main goal of this paper is to use the KNN algorithm and collaborative filtering algorithm to help users find user-interested movies automatically in massive movie information data, as well as to develop a prototype of a movie recommendation system based on the KNN collaborative filtering algorithm.

## Data :

The Movielens dataset has been taken from the Grouplens organization. The description of the dataset has been given as follows :

With a single header row, the dataset files are created as comma-separated values files. Double-quotes are used to escape columns with commas (,). ("). These files are UTF-8 encoded. If accented characters in movie titles or tag values (e.g., Les (1995)) display wrong, make sure that any software that reads the data, such as a text editor, terminal, or script, is set to UTF-8.

The dataset contains multiple files such as :

### Ratings.csv -

- The file ratings.csv contains all of the ratings. After the header row, each line in this file represents one user's rating of one movie, and has the following format:
- userId,movie,rating,timestamp
- The lines in this file are sorted by userId first, then by movieId inside each user.
- Ratings are given on a scale of one to five stars, with half-star increments (0.5 stars - 5.0 stars).
- The timestamps reflect seconds since midnight on January 1, 1970, in Coordinated Universal Time (UTC).

### Movies.csv -

- Movie information is contained in the file movies.csv. Each line of this file after the header row represents one movie, and has the following format:
- movieId,title,genres
- Movie titles are entered manually or imported from https://www.themoviedb.org/, and include the year of release in parentheses. Errors and inconsistencies may exist in these titles.
- Genres are a pipe-separated list, and are selected from the following:
  - Action
  - Adventure
  - Animation
  - Children's
  - Comedy
  - Crime
  - Documentary
  - Drama
  - Fantasy
  - Film-Noir
  - Horror

- Musical
- Mystery
- Romance
- Sci-Fi
- Thriller
- War
- Western
- (no genres listed)

**Links.csv -**

- The file links.csv contains identifiers that can be used to link to various sources of movie data. After the header row, each line of this file represents one movie and has the following format:
- movieId,imdb
- Id,tmdb
- https://movielens.org uses an identifier for movies called movieId. The movie Toy Story, for example, has a link https://movielens.org/movies/1.
- http://www.imdb.com uses an identification for movies called imdbId. The film Toy Story, for example, has the URL http://www.imdb.com/title/tt0114709/.
- https://www.themoviedb.org uses tmdbId as a movie identifier. For example, https://www.themoviedb.org/movie/862 is the link for the film Toy Story.
- The terms of each provider apply to the use of the resources listed above.

**Tags.csv -**

- All tags are contained in the file tags.csv. Each line of this file after the header row represents one tag applied to one movie by one user, and has the following format:
- userId,movieId,tag,timestamp
- The lines within this file are ordered first by userId, then, within user, by movieId.
- Tags are user-generated metadata about movies. Each tag is typically a single word or short phrase. The meaning, value, and purpose of a particular tag is determined by each user.
- Timestamps represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

## Methods :

### Recommender System Types

Content-based systems and collaborative filtering systems are the two types of machine learning algorithms used in recommender systems. Both approaches are combined in modern recommender systems.

Let's take a look at how they work, starting with movie recommendation algorithms.

### A) Content-Based Recommendation Systems for Films

The similarity of movie properties is used in content-based techniques. If a user sees one movie, comparable movies are recommended using this type of recommender system. If a user watches a comedy starring Adam Sandler, for example, the algorithm will suggest films in the same genre or starring the same actor, or both. With this in mind, movie qualities serve as the input for creating a content-based recommender system.

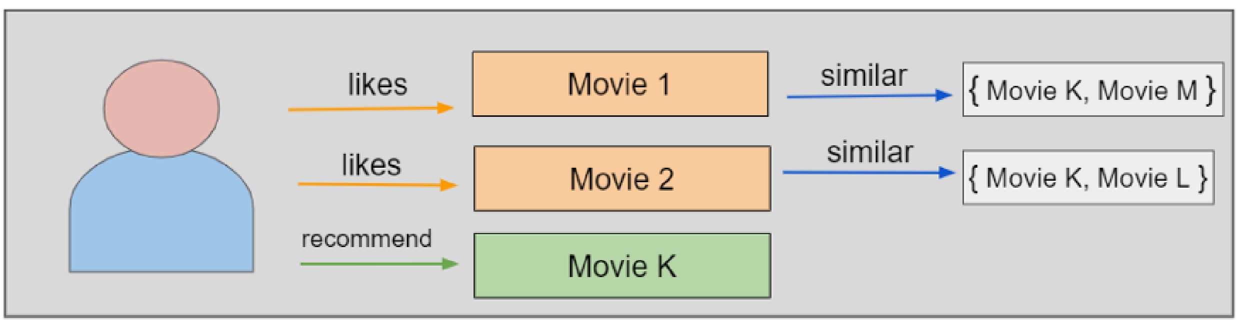



Figure 1: Content-based recommendation system overview

### B) Movie Recommendation Systems with Collaborative Filtering

The approach for collaborative filtering is based on previous interactions between users and movies. With this in mind, a collaborative filtering system's input is made up of historical data from user interactions with the movies they view.

If user A watches M1, M2, and M3, and user B watches M1, M3, and M4, we propose M1 and M3 to a user C who is comparable to user A. For a better understanding, look at the illustration below.

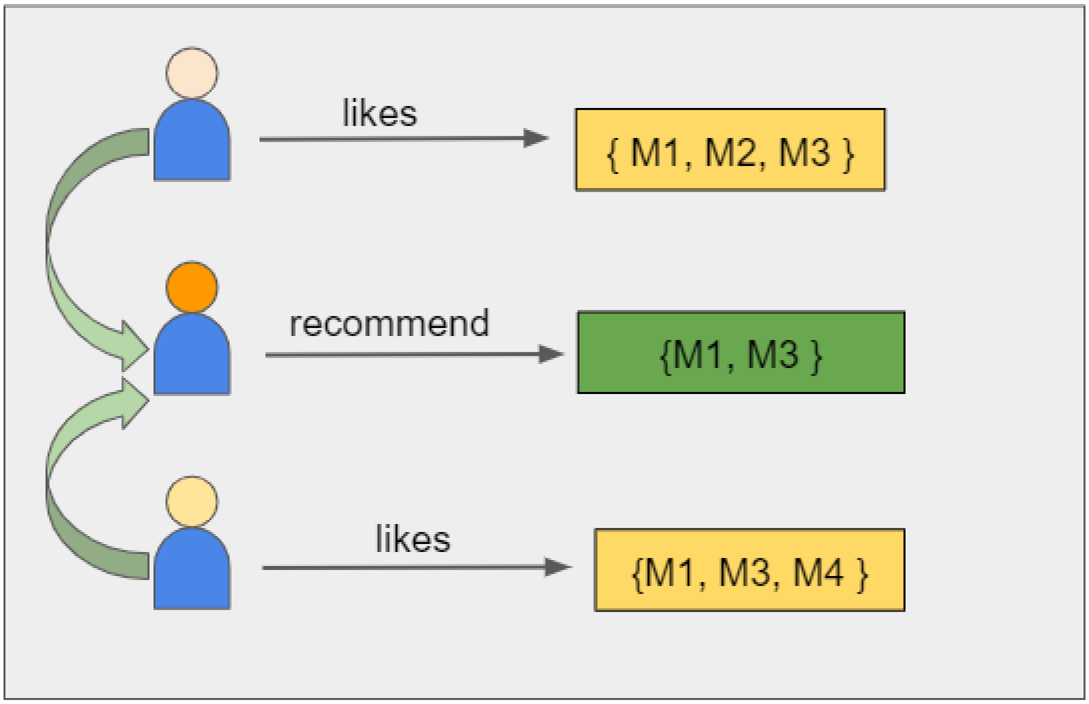



Figure 2: A collaborative filtering movie recommendation system in action.

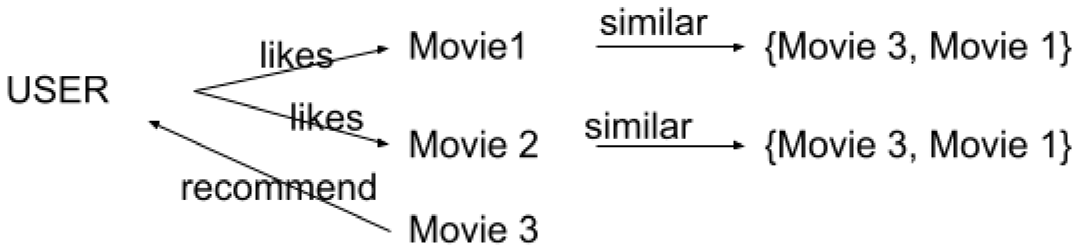
This information is recorded in a matrix known as the user-movie interactions matrix, which has rows for people and columns for movies.

## **Experiments and Results**:

Model-based collaborative filtering uses learning techniques to create a model to generate recommendations.
learning techniques in recommender systems can be categorized into the below approaches:

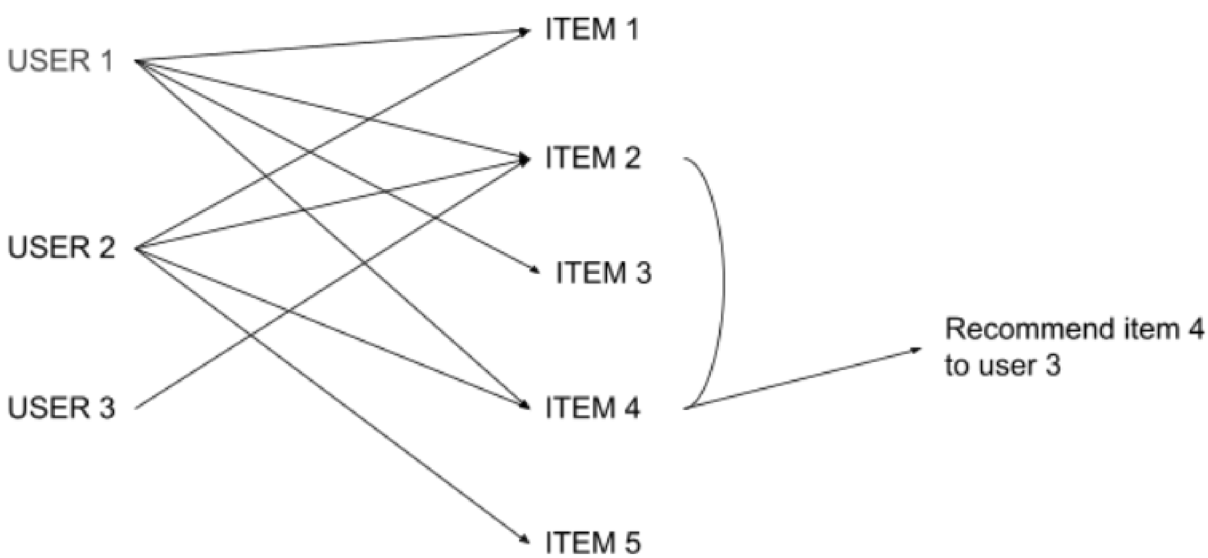
a) Using cosine similarity between non zero vectors like Content-based filtering,
b) Using item similarity like Item-item filtering,
c) Using User similarity approach like User-item filtering, and
d) using rating prediction of an item, for example, Singular Value Decomposition.
c) recommendation system using Amazon Personalize

**Content-based** methods are based on the similarity of movie attributes. Using this type of recommender system, if a user watches one movie, similar movies are recommended. For example, if a user watches a comedy movie starring Adam Sandler, the system will recommend them movies in the same genre or starring the same actor, or both. With this in mind, the input for building a content-based recommender system is movie attributes.




Recommend the new movies to the user based on the movies liked by the user by using concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) which are used for information retrieval and also content-based filtering used for recommending movies.
TF is simply the frequency of a word in a document. IDF is the inverse of the document frequency among the whole corpus of documents.
We first build the TF-IDF matrix using the genre of the movies, using this matrix we build a cosine similarity matrix which is used to recommend movies similar to the movies rated or liked by the user.
From the list of new movies, the top 10 movies are highly rated. This model is evaluated using KNN.
The hit ratio of the model built is 0.932 and fault ratio is 0.067

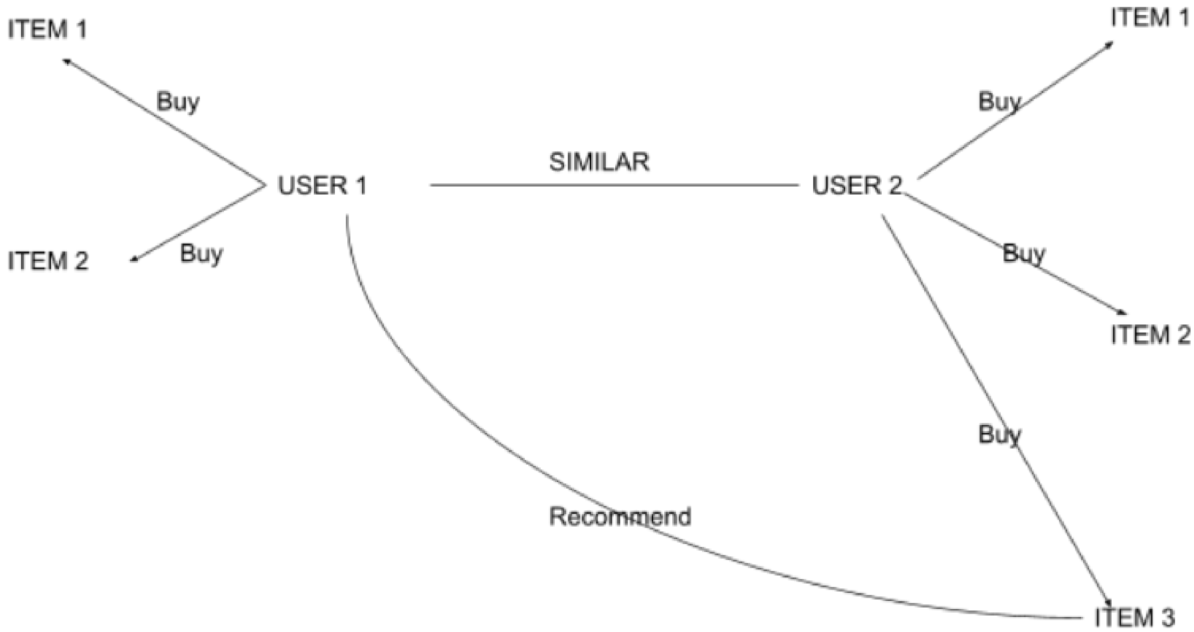
**Item-based filtering** for recommendation system based on the similarity between items calculated using user's rating of movies.




For movies that are similar to the movies that users already rated(pick only those movies from the user list which are highly rated by the user), the system will recommend those movies which have similarities greater than or equal to 0.5. These movies are arranged in the descending order of rating and the top 10 rated movies are recommended to the user.

This method of recommending the movies to the user based on his list of movies and rating to those movies is evaluated using offline analysis.

**User-item filtering** for recommendation system based on the similarity between the user and movie by finding the similar users and recommending movies based on the two similar user choices of movies.




In a similar way as we did for ItemItem similarity we will create a matrix but here we will keep rows as users and movies as we want a vector of different users. Then in similar ways, we will find distance and similarity between users. The models find the similar user who has similar interests as the given user id and find the list of new movies from the similar which are not watched by the given user. The top 10 rated movies are recommended to the user from the list of new movies.
The model build using user-item filtering is evaluated by offline analysis.

Challenges with User similarity

- The challenge with calculating user similarity is the user needs to have some prior purchases and should have rated them.
- This recommendation technique does not work for new users.

**Model Based filtering** is based on matrix factorization which has received greater exposure, mainly as an unsupervised learning method for latent variable decomposition and dimensionality reduction. Matrix factorization is widely used for recommender systems where it can deal better with scalability and sparsity than.
A well-known matrix factorization method is Singular value decomposition (SVD). At a high level, SVD is an algorithm that decomposes a matrix into the best lower rank (i.e. smaller/simpler) approximation of the original matrix. Mathematically, it decomposes the matrix into two unitary matrices and a diagonal matrix. These

matrices are used to make movie ratings predictions for every user. And based on these predictions new movies are recommended to the user.

**Amazon Personalized Recommendation System :**

Amazon Personalize is a fully managed machine learning service that enables developers to build customized recommendation systems. Amazon Personalize makes it easy for developers to build applications capable of delivering a wide range of personalization experiences, including specific product recommendations, personalized product re-ranking, and customized direct marketing. To build the system we first format the input and upload data in Amazon S3 bucket and then send the training algorithm to build on data uploaded. The algorithm used to HRNN.

**Movie Recommendation end-to-end system.**

We have built an end-to-end system movie recommendation system. We used the cosine similarity matrix of the movies to recommend the top 5 movies to the user based on his/her favorite movie. For the application as a first step we have saved the cosine similarity matrix and movie list and their ratings as a .pkl file. And based on the user's favorite movie a top 5 movies recommended will be displayed in the UI.

## Conclusion :

- We built a model for a movie recommendation system that learnt from movie ratings, genres, recent views and user preferences and recommended movies that are more likely to be watched by the user.
- We deployed the model in an application where it works with streaming data for more accurate recommendations.
- We used Amazon personalize to provide a machine learning service that goes beyond rigid static rule-based recommendation systems and trains, tunes, and deploys custom machine learning models to deliver highly personalized recommendations to customers across industries like retail and media and entertainment.