# MOVIES RECOMMENDATION SYSTEM

**Digvijay kumar (19BCS3878)**

Chandigarh university

19bcs3878@cuchd.in

## Abstract

Recommendation system are probably the most trending data science application today. Movie recommendation system provide a mechanism to assist users in classifying users with Similar interests. This make recommender system essentially a central part of websites and applications. All of the major tech companies are using recommendation system in some form or other. Like amazon is using it to suggest frequently bought together or customer who viewed this item also viewed. YouTube is using it to create an auto playlist based on your preferences. Goal of this project is making a movies recommendation system. Movie recommendation system is probably the most trending data science application today. They can be used to predict user preference for any particular items like comedy, romantic, action, top rated, trending now and Netflix original contents. We use TMDB API key for movies database which provide us to programmatically fetch and use data. So in our movie recommendation system you can see all the movies is recommended in different categories. Movie recommendation system provide a mechanism to assist users in classifying users with Similar interests. This make recommender system essentially and useful.
As the data quality and quantities increases, in future to provide a new perspective to the whole idea Of supporting users in developing, exploring and understanding their unique personal preferences. All of the major tech companies are using recommendation system in some form or other.

**Keywords**: Recommender systems, TMDB API movies database, react, node js. javascript, Web development, machine learning.

## Introduction

According to survey there find that people sometimes facing many different kinds of problems while watching and selecting movies and it's kill a lots of time while selecting or choosing which are top rated movies, which are trending movies similarly which are comedy, action, romance, horror, documentaries. So for solving these kinds of problems and people easily access their preferences, our movies recommendation system provide a good user interface so that people find their choices. and in this movie recommendation system we use TMDB API key for movie database which fetch data similar to Netflix contents and with the help of this API key we get recommendation of movies in different categories. This kind of recommendation system sort out the problem of people who find difficulties in selecting movies with their preferences. Before the industrial age, information age, and globalization, when the number of choices for the consumer in any market was limited, it was possible to solely rely on word of mouth, editorial and staff reviews of movies, books, etc., testimonials, endorsements and general surveys to make the right purchases . However, this natural social approach to gathering recommendations became impractical as the assortment of offerings in the market grew from a few dozens to millions, often overwhelming customers trying to decide what to buy, watch, read, etc. This paved the way for the deployment of intelligent recommender systems by businesses, applying statistical and knowledge discovery techniques on users' purchases datasets, to generate sound product suggestions for the buyers, as a value-added service. One such company, where a recommender system has now become indispensable, is Netflix. In 2006, Netflix announced a competition for developing recommender systems that can outperform its own system, Cinematch, leading to the succesful application of different techniques such as RBMs for movie rating predictions. To facilitate this, they released a dataset containing 100 million anonymous movie ratings and reported their Root Mean Squared Error (RMSE) performance on a test dataset as 0.9514 (James Bennett, 2007). A standard metric for evaluating the performance of rating predictors and, in general, classifiers (Herlocker, 2004), the RMSE metric is computed using the recommender's prediction, *oi* , and the actual rating provided by a user, *ti* , for *n* such ratings in the test set, as shown in equation 1. If the RMSE is squared, we get the Mean Squared Error (MSE).

$$RMSE = [\ 1\ n \sum (oi - ti\ )\ n\ 2\ i{=}1\ ]\ 1/2$$

This paper gives an overview of the key ideas of the recommendation techniques implemented in the project in section 2, followed by a description of their implementation and results in section 3, and finally concludes with a summary in section 4. In this paper, the terms "items" and "movies" are used interchangeably.

## React

React was created by Jordan Walke, a software engineer at Facebook, who released an early prototype of React called "FaxJS".He was influenced by XHP, an HTML component library for PHP. It was first deployed on Facebook's news feed in 2011 and later on instagram  in 2012. It was open-sourced at JSConf US in May 2013.React (also known as React.js or ReactJS) is a free open source frontend development for building user interface or UI components. It is maintained by meta (formerly Facebook) and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality. React code is made of entities called components. Components can be rendered to a particular element in the DOM using the React DOM library. When rendering a component, one can pass in values that are known as "props"
Here the example of how to create react page

```
npx create -react - app.
```

It is use in single page development. It work in html, css and java script.

## Node js

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

### Features of Node.js

Following are some of the important features that make Node.js the first choice of software architects.

- **Asynchronous and Event Driven −** All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

- **Very Fast −** Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.

- **Single Threaded but Highly Scalable −** Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

- **No Buffering** − Node.js applications never buffer any data. These applications simply output the data in chunks.

- **License** − Node.js is released under the MIT licence.

## TMDB MOVIES DATABASE

API is available for everyone to use. A TMDB user account is required to request an API key. Professional users are approved on a per application basis.
As always, you must attribute TMDB as the source of your data. To view all the methods available, you should head over to developers.themoviedb.org. Everything outlined on this page is simply a high level overview to help you understand what is available. The API service is for those of you interested in using our movie, TV show or actor images and/or data in your application. Our API is a system we provide for you and your team to programmatically fetch and use our data and/or images. The API provides a fast, consistent and reliable way to get third party data. A commercial API is for commercial projects and a developer API is for developers. Your project is considered commercial if the primary purpose is to create revenue for the benefit of the owner. You can apply for an API key by clicking the "API" link from the left hand sidebar within your account settings page. You need to have a legitimate business name, address, phone number and description to apply for an API key. Our API is free to use as long as you attribute TMDB as the source of the data and/or images. However, we reserve the right to charge for the commercial API key in the future.

### General Features

- Top rated movies
- Upcoming movies
- Now playing movies
- Popular movies
- Popular TV shows
- Top rated TV shows
- On the air TV shows
- Airing today TV shows
- Popular people

## TMDB

The Movie Database, also known as TMDB, is a database that contains detailed information on over 500,000 movies. The Movies dataframe has links to the movie posters in it, but the vast majority of them are outdated and no longer work. In order to find the poster links, the tmdbsimple library, which acts as a Python wrapper for the TMDB API was used. The ID of the movie is all that it takes to obtain that movie's updated information. The TMDB website has a simple structure to its URLs, therefore getting the link to each movie's page involves taking the base site link and adding the id of the movie, a dash, and the movie title.

## FUTURE WORK

if given more time, I would include Collaborative Filtering in my recommender, which would include obtaining reviews and other user generated data by using the millions of reviews in the movies dataset to find similarities between users that make it more effective at recommending movies.
Another option is to avoid using The Movies Dataset altogether and having my app use the TMDB database itself as the source of its data. This would allow the recommender to increase the number of movies by an order of magnitude. Finally, I would add additional features to the application to allow the user to restrict the types of movies that can be randomly selected by rating, year, foreign/domestic, etc.

## How work?

A classic problem people have is finding a good movie to watch without doing a lot of research. To overcome this problem,

recommender systems based on Machine Learning or Deep Learning are used to find movies that users are most likely to enjoy. There are 2 main types of recommender systems: Content Filtering (finds similarities between movies by their data), and Collaborative Filtering (finds similar movies by ratings and other data from users). This project uses Django to create a web application that allows the user to choose a movie and uses Content Filtering to recommend movies that are most similar to the one chosen by the user. Content Filtering is performed on the movie dataset, which contains metadata for over 45,000 movies. The main idea behind Content Filtering is that if a user likes a movie, then they will probably like other movies that are similar to it. Movie similarity is determined by the following metadata: Cast, Director, Keywords, and Genre. The metadata is text based, so some basic natural language processing is required before it can compare movies.
Processing the data involves the following steps:

1. Removing stop words such as the, and, or, etc. They add no relevant information, so they are not useful for most types of analysis.
2. Finding the director of the movie from the cast so the director has their own category.
3. Removing spaces between words. This is done so that names and other multiple word terms like "Johnny Depp" and "Johnny Galecki" are not treated the same.
4. Creating a metadata soup by appending each processed category to each other.

After the data is processed, Scikit-learn's CountVectorizer function is used to compute the similarity between movies by their metadata soup. Essentially, it takes the words in the soup and converts them into word vectors. Then, it performs a Cosine Similarity algorithm to compute the angle between word vectors. The smaller the angle between vectors, the more similar they are. The CountVectorizer creates a matrix with similarity scores for each pair of movies. For each movie, the scores were sorted to find the 10 most similar movies, which were added as a column to the dataframe, so they would not need to be calculated at runtime.

## CONCLUSION

Recommender systems add value to businesses by assisting their customers in selecting the right product out of uncountable choices. This project implemented five of the popular movie recommendation approaches to predict unknown ratings and recommend to users accordingly. After implementation, their performances were compared using RMSE and MSE metrics. SVD was found to have performed better than CF. The variation of $k$, the number of features, in SVD did not change RMSE significantly. Two different approaches were tried out for neural networks: one neural network for all users, and one neural network for each user. In both cases, ‘tanh’ activation function performed better than others.

## ACKNOWLEDGMENTS

We are highly thankful to the Grouplens for making the datasets available to be downloaded over internet which we used for our experiments and we are also very thankful to IMDB for providing API’s to fetch data from its database. At the last but not the least, we are thankful to our Professor and all TA’s for helping us to understand the concepts.

## REFERENCES

1.https://www.geeksforgeeks.org/reactjs-tutorials/
2. https://www.geeksforgeeks.org/how-to-fetch-data-from-an-api-in-reactjs/
3. https://www.themoviedb.org/documentation/api
4. https://www.npmjs.com/package/movie-trailer
5. https://arxiv.org/ftp/arxiv/papers/1909/1909.12749.pdf