BUILD A MOVIE RECOMMENDATION SYSTEM

Pratik Biswas (B20EE043)
Krishi Patel (B20EE030)
Palaskar Adarsh Mahesh (B20EE087)

Abstract - This paper reports our experience of building multiple models that recommend movies based on specific inputs that are provided to it, like user ratings or user preferences.

I. INTRODUCTION

A recommendation system, is a subclass of information filtering system that seeks to predict the rating or preference a user would give to an item.They aim to predict users' interests and recommend product items that quite likely are interesting for them. In our case, this item is a movie, therefore the main focus of our recommendation system is to filter and predict only those movies which a user would prefer given some data about the user him or herself.

Recommendation systems are mainly of two types -

1. **Collaborative Filtering** - Collaborative filtering approaches build a model from the user's past behaviour as well as similar decisions made by other users. This model is then used to predict items that users may have an interest in.
2. **Content based Filtering** - Content-based filtering approaches use discrete characteristics of an item in order to recommend additional items with similar properties. They are totally based on a description of the item and a profile of the user's preferences. It recommends items based on the user's past preferences.

Dataset

The MovieLens dataset has been used for this project, which describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018.

II. METHODOLOGY

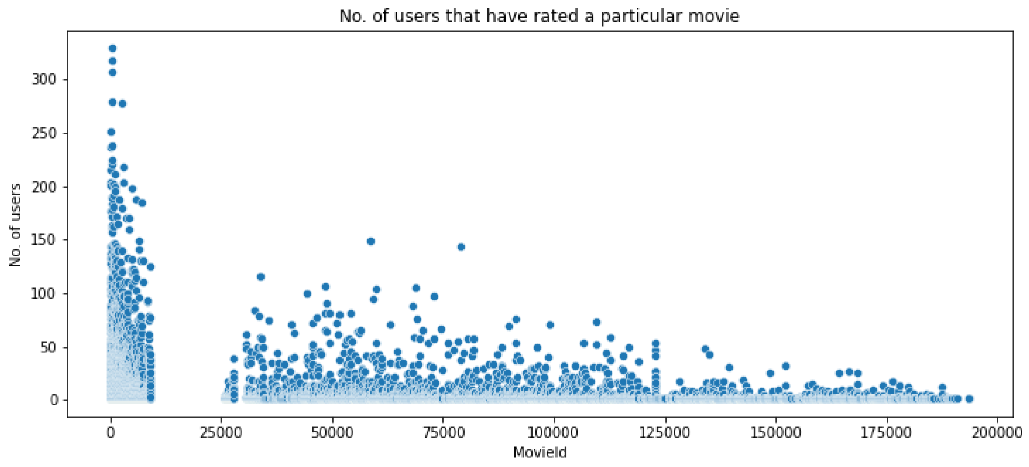
Overview

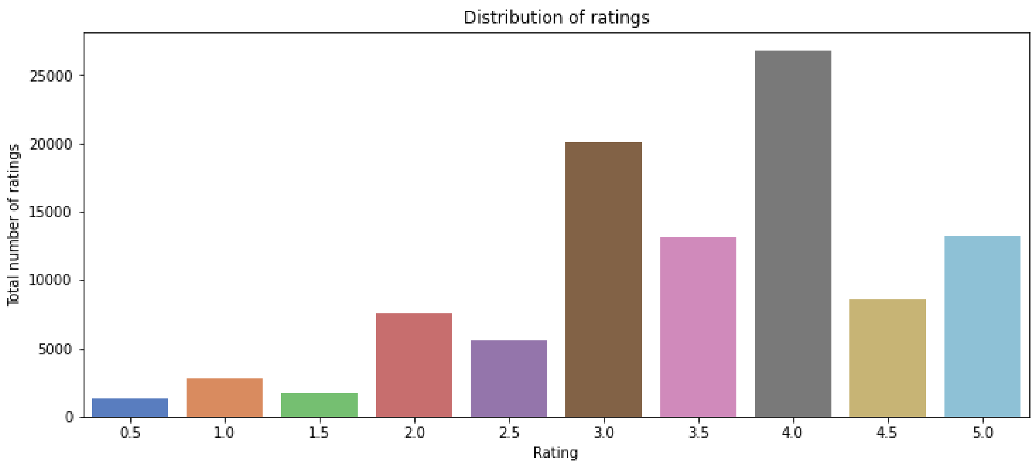
We have implemented the following algorithms -

1. Nearest Neighbours
2. Cosine Similarity
3. Pearson Similarity
4. Count Vectorizer and TF-IDF Vectorizer
5. Linear Regression

Exploratory Data Analysis

We have plotted graphs showing the various relationships between users and the movies that they have rated.





Data Preprocessing

No null or NaN values were found in the dataset, and there were no excessive or unwanted features. Hence, we do not need data preprocessing of any kind.

Implementation of Recommendation Systems

We have tried a total of 5 recommendation systems on our dataset. WIth each model, we have used them to recommend 10 movies by giving each the same movie as input and comparing their results. For some of the models, we have also evaluated their performance using RMSE as the evaluation metric.

The details of the systems used are -

1. User-user Collaborative Filtering using Nearest Neighbours

NearestNeighbors implements unsupervised nearest neighbours learning. The principle behind it is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be user-defined, or vary based on the local density of points.

In user-user collaborative filtering, we find look-alike users based on similarity and recommend movies which the first user’s look-alike has chosen in the past.

The steps used in this system are -

(1) From the ratings dataset, create a matrix with movieId as the rows and userId as the columns.
(2) Sparsity reduction of the matrix obtained above.
(3) Applying NearestNeighbors with k=10 to find the 10 movies closest to the input movie.

2. User-User Collaborative Filtering using Pearson and Cosine Similarity

The Pearson correlation coefficient is a measure of linear correlation between two sets of data. It is the ratio between the covariance of two variables and the product of their standard deviations, and has a value between −1 and 1.

Cosine similarity measures the similarity between two vectors of an inner product space. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction.

The steps used in this system are-

(1) From the ratings dataset, create a matrix with movieId as the rows and userId as the columns.
(2) The matrix was split into train and test data in the ratio 80:20.
(3) User based correlation was found with both Pearson and Cosine similarity.
(4) A similarity matrix was generated and then used to make predictions.

3. Item-Item Collaborative Filtering using Pearson and Cosine Similarity

Item-item collaborative filtering is a form of collaborative filtering for recommender systems based on the similarity between items calculated using people's ratings of those items.

The steps used in this system are the same as the ones used above, except item based correlation has been used to generate the similarity matrix.

4. Content-based Filtering using TF-IDF Vectorizer and Cosine Similarity

Content-based Filtering uses attributes such as genre, director, description, actors, etc. for movies to make suggestions for the users. The intuition behind this is that if a user liked a particular movie or show, he/she might like a movie or a show similar to it.

TF-IDF (Term Frequency Inverse Document Frequency), is a very common algorithm to transform text into a meaningful representation of numbers used to fit a machine algorithm for prediction.

The steps used in this system are -

(1) TF-IDF Vectorizer has been applied to the ‘genres’ column in the movies dataset, converting them into feature vectors.
(2) Cosine Similarity has then been used on the feature vectors to find out the similarity between any two movies.

5. Hybrid Filtering using Linear Regression

A hybrid recommendation system is a special type of recommendation system that can be considered as a combination of both content-based and collaborative filtering methods.

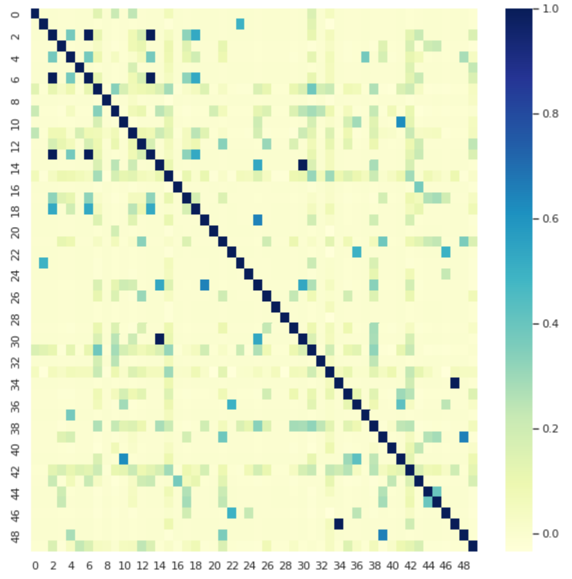
The steps used in this system are -

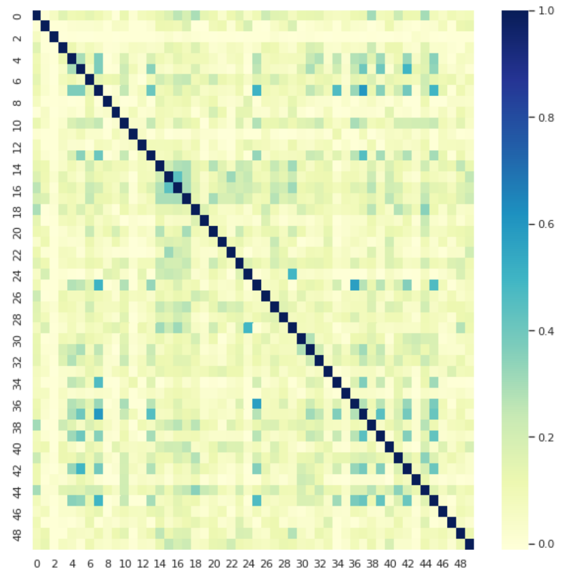
(1) Create a new dataframe of all the movies as rows and all the genres as columns. If a movie belongs to a particular genre, enter value 1 in the cell corresponding to that genre, else 0.
(2) All the movies watched by a particular user are taken as input. The rating of each movie is multiplied by the genre row of that movie.
(3) The sum of all the genre values for that movie is obtained, giving us a set of weights of the same length as the number of genres.
(4) Each movie’s genre row in the dataset is then multiplied with this set of weights. And the movies with the highest dot product are chosen as the recommendation.

III. SYSTEM RESULTS

1. Not all of the systems that we have made can be evaluated quantitatively. For those which can, we will first evaluate their performance using RMSE as our evaluation metric. The results are as -

Recommender Type	**Correlation Coefficient used**	**RMSE (train data)**	**RMSE (test data)**
User-user based collaborative filtering	**Cosine Similarity**	3.423	2.890
	Pearson's Similarity	3.478	2.914
Item-item based collaborative filtering	**Cosine Similarity**	3.578	3.074
	Pearson's Similarity	3.581	3.069





Left : Correlation visualised in user-user based filtering with Pearson's similarity.
Right : Correlation visualised in item-item based filtering with Pearson's similarity.

2. We will analyse the other systems qualitatively. We will take a particular movie as input, and feed it into the systems to find out the top 5 movie recommendations of each. We will then compare these recommendations to check if our system is performing well or not.

   Input movie chosen - Iron Man (2008).

Recommendation System used		**User-user with Nearest Neighbours**	**Content-based with TF-IDF and Cosine Similarity**	**Hybrid Filtering with Linear Regression**
Movies recommended by the system	**1.**	Guardians of the Galaxy	Star Wars: Episode IV - A New Hope	The Bourne Ultimatum
	2.	Pirates of the Caribbean: At World's End	Stargate	Sherlock Holmes
	3.	Star Trek	Demolition Man	Inception
	4.	Kung Fu Panda	Star Wars: Episode V - The Empire Strikes Back	Avatar
	5.	X-Men: First Class	Star Wars: Episode VI - Return of the Jedi	Wall-E

IV. ANALYSIS

1. For user-user collaborative filtering, Cosine Similarity gave us a better performance on the model as compared to Pearson's Similarity (lesser RMSE).
2. For item-item collaborative filtering however, Pearson's Similarity gave us a better performance as compared to Cosine Similarity.
3. For the other 3 systems, we had chosen Iron Man as our input movie. It is of the genres - Action, Superhero, Adventure, Thriller.
4. User-user collaborative filtering with Nearest Neighbours gives us good recommendations, with all movies being adventure type, having fictional superhero characters and aimed towards kids.
5. Content-based recommendations with TF-IDF and Cosine Similarity recommend mostly Star Wars movies. Its movies are also adventure type, but do not contain any superheroes. This system does not perform extremely well.
6. Hybrid Filtering with Linear Regression contains movies which are adventure type as well, but none of them contain superheroes. Also, some of the movies are more oriented towards a mature audience rather than kids, being more of the hard action genre than a lighter, fantasy one. This system does not perform very well.

CONTRIBUTIONS

1. **Pratik Biswas (B20EE043) -**

   Performed literature review, made an end-to-end implementation of the user-user collaborative recommender system using the Nearest Neighbours algorithm, and also made the report.

2. **Krishi Patel (B20EE030) -**

   Performed literature review and made End to end implementation the hybrid model of content and collaborative filtering

3. **Palaskar Adarsh Mahesh (B20EE087) -**

   Performed literature review about various methods and models used for recommendations. Made End to end implementation of TF-IDF vectorization model and item-item and user-user based collaborative filtering using Pearson and Cosine similarities.