ICBAI CONFERENCE 2021

Comprehensive Movie Recommendation System

AUTHORS:
HRISAV BHOWMICK
ANANDA CHATTERJEE
JAYDIP SEN

PRESENTER:
ANANDA CHATTERJEE

DATE: 21.12.2021

Objective

- Propose an overview of approaches and techniques to make a robust recommendation system to pitch movies of particular interest to users.
- Illustrate different strategies to overcome the constraint of proposed previous one.
- Document 8 techniques of recommendation with their merits and limitations and pick up the best one among them.

Introduction

Content based filtering- Recommend movies that are similar to the one the user watched before using information available about the already watched movie. Similarity is checked using cosine similarity metric based on the available information.

Collaborative filtering- Matches users who watched similar movies to recommend unseen movies. There are two kinds of collaborative filtering, one is user based and the other is item based.

- **User based filtering-** The main idea is to discover users who have similar prior preference patterns as user 'A,' and then recommend to him or her goods that those similar users have enjoyed but that 'A' has not yet encountered.
- **Item based filtering-** In this scenario, the goal is to locate similar movies rather than similar users, and then to recommend similar movies to those that 'A' has had in his or her past preferences. This is accomplished by locating every pair of items that were rated by the same user, calculating the similarity of those rated across all users who rated both, and then recommending them based on the similarity scores.

Data Collection & Methodology

- The data has been extracted from **Movie Lens dataset** which contains 1,00,836 ratings and 3,683 tag applications across 9,742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018.

- The **movie dataset** has 9,742 records and 3 columns which are movieid, title, genres. The **ratings dataset** contains 1,00,836 records and 5 columns. The **tags dataset** is contained of 3,683 records and 4 columns.

- **8 methods** have been applied in this work for recommendations.

Genre Based

- This form of recommendation system displays relevant items based on the content of the users' previously searched items.

- Suppose if ‘action’ genre is preferred to watch by a user then that user will be recommended top movies from action genre based on **weighted score**.

  score = (v/(v+m) *R) + (m/(m+v) *C)

  v = number of ratings for the movie
  m = minimum number of ratings required to be eligible
  R = average rating of the movie
  C = mean rating across whole data

- Top 5 recommended movies based on **“action” genre**: Fight Club (1993), Star Wars: Episode IV- A New Hope (1997), Dark Knight (2008), Princess Bride (1987), and Star Wars: Episode V - The Empire Strikes Back (1980).

Using Pearson Correlation Coefficient

- This method helps to find out **movies that are correlated** to another movie. The range of Pearson Correlation Coefficient values lie between **-1 to +1**, more it tends to +1 better is the correlation of that movie with the other one.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

xi, yi = individual sample points
n = sample size
$\bar{x}$ = mean
rxy = defines correlation coefficient

- Top 5 movies highly correlated with Titanic (1997), are Gossip (2000), White Man's Burden (1995), Rapid Fire (1992), Imaginary Heroes (2004), Killer Elite (2011).

Using Cosine Similarity

- This approach determines how similar two users are by calculating the **cosine of the angle between two vectors**. Here vectors are nothing but the movies. More the angle tends to 0, more the movies will be similar.

$$\text{Sim}(x,y)=\cos(\vec{X},\vec{Y}) = \frac{\vec{X}.\vec{Y}}{|X|.|Y|}$$

X, Y = two movies between which the cosine angle is to be found

- **cosine_similarity()** function was used to serve the purpose.

- Top 5 movies which are cosine similar with Titanic (1997) were Leaving Las Vegas (1995), Persuasion (1995), How to Make an American Quilt (1995), Bed of Roses (1996), and Angels and Insects (1995).

KNN algorithm (with cosine metric)

- KNN algorithm-based recommendation falls under collaborative filtering.
- In our work, **item based collaborative filtering** has been used where similarity between particular item and the K other particular items have been calculated using Cosine similarity.
- Movie vs User rating matrix was taken and KNN algorithm was fitted on that data. The value of **k has been taken as 5**. Cosine similarity was used as the metric to find the nearest neighbors. A movie index was sent as query, and its nearest or similar top 5 movies were returned as recommendation. So, the least distance between top 5 movies with the target movie has been evaluated and then recommended to the user who watched the target movie.
- Top 5 movies recommended movies for Mezzo Forte (1998) were Guyver: Dark Hero (1994), Harrison Bergeron (1995), Real Life (1979), The Punisher: Dirty Laundry (2012), House of Cards (1993).

Clustering of Movies

- Clustering is another approach of recommending movies where by looking at the movies a user watched, find similar movies based on its cluster and recommend them. **K-means clustering** algorithm has been used in our study for recommendation.

- The K-means algorithm's main goal is to reduce the sum of distances between points and their corresponding **cluster centroid**. Here the concept of **inertia** comes into action which calculates sum of the distances of all the movies within a cluster from their cluster centroid.

- Here are the 5 similar movies obtained from **"cluster 3"**: Toy Story (1995), Jumanji (1995), Tom and Huck (1995), Balto (1995), Now and Then (1995).

Content latent matrix

- A “metadata” column was made using reviews, genres, tags, movie names. It was **vectorized using TFIDF** for 9719 movies. A sparse matrix was formed of dimension 9719 x 9658. It was compressed to **1000 components using SVD**, and only top 1000 features were kept. Now choose any one movie and find top 5 similar movies using **cosine similarity** based on the matrix.

- Top 5 recommended movies for ‘Batman Begins (2005)’ were: ‘Batman: Mystery of the Batwoman (2003)’, Batman: Assault on Arkham (2014), Batman (1989), Batman: Year One (2011), Batman: The Killing Joke (2016).

Collaborative latent matrix

- A movies vs user “rating matrix” was formed of dimension 9719 x 610. The sparse matrix was compressed to **100 components using SVD**, and only top 100 features were kept. Now choose any one movie and find top 5 similar movies using **cosine similarity** based on the matrix.

- Top 5 recommended movies for ‘V for Vendetta (2006)’ were: Dark Knight, The (2008), Pirates of the Caribbean: The Curse of the Black Pearl (2003), Iron Man (2008), Kill Bill: Vol. 1 (2003), and 300 (2007).

Surprise Library (with KNN Basic)

- The dataframe having “rating” inputs is taken and passed through k-fold cross-validation (where k=5). The model is trained with **KNNBasic()** algorithm. Now suppose a user is chosen (with uid = 26) who has not seen a particular movie (let movieid = 400). Our model will predict what rating the user would have given to the movie (here it predicted a rating of 3.5).

- We can also get the estimated/predicted rating of all the movies that the user has not seen, and can recommend him top movies from that list.

- In our case for userid = 450, the movies were selected as top 5 were Braveheart (1995), Taxi Driver (1976), North by Northwest (1959), One Flew Over the Cuckoo's Nest (1975), Saving Private Ryan (1998).

Conclusion

- **Genre based recommendation** is the simplest one and user profile similarity doesn't matter here.
- User profile similarity can be found out using **Pearson Correlation Coefficient** based recommendation but its computationally complex, causing consumption of lot of time and memory. User profile similarity can also can be successfully tracked using **Cosine Similarity** metric.
- **Cluster based recommendation** brings movies of same genres in one cluster but fails in a case where a particular movie is related to many genres, then it might throw wrong recommendation.
- In **content based filtering**, model can capture the specific interests of a user, and can recommend movies that very few other users are interested in. But the model can only make recommendations based on existing interests of the user and also needs domain knowledge.
- In **collaborative filtering**, we don't need any domain knowledge. But it may suffer from **cold-start** problem.
- Among all, collaborative filtering algorithms suits best for recommendation.

THANK YOU