Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 04, April-2020

Movie Recommendation System Approaches

Prince Praveen
B.E(Information Technology)
Universal College of Engineering
Mumbai, India

Praveen Goud
B.E(Information Technology)
Universal College of Engineering
Mumbai, India

Sagar Parmar
B.E(Information Technology)
Universal College of Engineering
Mumbai, India

***Abstract*—A Movie Recommendation system is a system that provides movie suggestions to users based on some dataset. Such a system will predict what movies a user will like based on the attributes of previously liked movies by that user. Content-Based recommendations have long been in fashion but they tend to overlook some great suggestions that may not be covered by mere cosine similarities. To overcome such shortcomings, we will combine collaborative filtering techniques having a User-User matrix with neural networks to provide users(who have already rated movies previously) with appropriate suggestions.**

Keywords—Content-based, Collaborative Filtering, User-User matrix, Neural Networks.

I. Introduction

Recommendation Systems have been in existence for a long time and have served towards convenience of people. In the most general way, recommender systems are algorithms aimed at suggesting relevant items to users(items being books to read, products to buy, music to listen or as in our case, movies to watch.) The most common approach towards recommendation systems has been the Content-based recommendations. Content-based techniques develop representations of clients and items through the investigation of additional data, for example, record content, client profiles and the traits of items, to make suggestions .In many cases, the data that is utilized to build the pictures is difficult to get or even fake; subsequently, its presentation and application run experience the ill effects of significant restrictions. Thus, we take another approach called as the Collaborative Filtering where instead of only depending upon the attributes of the item, we let our users rate different items and based on such ratings we make recommendations using two ways, either by User-User similarities or Item-Item Similarities. In User-User similarity approach, we let our users rate movies and then find other users who rated movies similarly to our test user i.e we find other users who feel similarly as our test user. On the other hand, an Item-Item approach includes letting our user rate items and then find other items that have similar ratings. Although this method is simple and effective, with the rapid development of the Internet, the high sparsity of the data limits the performance of the algorithm. To overcome it, we will combine collaborative filtering with Neural Networks We will use neural network embeddings to improve our model.

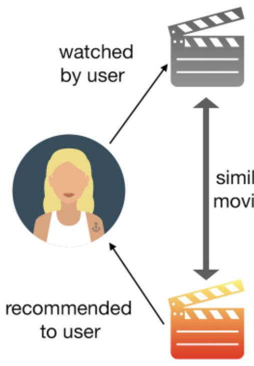
Embeddings are a way to represent discrete — categorical — variables as continuous vectors. In contrast to an encoding method like one-hot encoding, neural network embeddings are *low-dimensional and learned*, which means they place similar entities closer to one another in the embedding space. First, we use the user-item rating matrix to obtain the features of the users and items, then we pass those features as the input of our neural network. In the output layer, we will obtain some values that represent the probabilities of the scores that the user might give. Finally, the score with the highest probability will be used as the prediction result. Such a combination yields better results and enables our system to give out more accurate recommendations.

II. PROPOSED SYSTEM

We will build Movie recommendation systems with various approaches and with each step, we will get more advanced and improve the quality of the suggestions made by the proposed system.

A. Content-Based Recommendation System :

This approach for recommending movies does not involve other users. Based on what we like, our algorithm will pick similar items i.e items having similar content and recommend us.



In this approach, the diversity in recommendations will be the least as it only takes into consideration what the user specifically likes. E.g, A user that says they like Action movies will only be recommended other action movies until they try some other genre autonomously and decide to give it a like. Ofcourse, there are many categories we can calculate the similarity on: as in our case of movies, we can decide to find similarity based on genre, keyword, cast,director and so on.

Algorithm used :

COSINE SIMILARITY

To find similar content for our item, we used the cosine similarity algorithm. The dot product between two vectors is equal to the projection of one of them on the other. Therefore the dot product of two identical vectors is equal to their squared modules. On the other hand if the two vectors do not share any directions, the product will be zero. General formula for calculating dot product is given below:

$$\boldsymbol{u} \cdot \boldsymbol{v} = [u_1\ u_2\ \dots\ u_n] \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + \dots + u_n v_n = \sum_{i=1}^{n} u_i v_i$$

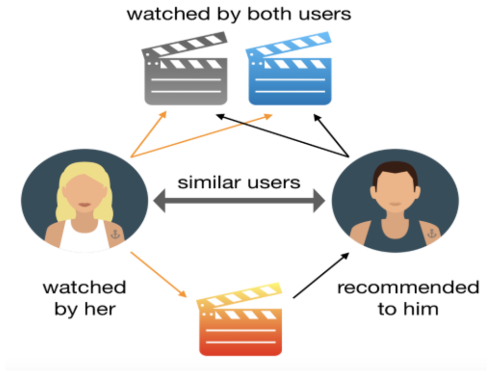
This dot product is important when defining the similarity as it is directly connected to it. The definition of similarity between two vectors u and v is in fact the ratio between their dot products and product of their magnitudes.

$$similarity = cos(\theta) = \frac{\boldsymbol{u} \cdot \boldsymbol{v}}{\|\boldsymbol{u}\| \|\boldsymbol{v}\|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \sqrt{\sum_{i=1}^{n} v_i^2}}$$

Thus, this will be equal to 1 if the two vectors are identical or it will be 0 if the two are orthogonal.

B. Collaborative Filtering Recommendation System :

The previous approach didn't involve other users and in so it had some shortcomings. Such limitations involve the recommendations not being diverse as discussed before. To solve such problems we use the collaborative filtering technique. This approach is based on the idea that the user rates, and the system will recommend different movies that the user has not watched but the other users similar to our test user have watched and liked. This type of collaborative filtering approach is called the User-to-User Collaborative filtering approach as we find similar users to our user.
To determine whether the two users are similar or not, we consider the movies watched by both of them and how they rated them. Thus by looking at items in common, we will predict the ratings a user will give to a movie who hasn't watched it yet, based on its similar user rates.



Algorithms Used:

i. K Nearest Neighbors:

The standard method of Collaborative Filtering is known as **Nearest Neighborhood** algorithm. We have an n × m matrix of ratings, with user u_i, i = 1, ...n and item p_j, j=1, …m. Now we want to predict the rating r_{ij} if target user i did not watch/rate an item j. The process is to calculate the similarities between target user i and all other users, select the top X similar users, and take the weighted average of ratings from these X users with similarities as weights.

$$r_{ij} = \frac{\sum_k Similaries(u_i, u_k) r_{kj}}{number\ of\ ratings}$$

However, not all users have the same baseline for giving ratings to movies. Some users may tend to give high scores generally while some are pretty strict with their ratings even though they are satisfied with the items. To avoid such bias, we will subtract each user's average ratings of all the items when computing weighted average, and add it back for the target user as shown:

$$r_{ij} = \bar{r}_i + \frac{\sum_k Similaries(u_i, u_k)(r_{kj} - \bar{r}_k)}{number\ of\ ratings}$$

ii. Matrix Factorization :

Sparsity is a big issue that needs to be addressed while creating collaborative filtering recommendation systems. Our approach creates matrices where rows are unique users in our environment and the columns represent different movies and the values within are the ratings that different users give to movies. However, it is rather obvious that not all movies will be rated by each user. Thus this matrix of ours faces the problem of sparsity that needs to be solved. For this purpose, we use Matrix Factorization. In this method, we decompose the original sparse matrix to low-dimensional matrices with latent features. Therefore matrix factorization gives us how much a user is aligned with a set of latent features, and how much a movie fits into this set of latent features.

The advantage of this approach over the previous algorithm is that even though two users haven't rated same movies, it is still possible to find out the similarity between them if they share similar latent features.

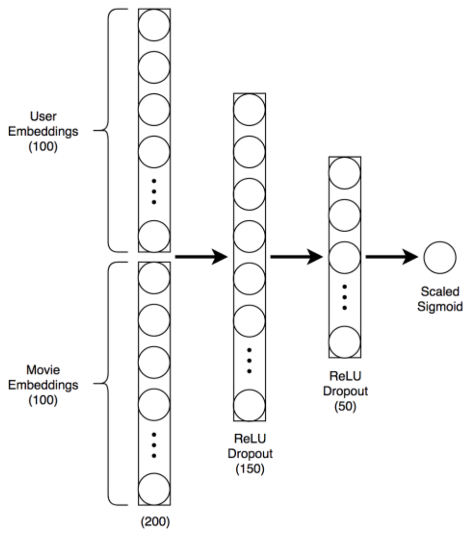
C. Neural Network Approach :

Lastly, we will combine our collaborative filtering results with neural network techniques to improve the quality of our recommendations. We will be using neural network embeddings to get desired results.
Such embeddings are a way to represent discrete categorical variables as continuous vectors. These are low-dimensional and learned, this means that they place similar entities closer to one another in the embedding space. There are many ways to compute embeddings, but the end goal is to map "things" to a latent space with complex and meaningful dimensions. In a

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 04, April-2020

movie recommender, you can imagine latent dimensions measuring genres such as “sci-fi” and “romance” and other concepts such as “dialogue-driven versus action-packed”. Users and movies are mapped to this space by how strongly associated they are with each of the dimensions.

A fully connected neural network is used to find movie and user embeddings. In this architecture, a user embedding matrix of size (n_users, n_factors) and a movie embedding matrix of size (n_movies, n_factors) are randomly initialized and subsequently learned via gradient descent. Each training data point is a user index, movie index and a rating (on a scale of 1–5). The user and movie indices are used to lookup the embedding vectors (which are rows of the embedding matrices). These vectors are then concatenated and passed in as the input to the neural network.



III. PROCESS CONCEPTUALIZED IN STEPS:

i. User Rates movies
ii. A matrix with user , movies and the respective ratings is created
iii. Sparsity of the matrix is dealt with by using matrix factorization
iv. Similar users to our target user is found using similarity algorithms such as Pearson Correlation or Cosine similarity.
v. Ratings for a movie by the target user is predicted by comparing them with other similar users.
vi. Latent features can also be compared and used for making recommendations.
vii. Such predictions are passed through neural network embeddings to find out the probability of ratings.
viii. Highest probability ratings are selected and assigned to new movies that the target user hasn’t watched.
ix. Based on the assigned ratings, it is decided whether or not to recommend that movie to our user.

IV. CONCLUSION

Recommendation systems have become an important part of everyone’s lives. With the enormous number of movies releasing worldwide every year, people often miss out on some amazing work of arts due to the lack of correct suggestion. Putting machine learning based Recommendation systems into work is thus very important to get the right recommendations. We saw content-based recommendation systems that although may not seem very effective on its own, but when combined with collaborative techniques can solve the cold start problems that collaborative filtering methods face when run independently. Similarly such systems can be improved further by applying neural network embeddings to uplift the quality of recommendations and make them more user personalized.

Thus we conclude that studying various approaches towards recommendation engine is vital to come up with a hybrid engine that overcomes the shortcomings of these independent approaches and multiplies their benefits. Where independent approaches towards a movie recommendation system may have shortcomings, when combined the right way they will help users get the accurate recommendations for movies.

REFERENCES

[1] M. J. Pazzani and D. Billsus, ‘‘Content-based recommendation systems,’’ in The Adaptive Web. Berlin, Germany: Springer, 2007, pp. 325–341.
[2] X. Su and T. M. Khoshgoftaar, ‘‘A survey of collaborative filtering techniques,’’ Adv. Artif. Intell., vol. 2009, Aug. 2009, Art. no. 421425.
[3] M. N. Jelassi, S. B. Yahia, and E. M. Nguifo, A personalized recommender system based on users’ information in folksonomies, in Proc. 22nd Int. Conf. World Wide Web, Rio de Janeiro, Brazil, 2013.
[4] I. Portugal, P. Alencar, and D. Cowan, The use of machine learning algorithms in recommender systems: A systematic review, Expert Syst. Appl., vol. 97, pp. 205–227, 2018
[5] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, Eigentaste: A constant time collaborative filtering algorithm, Information Retrieval, vol. 4, no. 2, pp. 133–151, 2001.
[6] J. S. Breese, D. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in Proc. 14th Conf. Uncertainty in Artificial Intelligence, Madison, WI, USA, 2013, pp. 43–52.