Movie Recommendation System

Dhruv Kaushik
Department of CSE
IIIT Delhi
New Delhi, India
dhruv18037@iiitd.ac.in

Gurpreet Singh
Department of CSE
IIIT Delhi
New Delhi, India
gurpreet18098@iiitd.ac.in

Wrik Bhadra
Department of CSE
IIIT Delhi
New Delhi, India
wrik18027@iiitd.ac.in

I. Problem Statement

Recommender systems have become ubiquitous in our lives. Be it e-commerce websites or social media platforms, recommender systems add the "what-next" factor to it. Due to the advances in recommender systems, users constantly expect good recommendations. They have a low threshold for services that are not able to make appropriate suggestions. This has led to a high emphasis by tech companies on improving their recommendation systems. However, the problem is more complex than it seems.

In this project, we aim to give personalized recommendations to users based on the movies that they have already rated.

II. Literature Review

Content-based filtering makes recommendation based on similarity in item features. Popular techniques in content-based filtering include the term frequency / inverse document frequency (tf-idf) weighting technique in information retrieval [1][2] and word2vec in natural language processing. An extension of word2vec, called doc2vec [3] is used to extract information contained in the context of movie descriptions. Content-based filtering works well when there hasn't been enough users or when the contents haven't been rated. Collaborative filtering recommends items that similar users like, and avoids the need to collect data on each item by utilizing the underlying structure of users' preference. One major approach in collaborative filtering is neighborhood model [4]. The neighborhood model recommends the closest items or the closest user's top rated items.

III. Dataset Description

Dataset is provided at Kaggle as The MovieLens Dataset [4]. The original dataset contains data of 45,000 movies with features like cast, genre, revenue, language, release date, etc. The whole dataset contains 26 million ratings rated by 270,000 users. A rating is a decimal value between 0 and 5 in multiples of 0.5.
Our project considers a subset of the original data comprising of details of 4320 movies and a total of 1,19,000 ratings given by 6000 users.

IV. Tasks Completed

The following tasks have been completed until now.

A. *Feature Selection & Normalization*

The baseline model performs content-based filtering using Tf-Idf scores of features. Features selected for finding similarity between movies include actor names, director names, genres, keywords and overview. Punctuations and spaces are removed from actor and director names individually and they are converted to lower case to form single tokens. Movie descriptions are usually long, whereas actor and director names are single token features. Combining these features into a single vector would overweight the overview feature and underweight the remaining features. Thus, we formed two feature groups – one containing just overview and another containing the remaining features.

B. *Vector Generation & Visualization*

For each movie, we generated two feature vectors; one feature vector corresponding to each feature group. A feature vector contains Tf-Idf values for the tokens present in the corresponding features of the movie. Thus, there are two feature vector spaces which are visualized as follows:

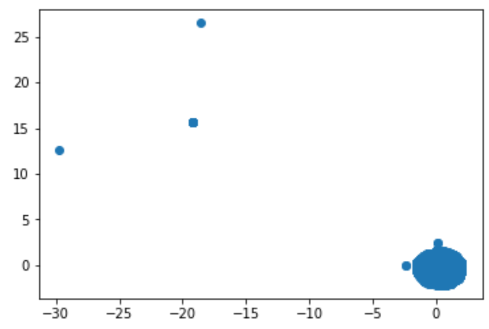

Fig. 1: TSNE plot for feature space corresponding to *overview* feature

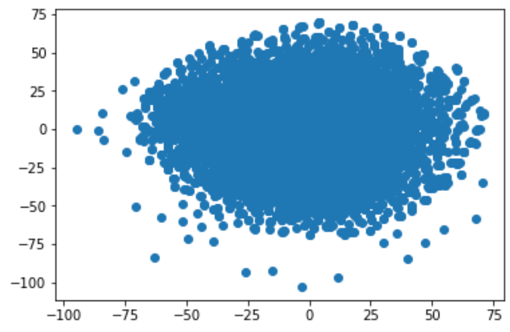

Fig. 2: TSNE plot for feature space corresponding to features *actor*, *director*, *genres* and *keywords*

C. *Similarity Matrix Generation*

For each vector space, we generate a similarity matrix. To compute similarity score between two movies, cosine similarity is calculated between their respective feature vectors. To compute similarity matrix for one feature group, we compute similarity score for each pair of vectors in the corresponding vector space.

The overall similarity score, Sim_sc(i,j) between movie i and movie j is obtained as :

Sim_sc(i, j) = alpha * Sim_sc_1(i, j) + (1 - alpha) * Sim_sc_2(i, j)

where,
Sim_sc_1(i, j): value of cell (i, j) in Similarity Matrix 1
Sim_sc_2(i, j): value of cell (i, j) in Similarity Matrix 2

Rating predicted of movie i for user u

$$\widehat{r_{ui}} = \mu_u + \frac{\sum_j sim(i,j)(r_j - \mu_u)}{\sum_j sim(i,j)}$$

Where μ_u is the average rating done by user 'u'.
And r_j is rating of movie 'j'.
And $\widehat{r_{ui}}$ is predicted rating of user 'u' for movie 'i'.
And $sim(i,j)$ is similarity value between movie 'i' and movie 'j'.

D. *Training and Testing*

We partitioned 6000 users in two sets – one is validation set of 5100 users and the other as testing set of 900 users. We need to estimate the optimal value of weight parameter alpha. We considered different values of alpha ranging from 0.05 to 1, in successions of 0.05, and for each value of alpha, we computed the corresponding RMSE (Root Mean Square Error) over the validation set, plotted below:

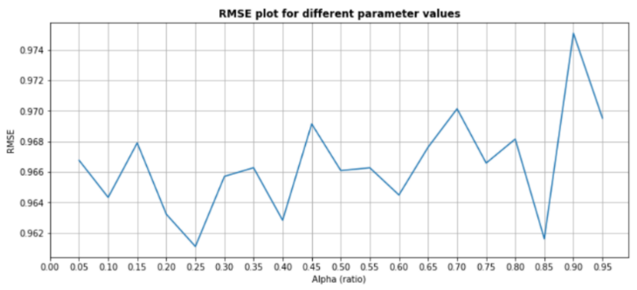

Fig. 3: Plot of RMSE over validation set for different values of alpha

For alpha = 0.25, we got minimum RMSE value = 0.9611 as shown above. For this value of alpha, we found the test set RMSE value = 0.9409.

The test set RMSE is very close to the validation set RMSE which indicates that the model performs well to anonymous users too that have not been considered while estimating parameter alpha.

V. Future Work

In future, we plan to use Doc2Vec similarity scores for features that represent contents of movies like overview. We would generate similarity matrix for such feature's group using the Doc2Vec similarity scores.

We would also develop a recommender model using collaborative filtering technique to predict user's rating of movie i using a weighted sum of movie i's rating from the k nearest users based on their ratings' similarity score.

References

[1] A. Tuzhilin and G. Adomavicius. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge & Data Engineering, vol. 17, no.6, pp. 734-749, 2005.

[2] G. Salton. Automatic Text Processing. Addison-Wesley (1989)

[3] https://radimrehurek.com/gensim/models/doc2vec.html

[4] https://www.kaggle.com/rounakbanik/the-movies-dataset