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 also 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. Proposed Methodology

1. Feature Extraction

Features selected for finding similarity between movies include actor names, director names, genres, keywords, country and overview. A combination of these features would be used to analyze and arrive at the best performing model.

2. Method 1: Item-based Collaborative Filtering

In item-based collaborative filtering, the similarities between different items in the training dataset are calculated using cosine similarity measure, and then these similarity values are used to predict ratings for the user-item pairs that are not present in the training dataset.

The method 1 is executed in three phases as described below:

Phase 1: Baseline model

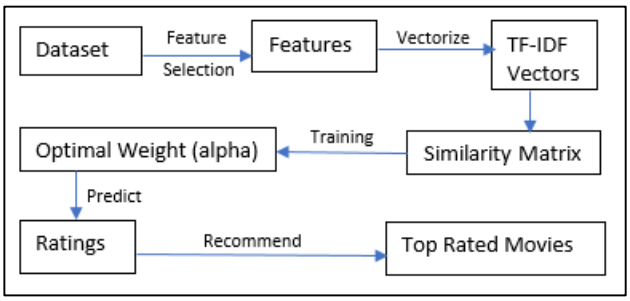

Fig. 1: Pipeline followed in Phase 1

A. Creating two feature spaces

Movie descriptions (overview) are usually long, whereas country is single token feature. Also, genres, actors and directors are on average 3 to 4 tokens long. Combining these features into a single feature vector would overweight the overview feature and underweight the remaining features. Thus, we formed two feature groups – one containing just overview, called F2, and another containing the remaining features (actors, director, country, genres, keyterms), called F1.

B. Pre-processing

Punctuations and spaces are removed from actor and director names individually. All the feature values are lowered (lower alphabetical order). Space between tokens of each actor/ director name is trimmed to make it a single token.

C. Vector Generation

For each movie, we generated two feature vectors; one feature vector corresponding to each feature space (F1 and F2). A feature vector contains Tf-Idf values for the tokens present in the corresponding features of the movie.

D. Similarity Matrix Generation & Rating Prediction

For vector space corresponding to each feature 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(i,j) between movie i and movie j is obtained as :

$$Sim(i,j) = alpha * Sim_1(i,j) + (1 - alpha) * Sim_2(i,j)$$

$Sim_1(i, j)$: value of cell (i, j) in Similarity Matrix for F1
$Sim_2(i, j)$: value of cell (i, j) in Similarity Matrix for F2
alpha: arbitrary weight in the range [0,1]

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

E. Weight (alpha) estimation and RMSE

The user ratings set is split into training and testing set. Training set consists of 5100 users and testing set consists of 900 users. For each user, his ratings are split in 80:20 ratio, where 80% of his ratings are used for making predictions over the remaining 20% movie ratings. The difference between predicted rating and true rating is taken as error, and taking Root Mean Square Error (RMSE) over the 20% movie ratings of the 5100 training users, we get Training RMSE. Training RMSE is calculated for different values of alpha ranging from 0.05 to 1, in successions of 0.05. The alpha corresponding to minimum Training RMSE is selected as the optimal value of weight parameter alpha. Over the estimated value of alpha, Test RMSE is obtained in the same manner over the 900 Test users, by making predictions over 20% of their movie ratings and finding their root mean square error.

RMSE is calculated as follows:

$$RMSE = \sqrt{\sum_{i=1}^{N} (pred_i - true_i)^2 / N}$$

where $pred_i$: predicted value of the i^{th} sample
$true_i$: true value of the i^{th} sample
N : total number of samples

Phase 2: Feature Selection

The F1 feature space consists of 5 features, all of which may not be significant with respect to automatically predicting movie rating. Thus, in order to remove such irrelevant or redundant features from F1, forward sequential Wrapper method is applied over that feature group. F2 feature space consists of just one feature i.e. overview, thus there is no scope of feature selection in it. While performing wrapper method over F1, different feature combinations of F1 are tested by following the same item-based collaborative filtering pipeline followed in phase 1. For every feature combination of F1, the F2 feature space is kept constant while testing, thus variations in results are only due to variations in F1 feature combinations.

Phase 3: Dimensionality Reduction

The tf-idf vectors of a movie, for both feature spaces, are very sparse. The dimensions for both feature vectors are more than 22k, whereas a movie on average consists of 20 to 25 tokens, each for F1 (all 5 features combined) as well as F2 features. Thus, it is necessary to examine the behavior of movie vectors at lower dimensions.

To reduce dimensionality of feature space, Principal Component Analysis (PCA) technique is used. We cannot use techniques like Linear Discriminant Analysis (LDA) here as they require class labels for the data, so as to project it in such dimensions so as to make them more class wise separable. But, our problem is more closely related to regression (predicting decimal values/ ratings) than classification. Thus, PCA is more suitable for dimensionality reduction in solving our problem statement.

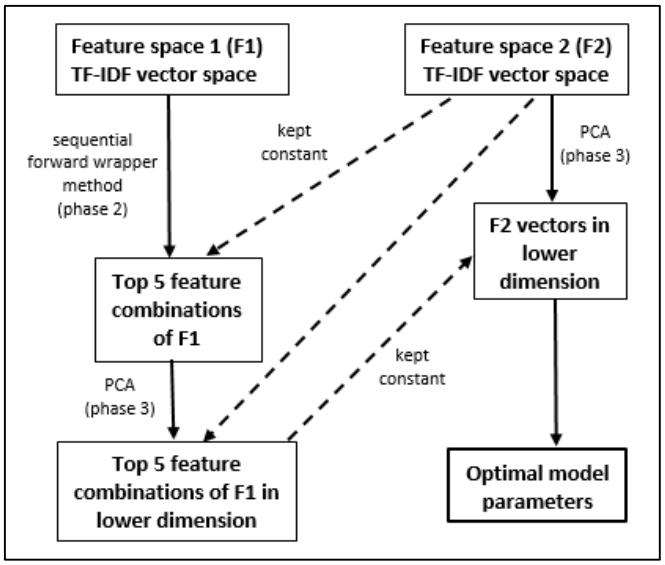

Fig. 2: Pipeline followed in Phase 2 and 3

The top 5 feature combinations in F1 feature space, having minimum Test RMSE, are selected based on the results of Wrapper method that are obtained in Phase 2. We need to test their performance in lower dimensions. PCA is performed multiple times on each of the 5 feature combinations, preserving eigen-energy varying from 50% to 99%, with increase of 5% in successions. The pipeline of Phase 1 is to be followed for each feature combination and for each of its reduced dimension, so as to get its Test RMSE at that dimension. While testing performance for all the 5 F1 feature combinations in lower dimensions, the same tf-idf based vector space for F2 feature space is used as that was used in phase 2. This makes sure that the variations in results (Test RMSE) are either due to reduced dimension or different feature combination of F1.

The F1 feature combination among the five selected, giving minimum Test RMSE in reduced dimension is selected. Also, its vector space corresponding to that reduced dimension which is giving minimum Test RMSE is to be used for further evaluating performance of F2 in reduced dimension.

Similar to F1 vectors, the tf-idf based vectors corresponding to F2 feature space are also very sparse. Thus, we need to test their performance in lower dimensions too. By keeping the optimal vector space corresponding to F1, obtained above, as fixed, PCA is performed multiple times on F2 tf-idf vectors preserving eigen-energy varying from 50% to 99%, with increase of 5% in successions. As a result, we get the best performing (min Test RMSE) vector space for F2 in reduced dimension, with respect to the optimal vector space of F1 kept fixed.

Overall, this phase gives us the lower dimensions corresponding to both F1 and F2 at which they give minimum error and the corresponding value of parameter alpha for obtaining similarity score using the similarity matrices formed by using the F1 and F2 lower dimension vector spaces.

3. Method 3: K-Nearest Neighbor (KNN)

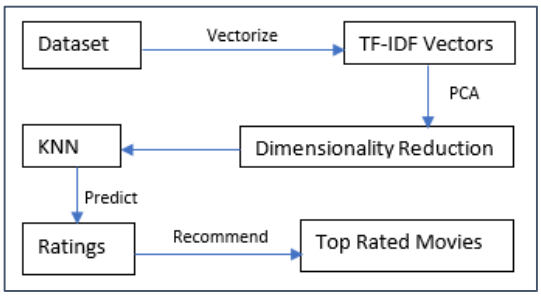

Fig. 3: Pipeline followed in Method 2

A. Feature space

The KNN algorithm uses Eucledian distance to measure similarity between movies. The features which are of low dimensions (like country, genres, directors, etc.) usually have higher tf-idf values than for feature of high dimension (overview). Thus, though the low dimension features have lesser number of values to contribute in measuring distance, but as their (tf-idf) values are relatively larger, thus their overall contribution in distance is similar to that of high dimension feature (overview). Thus, we need not generate separate feature spaces for low and high dimension features, and can perform movie rating prediction using KNN over vectors of single feature space having all the 6 features combined.

B. Pre-processing and Vector generation

The features are pre-processed same as that in method1. The tf-idf based feature vector is obtained for each movie over all the 6 features. Each movie vector is length normalized so the number of tokens in a movie's data doesn't affect the similarity value/ distance from another movie.

C. KNN without dimensionality reduction

The only parameter to be tuned here is K. As there is no weight parameter of features, like alpha, there is no need of division of users as training and testing sets. The KNN algorithm is run individually for each of the 6000 users. For each user, 80 % of his ratings are taken as training set, remaining 20% forms the test set for which ratings are to be predicted.

The difference between predicted rating and true rating, given by respective user, is considered as error. This way, RMSE is calculated over 20% of the ratings for all the users in the user set. This gives us the Test RMSE. The KNN algorithm is run for multiple values of K, varying from 1 to 4, and obtaining Test RMSE for each K. In our user dataset, a considered user has rated minimum 5 movies. Thus, as we are doing 80:20 train-test split, thus training set can have minimum 4 movies. So, we kept the range of K is limited to 1 to 4.

D. KNN with dimensionality reduction

The tf-idf vectors of movies are very sparse. Thus, we need to test their performance after dimensionality reduction too. PCA is performed multiple times for each value of K = 1 to 4, preserving eigen-energy varying from 50% to 99%, with increase of 5% in successions. As overall result in method 2, we get the reduced dimension as well as the value of K for which we get minimum prediction error (Test RMSE).

V. Results and Inferences

Method 1: Item-based Collaborative Filtering

Phase 1: Baseline Model

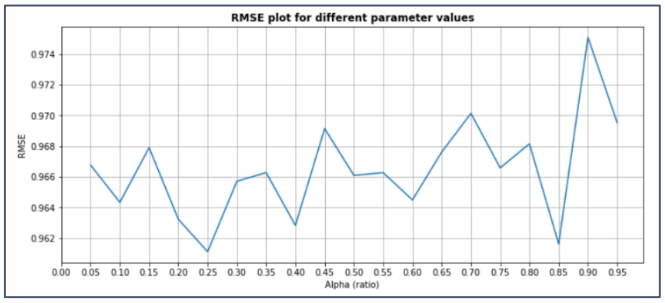

Fig. 4: Plot of RMSE over test set for different values of alpha

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.

Phase 2: Feature Selection

Best feature combination for F1: {Country, Genre, Actor, Keyterms}. Best value of alpha for this feature combination is 0.60.

F1 Features	Alpha	Train RMSE	Test RMSE
Actor	0.90	0.9622	0.9531
Director	0.45	0.9609	0.9342
Keyterms	0.20	0.9663	0.9542
Genre	0.15	0.9623	0.9408
Country	0.15	0.9654	0.9407
Country, Actor	0.30	0.9637	0.9486
Country, Director	0.30	0.9655	0.9501
Country, Keyterms	0.30	0.9636	0.9529
Country, Genre	0.40	0.9633	0.9346
Country, Genre, Actor	0.30	0.9635	0.9437
Country,Genre, Director	0.70	0.9643	0.9536
Country,Genre,Keyterms	0.30	0.9623	0.9542
Country,Genre,Actor, Director	0.20	0.9653	0.9400
Country,Genre,Actor, Keyterms	**0.60**	**0.9601**	**0.9314**
Country,Genre,Actor, Keyterms, Director	0.60	0.9628	0.9476

Table 1: Results for forward sequential wrapper method

It is observed that usually lesser Test RMSE is obtained for values of alpha ranging between 0.25 and 0.75 than border cases (alpha value close to o or 1). The reason being that at border cases, either only F1 (for alpha = 1) or F2 (for alpha = 0) contribute in overall similarity score. Thus, contribution of other feature space remains absent, which increases error in prediction.

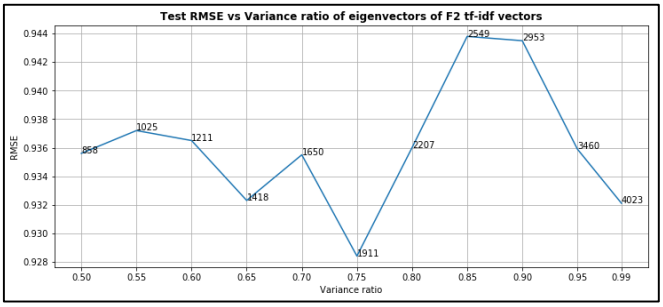

Fig. 5: Test RMSE vs Variance ratio plot for best feature combination of F1 after wrapper method. The value against variance ratios in the graph denote dimension of F1

At variance ratio of 0.75, we get minimum Test RMSE i.e. close to 0.928. This shows that data performs slightly better at lower dimension for same feature combination of F1.

Phase 3: Dimensionality Reduction

Out of the 5 feature combination for F1 feature space selected from Wrapper method, after PCA minimum RMSE is obtained for feature combination: {Country}.

Minimum Test RMSE for the selected feature combination of F1 is 0.961, which is obtained at alpha equal to 0.40.

F1 Features	Variance Ratio	F1 Dim	Alpha	Test RMSE
Country	**0.70**	**5**	**0.40**	**0.9241**
Country, Genre	0.99	71	0.35	0.9345
Country, Genre, Actor	0.90	3328	0.60	0.9308
Country, Genre, Actor, Director	0.60	1817	0.70	0.9280
Country, Genre, Actor, Keyterms	0.60	1623	0.70	0.9276

Table 2: Results of PCA over selected feature combinations

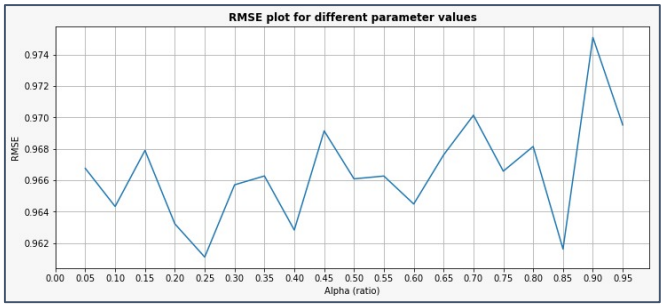

Fig. 6: Test RMSE vs alpha plot for best feature combination of F1 after PCA

Method 2: KNN

K	Feature Dim	Test RMSE
1	**52257**	**1.3336**
2	52257	1.3780
3	52257	1.4170
4	52257	1.3586

Table 3: Results of kNN without dimensionality reduction

K = 1 gives best performance and k = 4 gives second-best performance in KNN for lower as well as higher dimension feature vectors. The variation of Test RMSE with varying variance ratio is shown below:

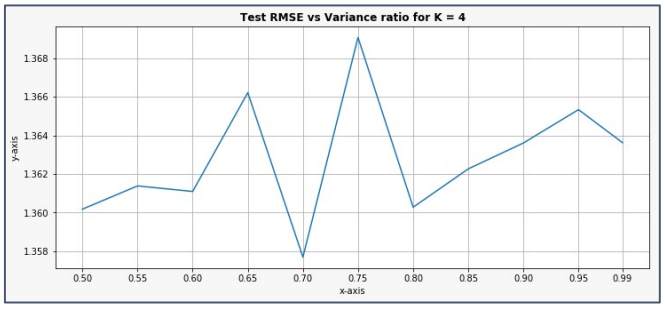

Fig. 7: Plot of test RMSE vs variance ratio in kNN for k = 4

VI. Conclusion

Results show that country plays the most significant role among all the 6 features in rating a movie. Among all the feature combinations in F1 feature space selected using wrapper method, country was included in all the 5 combinations. Genre and actor taken together play the second-most significant role.

In method 1, PCA helps in reducing test RMSE slightly (by around 0.01). The performance of Item-based collaborative filtering is always found to be better than KNN, irrespective of chosen value of K.

In KNN, the relative order of performance for different values of K is found to remain same for all variations in eigen-energy irrespective of dimensionality reduction by PCA.

VII. 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.

VIII. 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