Chapter 4

Diversification in Recommendation System: Cluster-based Machine learning approaches

4.1 Introduction

Recommendation systems have become integral to our online experiences, helping us discover new products, movies, music, and other content that align with our preferences. However, traditional recommendation systems often need more diversity, reinforcing popular or mainstream choices and limiting exposure to new or niche options ([134,135]). Recommending similar items based on previous user interactions will make the recommendation too obvious. Introducing diversity is a way out of this problem, which was introduced in 2001 by Bradley and Smyth ([136]). Diversity in recommendation systems refers to including a wide range of items in the recommendation list provided to users. It aims to overcome biases and uniformity by offering users a more diverse and broad item selection based on their past preferences([137–139]). Various approaches and techniques have been developed to achieve diversity in recommendation systems. These include content-based filtering, collaborative filtering, hybrid

methods that combine multiple techniques, serendipity and novelty techniques, fairness and inclusivity considerations, user control and transparency features, and appropriate evaluation metrics to measure diversity. A number of research works were motivated by the work of Bradley and Smyth to introduce different algorithms for the diversification of results in a recommender system ([136, 140, 141]). Most methods that aim to improve recommendation diversity consider diversity optimization as a post-processing step regardless of the recommendation generation model. At first, recommendations are generated for each user. The list is then shortened and modified with some re-ranking and intent-based methods that consider the user's intent and item diversity. ([140, 142, 143]).

4.2 Problem Description

Let us have a set of users $U = \{u_1, u_2, u_3 \dots u_m\}$ and a set of items $I = \{i_1, i_2, i_3 \dots i_n\}$ and we have set of item features $F = \{f_1, f_2, f_3 \dots f_k\}$. Therefore we have a user-item interaction matrix R of size $m \times n$ and an item-feature matrix of size $(n \times k)$. In the interaction matrix, each user has rated some items, the rating for item i by user u is r_{ui}, and the rating value ranges between integer value 1 to 5 (5 being the best and 1 the worst). The item feature matrix S is a binary matrix, and the entry (s_{ik}) is either 1 if item i has feature k else 0 (item I does not have feature k).

Given the interaction matrix $R_{m \times n}$, item-feature matrix $S_{n \times k}$, we generate user-feature matrix $(G_{m \times k})$. For a set of target users, the goal of our proposed model is to predict the ratings of all the non-rated items for the target users and recommend top-N items for each of the target users.

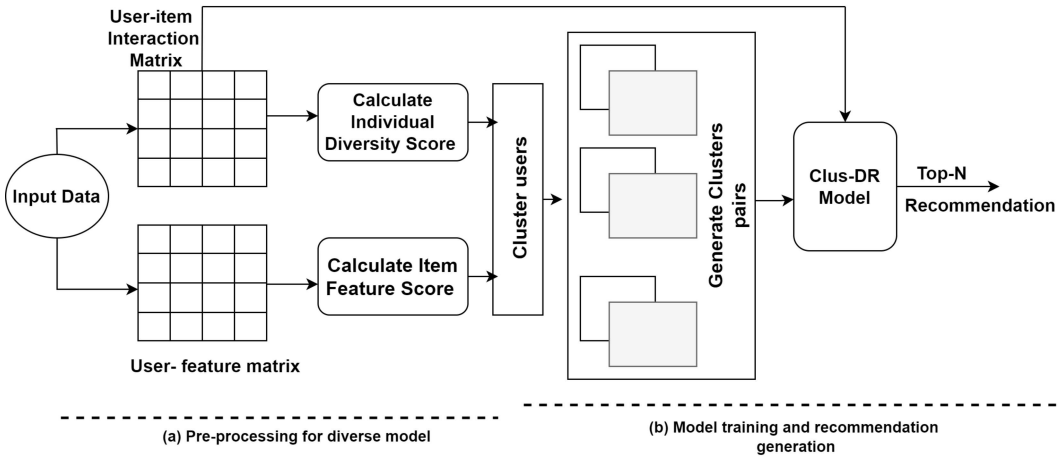


Figure 4.1: Architecture of the proposed Clus-DR recommendation model

4.3 Proposed Methodology

This section presents the overall architecture and operational methodology of our proposed Clus-DR model (Figure 4.1). We use two inputs: a user-item interaction matrix ($R_{m\times n}$), and an item-feature matrix ($S_{n\times k}$). While $R_{m\times n}$ can be readily used, $S_{n\times k}$ needs to be processed to transform into a user-feature matrix. The workflow of our proposed Clus-DR model can be divided into three parts. First, individual diversity calculation and item feature score calculation for the user; second, clustering of users based on item features and individual user's diversity score; and third, training our model using Non-Negative Matrix Factorization (NNMF) technique to generate a diverse recommendation. Apart from NNMF, we also evaluate our model performance for SVD and SVDpp algorithms. Our goal is to exhibit the personalized diversity that should be included in the recommendation system before rating prediction. The flow chart of the proposed Clus-DR model is shown in Figure 4.2. The overall architecture of our proposed methodology is divided into two parts online and offline part. The online part contains input where we calculate IDS and IFS scores. These scores will update whenever a new user and item are introduced in the system, and in the offline part, we train this input information for recommendation generation.

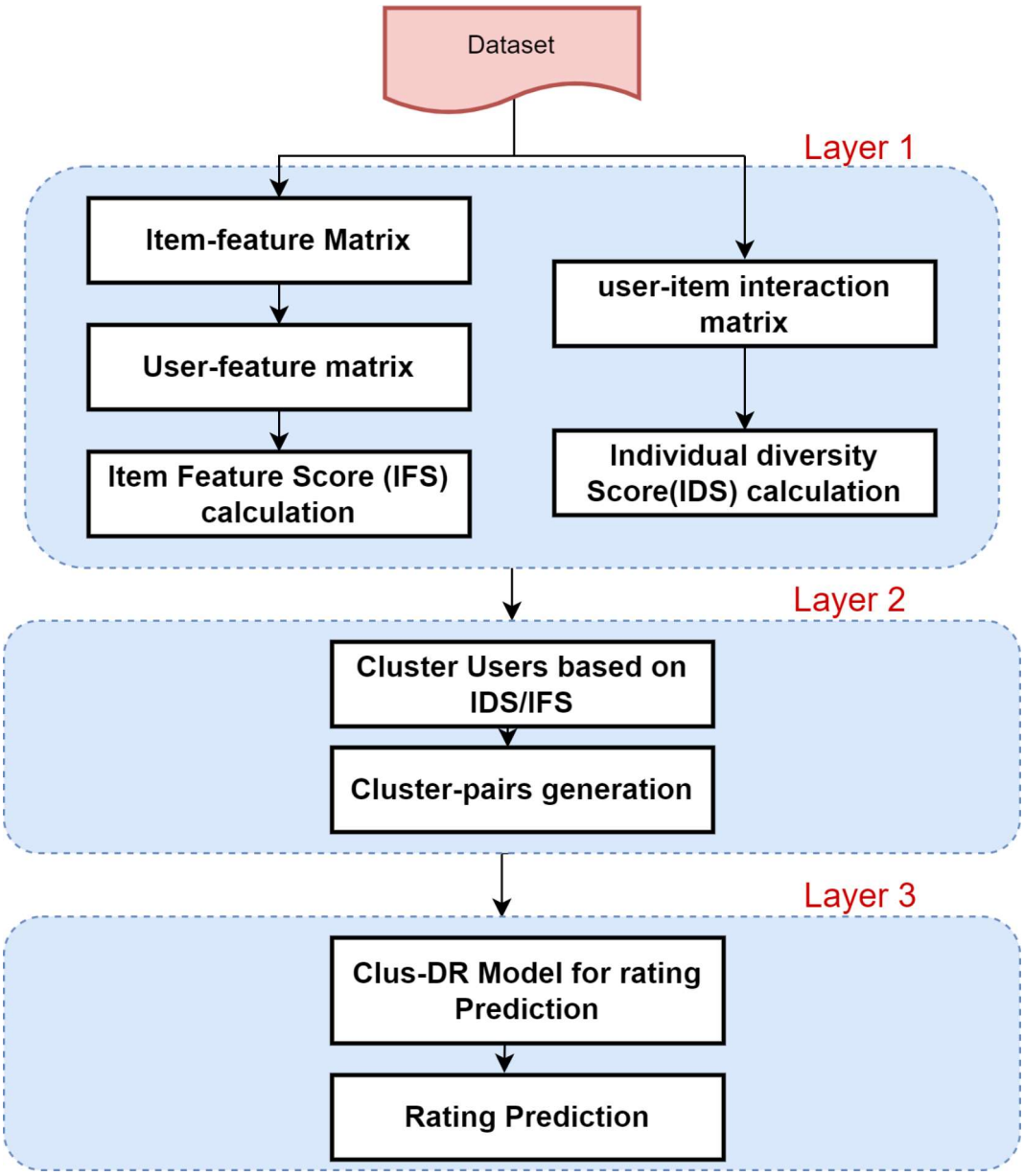


Figure 4.2: Flow-chart of the proposed Clus-DR recommendation model

4.3.1 Individual Diversity Score for User

We describe here an attempt to quantify the user's need for diversity. Every user has different preferences in their daily life. For example, in the movie domain users have different choices in terms of artists, genres etc. We factor in these user-specific preferences for diversity into our model and introduce an Individual Diversity Score (IDS) for users, representing how a user liked diversity in her past preferences. This diversity score of a user is used as the minimum threshold for diversification of our recommendation. We define the diversity in terms of cosine similarity. The similarity between items will be calculated using the equation (4.2). The average dissimilarity between items rated by user u is considered as the individual diversity score for the user u. In equation (4.1), diversity is calculated for the user u, where i_j and i_k are two different items rated by user u from their past interacted item list of size n.

$$\text{IDS(u)} = \frac{1}{n(n-1)} \sum_{j=1}^{n} \sum_{\substack{k=1 \\ (j \neq k)}}^{n} (1 - sim(i_j, i_k)) \tag{4.1}$$

$$\text{Similarity}(i_j, i_k) = \frac{\sum_{p=1}^{m} r_{pj}.r_{pk}}{\sqrt{\sum_{p=1}^{m}(r_{pj})^2 . \sum_{p=1}^{m}(r_{pk})^2}} \tag{4.2}$$

In equation (4.2) we calculate a cosine similarity between two items i_j and i_k rated by user u where r_{pj} and r_{pk} are the ratings for item i_j and i_k by user p and m is the number of users present in the dataset. IDS(u) is calculated for each user and is termed as individual diversity score for user u. We consider IDS(u) as a threshold for the user u, which decides the user u's minimum diversity level.

4.3.2 Item Feature Score for User

In users' item selection, item features information also plays a significant role. Users often select items based on their content information like genre, year, artist etc. So we argue that item feature information will also be a decisive criterion in diversity

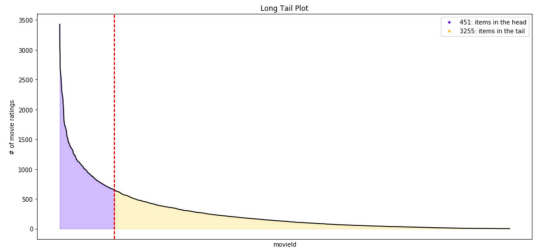


Figure 4.3: The Long-tail of the popularity of items in the MovieLens dataset

enhancement in the recommendation system. We propose an item feature score for each user based on this assumption. Here in our proposed algorithm, we use genre information of movies and music and tags information of books to calculate a feature score for each user, which defines each user's preference for a particularized genre. For generalization apart from genre score, we can also use other content information of items like in music domain, artists, genre, language, etc. We use MovieLens, LastFM, and Goodbooks datasets, where each movie has at least two genres, and in LastFM and Goodbooks, we use the tag information of each artist to calculate the feature score. The tags are the user-defined tags/shelves/genres in each artist/book dataset. In other terms, we encode the item's metadata information for the clustering of users.

Item Feature Score(u) is calculated for each user based on a portion of particular user interaction information. Let user u have rated n_1 items from a complete set of n items ($n_1 <= n$). The calculation Item Feature Score(u) is calculated as follows:

$$\text{Item Feature Score}(u) = \frac{\sum_{j=1}^{n_1} \sum_{k=1}^{Q} G(i_j, f_k)}{\sum_{j=1}^{n} \sum_{k=1}^{Q} G(i_j, f_k)} \tag{4.3}$$

Here in equation (4.3) we calculate Item Feature Score(u) for user u. where function $G(i_j, f_k)$ is a boolean function defined as follows:

$$G(i_j, f_k) = \begin{cases} 1, & \text{if } i_j \text{ has feature } f_k \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

and $f_1, f_2, \ldots f_Q$ represents a set of Q distinct features (genre) and $j \in \{1, \ldots, n\}$ where n is total number of items present in the dataset.

4.3.3 Clustering Algorithm

The second step of our proposed algorithm is the clustering of users based on IDS and IFS. In our proposed algorithm, clustering is performed using two different techniques:

1. Clustering is first performed using IFS
2. Clustering is also performed using IDS.

The flexibility of our approach is that we can use any feature information apart from genre value and use any unsupervised clustering algorithm. First, clustering is performed using the feature information of an item, i.e., the genre information of the movie and second, clustering is performed using the individual diversity of a user. The agenda for both algorithms is different. The inclusion of genre scores will carry items that have similar genre-ratio information. Similarly, the other clustering will cluster users with the same diversity level. We use a k-means clustering algorithm. However, any unsupervised algorithm can also be used instead. The k-means algorithm in our approach works as follows.

- Define the number of clusters as a total number of distinct genres present in the data set.
- Selecting a random set of k data points and then calculating the centroid without data shuffling.
- Keep iterating until the centroid value stops changing. It also means that the data value assigned to the cluster is not changed.
- Compute Euclidean distance from one data point to other data points and assign data points to each cluster with which they have the minimum distance and then compute the centroid.

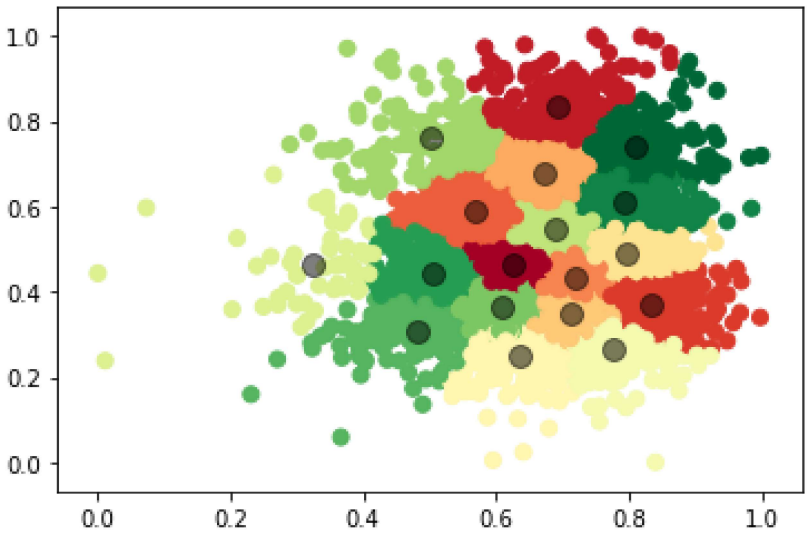


Figure 4.4: t-SNE plot for k-means Clustering of a user based on item feature score

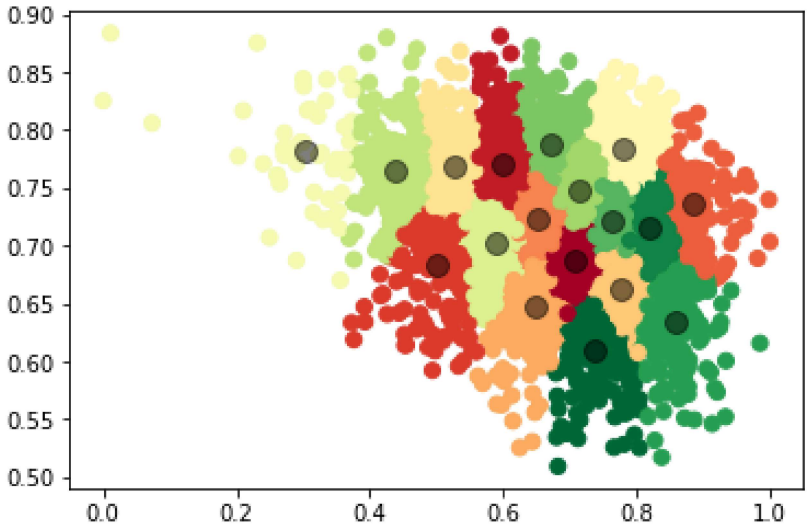


Figure 4.5: t-SNE plot for k-means Clustering of a user based on individual diversity scores

4.3.4 Aggregate Cluster Diversity

Next, in our proposed algorithm, we need to calculate aggregate cluster diversity. The aggregate cluster diversity is the average of each user's individual diversity present in the cluster. The urge behind the calculation of aggregate cluster diversity is that it includes users with similar diversity levels. The further use of aggregate cluster diversity is to make cluster pairs for the training of our model. The intention behind training our model with these clusters having more diverse users will help other less diverse users so that the interaction domain of these users will expand, and the model will generate more diverse recommendations. The pseudo-code for aggregate cluster diversity is described in Algorithm 2.

Algorithm 2: Pseudo-code for Aggregate Diversity for a Cluster

```
Input User set U ∈ (u_1, u_2, ... u_m) Users in Cluster (u_{c_k}), Total Cluster(C_k),
Individual diversity of user u (Div_u) Output ADivClus ← [] ;      ▷ aggregate
cluster diversity
for C_k = 1, 2, ... k do
    for each u_c ∈ U do
        if u_c ∈ C_k then
            ADivClus ← Div(u_c)
        end
    end
    return ADivClus(C_k)
end
```

The aggregate diversity for the ML-1M dataset is calculated after applying the k-means clustering algorithm for cluster size (k)=18. The optimal value for cluster size(k) is chosen from a set of experiments performed from various k values, the detailed analysis is present in section 4.5. The aggregate cluster diversity for clusters is shown in Table 4.1, 4.2. The geographical representation of clusters into 2-Dimensional t-SNE (t-Distributed Stochastic Neighbor Embedding) plots are represented in Figures 4.4, 4.5.

Table 4.1: Aggregate Diversity of Clusters based on Genre-score Clustering for ML-1M dataset

Cluster_ID	Agg-Diversity	Cluster_ID	Agg-Diversity
1	0.9934	7	0.9535
17	0.9905	4	0.9277
12	0.9838	0	0.9231
5	0.9810	2	0.9221
16	0.9741	6	0.9101
9	0.9717	10	0.9087
11	0.9580	13	0.8899
3	0.9569	8	0.8758
15	0.8750	14	0.8581

Table 4.2: Aggregate Diversity of Clusters based on user's Individual Diversity Score clustering for ML-1M dataset

Cluster_ID	Agg-Diversity	Cluster_ID	Agg-Diversity
17	0.9956	14	0.9702
15	0.9928	4	0.9603
5	0.9905	12	0.9381
6	0.9898	9	0.9046
0	0.9808	1	0.9001
2	0.9799	8	0.8948
16	0.9758	7	0.8814
3	0.9748	11	0.8604
10	0.9708	13	0.8582

4.3.5 Clus-DR Model

This section includes the details of our proposed model Clus-DR. In our proposed model we tried to train our model for diverse item recommendation generation. In pre-processing, we calculated IDS and IFS score, which defines each user's diversity score. Furthermore, we group users, according to their ratings, who share comparable preferences for diversity in their prior contact history. Making cluster pairs for our model's training involves using aggregate cluster diversity as a piece of additional information. The purpose of training our model with these clusters of more diverse users is to assist other less diverse users, thereby expanding their interaction domain and allowing the model to produce more diverse recommendations. Training these cluster pairs involves various matrix-factorization based models to generate a final recommendation. The proposed Clus-DR model generates the top-k prediction for each user, which are more diverse than the other state-of-the-art recommendation algorithms.

The architecture of the proposed approach is shown in Figure 4.1. We start by describing the data we used and its pre-processing procedure. The pre-processing for the Clus-DR model involves calculating the diversity feature score for each user, where we first calculate the individual diversity score and item feature score for each user. This score is further used to cluster users into groups based on the assumption that more diverse users will be in the same group. The user clusters $C_i = (C_1, C_2, \ldots C_k)$ are obtained by using any unsupervised clustering algorithm. In our proposed architecture. We utilise the k-means clustering algorithm, although any clustering algorithm may be used for generating user clusters. We first utilise the ILD score to cluster users, and then we train our model using SVD, SVDpp, and a non-negative matrix factorization (NNMF) method. Our method considers NMF with a restriction that specifies extra data in addition to each user's user-item input matrix and cluster data. The user clustering procedure is described in section 4.3.3. Further, these clusters are utilized as a training pair for the Clus-DR model. We used standard matrix factorization algorithms

(SVD, SVDpp and NNMF) to generate diverse recommendations. Matrix factorization is a popular method used in recommendation systems to predict and recommend items to users based on their preferences and behaviour. It works by decomposing a user-item interaction matrix into two lower-dimensional matrices. Matrix factorization addresses this sparsity by approximating the original matrix with two smaller matrices, often called the user matrix and the item matrix. These matrices are of lower dimensions, which helps capture the underlying factors or latent features that influence user-item interactions. Matrix factorization can be performed in the recommendation system using various optimization techniques, such as gradient descent or singular value decomposition. These methods iteratively adjust the user and item matrices to minimize the difference between the predicted and actual ratings in the original matrix.

We used typical matrix factorization algorithms (SVD, SVDpp and NNMF) to generate diverse recommendations. NNMF imposes a non-negativity constraint on the user and item matrices during the factorization process. This means that all elements in the matrices must be non-negative. This constraint is helpful in scenarios where the extracted factors should only have positive values, such as in document-topic modelling or image processing. SVD does not have this constraint and can work with both positive and negative values. A standard NNMF is used as a dimensionality reduction algorithm which can be written as $R \approx U.V$ where U and V are two matrices having non-negative values ([144]). In recommendation system, U and V are the column matrices used for user and item representation. In general, matrix R contains some unknown values, which are the values of user-item rating information which is not available in the dataset. The data is represented in a user-item interaction matrix $R \in \mathbb{R}^{m \times n}$. Our goal is to represent it by the product of two low dimensional matrices where $U \in \mathbb{R}^{m \times d}$ and $V \in \mathbb{R}^{d \times n}$. The NNMF problem is $\min_{U \geq 0 V \geq 0} \|R - (U.V)\|$. The objective of the cost function is to minimize the error between the original rating matrix R and the product of the latent vectors of user U and item V. The cost function for NNMF often depends

on the probability distribution of data, so the simple way is to use the Frobenius-norm measure. The NNMF cost function for error minimization is defined as follows:

$$D_F(R||UV^T) = \left\|R - UV^T\right\|_F^2 = \sum_{u,v}(R_{uv} - \sum_{d=1}^{k} U_{uf}.V_{fv}) \tag{4.5}$$

In equation (4.5) we calculated the error for NNMF algorithm where the error is nothing but the l_2 norm of the matrix of size $(M \times N)$ which is a difference matrix obtained from the original matrix R and the resultant matrix of $U.V^T$ where U and V are the latent vectors of the users and items, respectively. Here M is the total number of users and N is the total items in the dataset, and R_{uv} is the original rating of item v given by the user u. The above cost functions are convex concerning either the entries of the matrix U or the matrix V, but not both. Therefore, it is impossible to solve this problem in the sense of finding a global minimum. However, many numerical optimization methods can be applied to discover local minimums, so we adopted gradient descent for faster convergence. The update rule for the user (U) and item (V) is derived using gradient descent. Gradient descent for multi-variable function $F(\alpha)$ is defined and differentiable in a neighborhood of a point A. So it follows rule $\beta \leftarrow \alpha - \eta\nabla f(\alpha)$ for very small learning rate (η) then $f(\alpha) \geq f(\beta)$. Now the update rules for U and V that reduce the divergence are as follows:

$$U_{uf} = U_{uf} + \eta_{uf}\left[\sum_{m=1}^{M} V_{fm}^T \frac{R_{um}}{(U.V^T)_{um}} - \sum_{m=1}^{M} V_{fm}^T\right] \tag{4.6}$$

$$V_{fv} = V_{fv} + \eta_{fv}\left[\sum_{n=1}^{N} U_{nf}^T \frac{R_{nu}}{(U.V^T)_{nu}} - \sum_{n=1}^{N} U_{nf}^T\right] \tag{4.7}$$

These rules are used to calculate the user and item latent factor. The update rule for user U_{uf} is defined in equation (4.6) and the update for item V_{fv} is defined in equation (4.7), where η_{uf} and η_{fv} are the learning rate. We have used divergence based cost function to minimize the difference between the original rating matrix R and the

matrix obtained from user and item latent factors. In equation (4.6), (4.7) we have set the value of the learning rate as follows:

$$\eta_{uf} = U_{uf} / \sum_{\eta=1}^{M} {V_{f\eta}}^{T} \tag{4.8}$$

$$\eta_{fi} = V_{fv} / \sum_{\eta=1}^{N} {U_{\eta f}}^{T} \tag{4.9}$$

Still, the main aim is to increase the diversity of the algorithm, so training of the algorithm is the core part that induces diversity, which we discuss below in the section 4.3.6.

4.3.6 Clus-DR Model Training and Recommendation Generation

In this section, we define our training procedure for the Clus-DR model. We use movie, book and music domain data to train our model. We first calculate the user-item interaction matrix, and then we calculate individual diversity and item feature scores for each user discussed in section 4.3.1 and 4.3.2, respectively. After that, we apply the k-means algorithm for user clustering based on individual diversity and item feature scores. The assumption over the clustering algorithm using individual diversity score is to group users following the same diversity levels. The speculation over using the clustering of item feature scores is that users following the same item features will be in one group. At a time, we use only one score for clustering to check which score performs better in diversity. Another essential motive is to check whether content information is necessary or not for diversification in a recommendation.

Once we get the clusters of users, we calculate aggregate diversity for each cluster defined in algorithm 2. The overall goal of the aggregate cluster diversity is used as a threshold value for training pair generation for users based on their diversity level. The aggregate value for each cluster is nothing but an average of the individual diversity score of each user present in the cluster. Suppose there are three clusters C_1, C_2 and

C_3 and their aggregate diversity are $0.988, 0.899$ and 0.901, then training pairs are $(C_1, C_2), (C_2, C_3)$ and (C_1, C_3). So the input for our Clus-DR model training are the cluster pairs and their corresponding user-item interaction pairs which will be processed according to the model description discussed in section 4.3.5.

For model training we need to define some hyper-parameter for model training. The first is the k value, which is number of clusters. We set $k = 18$ for our model and also check our results for $k = 10$. Another hyper-parameter the is learning rate $\eta = 0.01$. After cluster pair generation and hyper-parameter setting, we train our model for three prediction algorithms. First, we use *(Clus-NMF)* algorithm to train our model, apart from Clus-NMF, we also use SVD *(Clus-SVD)*, and SVDpp *(Clus-SVDpp)* algorithm for our proposed model, but the result obtained from *Clus-NMF* are good enough. In the Clus-DR model, training is based on the cluster of more diverse users, which leads to sparsity. We strictly limit ourselves to only those users who belong to the same cluster or more diverse cluster for recommendation generation. This may lead to some information loss in the model and leads to sparsity. The cluster-pair formation restricts our model in search of similar users for target user recommendation generation. For explanation, unlike the traditional CF-based algorithm, while searching for similar users from the complete list of user sets, we will search similar users in clusters. So this might be the reason for information loss and leads to the accuracy drop of our proposed model.

4.4 Experiments

In this section, the experiment setup and the dataset used for the proposed algorithm are described. Later on, a performance comparison of the proposed model with the baseline recommendation model is reported. The baseline for the recommendation model is described in section 4.4.3.

4.4.1 Dataset

The recommender system is based on the purchase history of users, items and their interaction history. Based on that, it predicts the most suitable items for users. So, an essential requirement for these models is a set of user items and their feedback that describe the products. To evaluate the performance of our proposed methodology, we use ML-1M[1] and ML-100k[2], an established dataset for movie recommender systems, LastFM(2k)[3] dataset for the music domain and Goodbooks (10k)[4] dataset from the book domain. The statistics of the datasets are present in Table 5.7. 2(k) and 10(k) are abbreviations used in the dataset, indicating the number of distinct users and interactions. The LastFM(2k) is a music dataset for 2000 different users' music listening datasets. Similarly, the MovieLens (100k) is a movie dataset for 943 distinct users having 100000 interactions of users and movies in the dataset. For ML-1M and ML-100k dataset, every movie is marked with genre information. In these datasets, a total of 18 different genres are present, and every movie is described with more than one genre, e.g. the genres Fantasy and Sci-fi describe the movie Star Wars. Before using our approach for recommendation generation, the first step is to preprocess the raw dataset.

We also use a different domain dataset to evaluate our model's flexibility, and we use a music dataset LastFM. In this dataset, there is a play count value that describes each user's total number of plays for a particular artist instead of rating. As a pre-processing step, we converted the play count value into a rating scale of 1 to 5. Similarly, in the Goodbook dataset we have users, books and their corresponding rating with the author and tag information of each book. For result generation, we need to transform every preprocessed dataset into a user-item interaction matrix. This matrix is then used for recommendation generation.

[1]https://files.grouplens.org/datasets/movielens/ml-1m.zip
[2]https://files.grouplens.org/datasets/movielens/ml-100k.zip
[3]http://millionsongdataset.com/lastfm
[4]https://www.kaggle.com/zygmunt/goodbooks-10k

Table 4.3: Statistics of the datasets used for evaluation of our approach

Dataset	Users	Movies	Interaction
LastFM (2k)	1892	17632	92834
ML-1M	71567	10681	10000054
Goodbooks (10k)	53424	10000	981756
ML-100k	943	1682	100000

4.4.2 Diversity Evaluation Metrics

In this section, we discuss the evaluation metrics used for the evaluation of our proposed model. There are numerous metrics available for the evaluation of a recommendation model. We use Root Mean Square Error (RMSE) and MAE (Mean Absolute Error) to evaluate our model's accuracy. The accuracy metric for this model is not sufficient because the fundamental goal for this proposed model is to generate some set of diverse as well as accurate recommendations. Aggregate diversity is an essential feature of the diversity of recommendation systems. The aggregate diversity of a recommended list is calculated using the average dissimilarity between items present in the list. Diversity for top-n recommendations is calculated using the intralist similarity measure defined in equation (4.10). We also include user coverage results, which show the coverage of item set for each user. For further evaluation, we use accuracy measures in terms of RMSE and MAE (mean absolute error) with top-k Precision, Recall and nDCG (Normalized Discounted Cumulative Gain) which are important measures for recommender system evaluation. The aggregate Diversity of a recommended list is calculated using Intralist Diversity (IL-D) measure defined in equation (4.10). IL-D is the average dissimilarity between items present in the recommendation list size of R for each user.

$$\text{IL-D} = \frac{1}{|R|\,|R-1|} \sum_{i \in R} \sum_{j \in R, i \neq j} 1 - d(i,j) \tag{4.10}$$

For recommendation systems, system coverage is how many items appear in the recommended results or top-n recommended result. Coverage is defined as below:

Table 4.4: Comparison of error evaluation metrics for the proposed Clus-DR model in terms of RMSE (Lower the better value) and MAE (Lower the better value).

Dataset	Proposed Model	Algorithm	RMSE	MAE
ML-1M	Clus-DR (IDS)	Clus-SVD	1.1107	0.9707
		Clus-SVDpp	1.1181	0.8987
		Clus-NMF	**1.1107**	**0.9250**
	Clus-DR (IFS)	Clus-SVD	1.1120	0.8906
		Clus-SVDpp	1.1181	0.8987
		Clus-NMF	1.1141	0.9765
ML-100k	Clus-DR (IDS)	Clus-SVD	1.0783	0.8660
		Clus-SVDpp	**1.0776**	**0.8649**
		Clus-NMF	1.1228	0.9296
	Clus-DR (IFS)	Clus-SVD	1.0991	0.8772
		Clus-SVDpp	1.0891	0.9055
		Clus-NMF	1.1196	0.8866
LastFM (2k)	Clus-DR (IDS)	Clus-SVD	1.1196	0.8866
		Clus-SVDpp	1.1228	0.8833
		Clus-NMF	**1.1008**	**0.9368**
	Clus-DR (IFS)	Clus-SVD	1.1233	0.8987
		Clus-SVDpp	1.1229	0.9033
		Clus-NMF	1.0988	0.9299
Goodbooks (10k)	Clus-DR (IDS)	Clus-SVD	1.2026	0.9885
		Clus-SVDpp	1.2118	0.9442
		Clus-NMF	**1.1803**	**0.9003**
	Clus-DR (IFS)	Clus-SVD	1.2103	0.9987
		Clus-SVDpp	1.2269	0.9893
		Clus-NMF	1.1991	0.9221

Clus-DR (IDS) and Clus-DR (IFS) are the models based on IDS score and Item Feature Score, respectively.

$$\text{Coverage}(M) = \bigcup_{u} R_{M,k}(u) \tag{4.11}$$

Coverage for model M is $R_{M,k}$ is test set R for user u, retrieved from the model M. A higher value for coverage is better and shows that the model recommends a wide range of items.

Table 4.5: Comparison of various accuracy metrics for the proposed Clus-DR model for precision (higher value is better), nDCG (higher value is better), diversity, and coverage.

Dataset	Methodology	Algorithm	Precision	nDCG	IL-D	Coverage
ML-1M	Clus-DR(IDS)	Clus-SVD	0.2831	0.2781	0.7336	53.98
		Clus-SVDpp	0.2646	0.2358	0.7703	50.33
		Clus-NMF	0.2697	0.2361	**0.8028**	**55.62**
	Clus-DR(IFS)	Clus-SVD	0.2557	0.2667	0.7674	51.41
		Clus-SVDpp	0.2203	0.2273	0.7872	52.41
		Clus-NMF	0.2332	0.2443	**0.8058**	**55.67**
ML-100k	Clus-DR(IDS)	Clus-SVD	0.3435	0.3552	0.7138	52.16
		Clus-SVDpp	0.2422	0.2398	0.7071	49.32
		Clus-NMF	0.1900	0.2039	**0.7875**	**82.67**
	Clus-DR(IFS)	Clus-SVD	0.3214	0.3364	0.6621	41.03
		Clus-SVDpp	0.2215	0.1933	0.6229	42.93
		Clus-NMF	0.1792	0.1822	**0.7003**	**48.55**
LastFM(2k)	Clus-DR(IDS)	Clus-SVD	0.2245	0.2247	0.5336	39.22
		Clus-SVDpp	0.2117	0.2338	0.5662	40.33
		Clus-NMF	0.2998	0.2445	**0.6028**	**41.22**
	Clus-DR (IFS)	Clus-SVD	0.2007	0.1998	0.5028	37.44
		Clus-SVDpp	0.2109	0.2188	0.5331	35.22
		Clus-NMF	0.2398	0.2239	**0.5773**	**39.88**
Goodbooks(10k)	Clus-DR(IDS)	Clus-SVD	0.2008	0.1947	0.4036	29.83
		Clus-SVDpp	0.1988	0.2038	0.4632	25.09
		Clus-NMF	0.2198	0.2105	**0.5088**	**38.98**
	Clus-DR(IFS)	Clus-SVD	0.1992	0.2083	0.4108	27.38
		Clus-SVDpp	0.1889	0.1798	0.3931	25.07
		Clus-NMF	0.2038	0.2089	**0.4273**	**35.88**

The diversity value ranges from 0 to 1 and higher values are better, and values for coverage are given in percentage (%). All results are for top-5 values.

4.4.3 Experimental Results and Baseline

In this Section, the results of the experiments are presented. Here, we show our model's performance for different evaluation metrics used for the recommendation model, followed by a comparative analysis with baseline algorithms used for the diversification of the recommendation system.

4.4.3.1 Performance of the Clus-DR Model

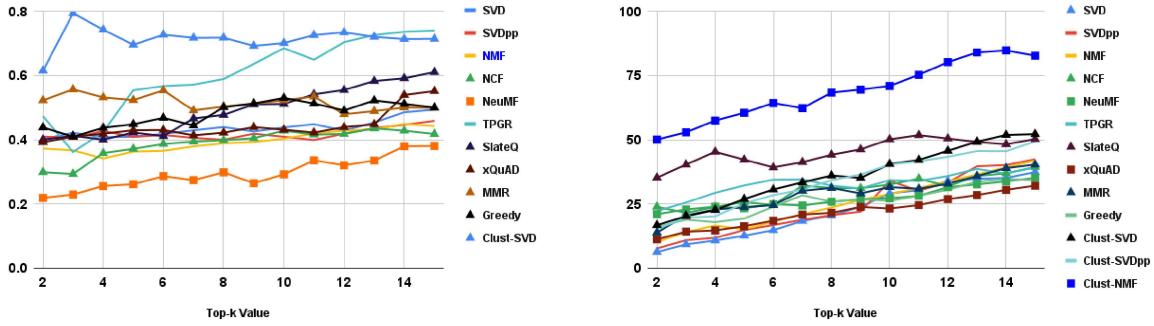
In the experiment of our Clus-DR model, we selected SVD, SVDpp, and NNMF approaches for training the model. The number of clusters (k) is chosen to be 18 with

a learning rate of 0.01. The value of cluster size was decided after we experimented with several hyper-parameters. The accuracy of the Clus-DR model is calculated using RMSE and MAE in Table 4.4. The best accuracy of our model is for ML-1M dataset, where RMSE is 1.1107, and MAE is 0.9250 for the Clus-NMF algorithm. We used three different datasets of movies, books and music domains to prove the generalization of our approach. Statistics of these datasets are shown in Table 5.7. The MovieLens dataset suffers from the long tail problem. The long-tail graph for the ML-1M dataset is shown in Figure 4.3. The long-tail graph shows the distribution of item popularity in the dataset, where the x-axis shows the items and y-axis shows the popularity of items in terms of rating provided by the users. This long-tail problem shows the uneven distribution of the rating behaviour of the user, which makes the dataset sparse.

Next, following our Clus-DR model algorithm, we need to group users based on individual diversity and item-feature score. For this, we applied a k-means clustering algorithm, and user clusters for the ML-1M dataset are depicted in Figures 4.4, 4.5. After cluster generation, we need to calculate the aggregate diversity of clusters, which is nothing but the average of the individual diversity of users present in the cluster. We created 18 clusters for each dataset, so the aggregate diversity for each cluster for ML-1M dataset are represented in Tables 4.1, 4.2. We create 18 clusters because, in this MovieLens dataset, there are 18 different genres available. The flexibility of our approach is we can use any unsupervised clustering algorithm in place of the k-means. The k-means algorithm is used because we can maximize user similarity within clusters by utilizing this algorithm and minimize the similarity of users in different clusters.

Moving forward, we need to check our model's performance in terms of accuracy and diversity. As per the accuracy-diversity trade-off, our model loses some accuracy compared with the baseline algorithms for the recommender system. For our comparative model study, we use RMSE and MAE for accuracy analysis. All the results for our Clus-DR model over various datasets are presented in Table 4.4. Table 4.4 shows

our model's accuracy measures in the context of different datasets. This main aim is to propose an approach that gives diverse recommendations for each user. Apart from using any post-processing step, we tried to train our model to generate diverse recommendations. The table shows our model's accuracy in terms of RMSE and MAE for ML-1M, Ml-100k, LastFM(2k), Goodbooks(10k) datasets. From Table 4.4, we can conclude that the ML-1M is the best-performing dataset for our Clus-DR model. For performance evaluation of the proposed Clus-DR model, we use precision, nDCG, IL-D (intralist diversity), and coverage. The complete results are shown in Table 4.5. The results discussed in Table 4.5 are for top-15 recommendations for each user.



(a) Diversity Graph

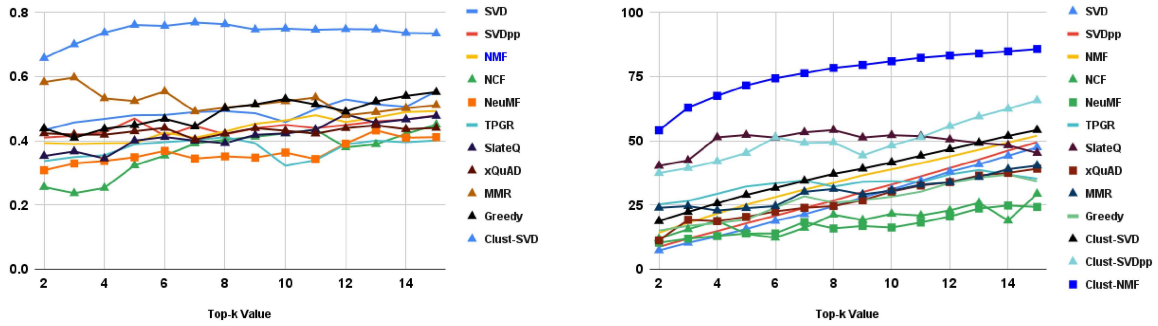
(b) Coverage Graph

Figure 4.6: A demonstration for top-k diversity and coverage analysis for ML-100k dataset for baseline and our proposed Clus-DR model

4.4.3.2 Comparative Analysis

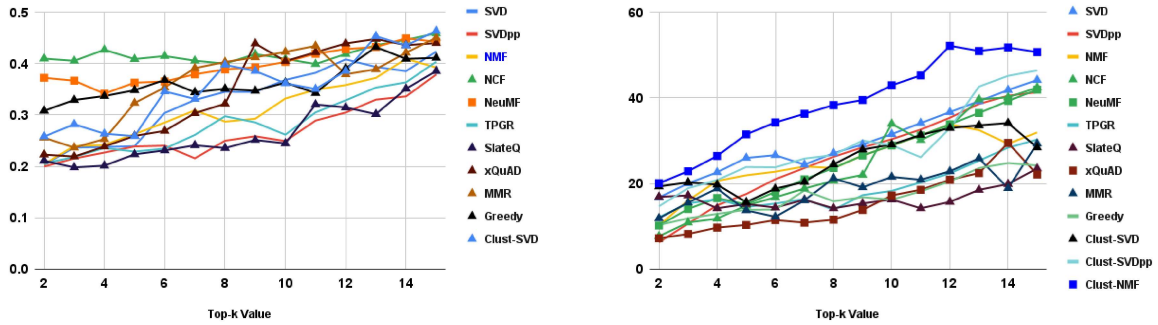
The comparison of our proposed model Clus-DR model for diverse recommendation systems is conducted over a large and real-time dataset MovieLens-1M, ML-100k, LastFM (2k) and Goodbooks (10k) datasets. We compare our model with two different techniques:

1. Accuracy Centric: We use basic collaborative algorithms like SVD (Singular value decomposition) ([145]), SVDpp (Singular value decomposition++) ([146]), NNMF (Non-negative Matrix Factorization) ([147]) and both variants of Neu-



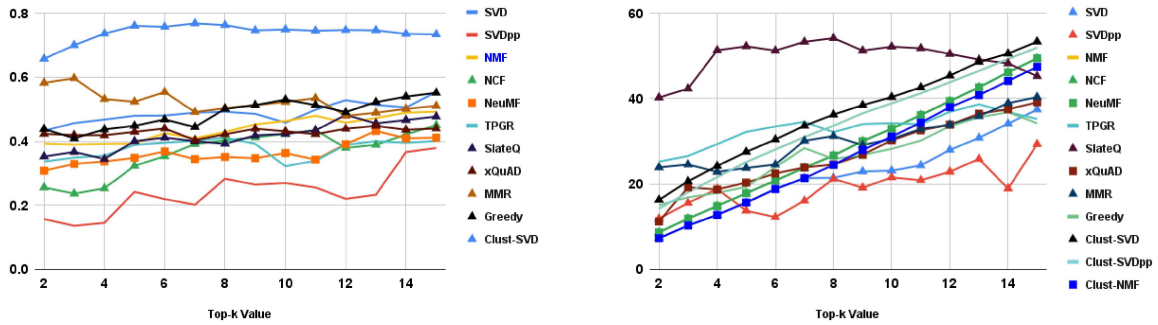
(a) Diversity Graph (b) Coverage Graph

Figure 4.7: A demonstration for top-k diversity and coverage analysis for ML-1M dataset for baseline and our proposed Clus-DR model



(a) Diversity Graph (b) Coverage Graph

Figure 4.8: A demonstration for top-k diversity and coverage analysis for LastFM dataset for baseline and our proposed Clus-DR model



(a) Diversity Graph (b) Coverage Graph

Figure 4.9: A demonstration for top-k diversity and coverage analysis for Goodbooks (10k) dataset for baseline and our proposed Clus-DR model

Table 4.6: Comparison of various results for the proposed model with the baseline model for diversity and coverage for top 5,10,15 recommendations.

Methodology	Algorithm	Intralist Diversity			Coverage		
		Top-5	Top-10	Top-15	Top-5	Top-10	Top-15
Baseline (Accuracy based algorithm)	SVD [145]	0.4794	0.4582	0.5535	15.65	31.11	47.41
	SVDpp ([146])	0.4688	0.4492	0.4788	17.86	32.95	49.41
	NMF ([147])	0.3928	0.4632	0.4923	25.12	38.88	51.92
	NCF ([148])	0.3334	0.3652	0.3981	29.76	32.21	39.88
	NeuMF ([148])	0.3674	0.3988	0.4221	32.54	34.32	42.02
	TPGR ([149])	0.3885	0.3221	0.4008	32.19	34.25	35.22
	SlateQ ([150])	0.4002	0.4229	0.4771	45.22	52.14	51.29
Baseline (Diversity algorithm)	xQuAD ([151])	0.4293	0.4311	0.4398	20.33	30.22	39.11
	MMR ([152])	0.5233	0.5227	0.5103	23.74	30.55	40.33
	Greedy ([142])	0.4482	0.5299	0.5512	19.33	28.22	34.22
Clus-DR (Individual Diversity)	Clust-SVD	**0.7601**	0.7489	0.7336	28.90	41.53	54.16
	Clust-SVDpp	0.7069	**0.7607**	0.7703	27.50	40.34	53.32
	Clust-NMF	0.687	0.713	**0.8028**	**71.48**	**80.92**	**85.67**
Clus-DR (item feature)	Clust-SVD	0.7571	0.745	0.7036	25.65	41.11	51.41
	Clust-SVDpp	0.7244	0.7641	0.7223	27.86	42.95	52.41
	Clust-NMF	0.4251	0.7531	0.7728	41.48	50.92	55.67

ral Collaborative Filtering (NCF) and NeuMF [148]. We also evaluated our proposed approach with reinforcement learning-based recommendation model's TPGR ([149]) and SlateQ ([150]) for three datasets ML-100k, Ml-1M, and LastFM (2k).

2. Diversity Centric: Next, we evaluate our model for diversity so we use state of an art diversification algorithm first is the Greedy algorithm ([142]), second is xQuAD ([151]), and the third one is MMR (Maximal Marginal Relevance) ([152]). These algorithms are used as re-ranking approaches used as a post-

processing step for diversity in the recommendation system.

Table 4.7: Comparison of accuracy measures and top-15 results for the proposed model with the baseline model for precision and nDCG measure.

Proposed Model	Algorithm	RMSE	MAE	Precision	nDCG
Baseline (Accuracy based algorithm)	SVD ([145])	0.974	0.8319	0.1572	0.1654
	SVDpp ([146])	**0.8648**	**0.6022**	0.1361	0.1444
	NMF ([147])	1.1058	0.9928	0.1453	0.1579
	NCF ([148])	1.1021	0.9232	0.3211	0.4112
	NeuMF ([148])	1.0132	0.8965	0.3487	0.4321
	TPGR ([149])	1.0998	0.9823	0.2544	0.2779
	SlateQ ([150])	1.0811	0.9466	0.2311	0.2841
Baseline (Diversity based algorithm)	xQuAD ([151])	-	-	0.2422	0.2422
	MMR ([152])	-	-	0.2196	0.2347
	Greedy ([142])	-	-	0.2016	0.2051
Clus-DR (individual diversity score)	Clus-SVD	1.1107	0.9707	**0.2831**	**0.2781**
	Clus-SVDpp	1.1181	0.8987	0.2646	0.2358
	Clus-NMF	1.1107	0.925	0.2697	0.2361
Clus-DR (item feature)	Clus-SVD	1.112	0.8906	0.2557	0.2667
	Clus-SVDpp	1.1181	0.8987	0.2203	0.2273
	Clus-NMF	1.1141	0.9765	0.2332	0.2443

For comparative analysis of our Clus-DR model, we use two different algorithms. First, we use state-of-the-art algorithms for accurate recommendation generation, and second, algorithms are used for diversity enhancement of the recommendation system. Results for our proposed approach are shown in Tables 4.7, 4.6. In terms of accuracy, our model performance is limited. The accuracy of the reinforcement-based learning methods is perfect. The reason behind this accuracy drop in our proposed model is the diversity-accuracy trade-off and the sparsity of the dataset which is insignificantly

enhanced due to the clustering and pair-wise training. However, the main aim of our approach here is to increase the model performance for diversity and coverage, and our model performs best in terms of diversity and coverage which is shown in Table 4.5. The comparison of the diversity in top-n recommendation generation is calculated using intralist-diversity (IL-D) measure defined in the equation (4.10).

The experiment of the proposed diverse recommendation generation method with other baseline methods is implemented with the ML-1M dataset. The results obtained from the Clus-DR model are analyzed using various accuracy metrics. Table 4.5 shows that precision and nDCG for the Clus-DR model are more reliable than the baseline recommendation model. From Table 4.5, we concluded that using an individual diversity score for user clustering is good enough for diversity and accuracy instead of item feature scores. Accuracy drop in Clus-NMF and Clus-SVD are almost similar, but in terms of diversity and coverage, Clus-NMF performance is better than Clus-SVD.

We also compared our model's diversity with other baseline algorithms (Accuracy Centric, Diversity Centric) using intra-list diversity measure defined in equation (4.10). Table 4.6 shows the comparison result for diversity for various recommendation list sizes, and it presents a better result for the Clus-DR model. From the comparative results, it is notable that we got on average **51.2%** more diverse items in the top-15 recommendation list than items recommended from the basic collaborative algorithm.

Next, the comparative results are for coverage analysis, which is for item space coverage, and shows the range of items that a recommender system can predict. Here, in this comparison graph, item space coverage is evaluated based on the genre of movies. We evaluated the Clus-DR model for coverage analysis, and the Table 4.6 shows the comparison result for Clus-DR model and another recommendation model for various recommendation list sizes. In coverage also we got average improvement in **29.8**% in item space. The comparative analysis for diversity and coverage for datasets ML-1M, ML-100k, LastFM and Goodbooks (10k) are shown in Figures 4.6, 4.7, 4.8, 4.9.

Similarly, Figures 4.10, 4.11, 4.12 represent the obtained results and their comparison graph of precision, recall, nDCG measures using a bar graph for ML-1M dataset.

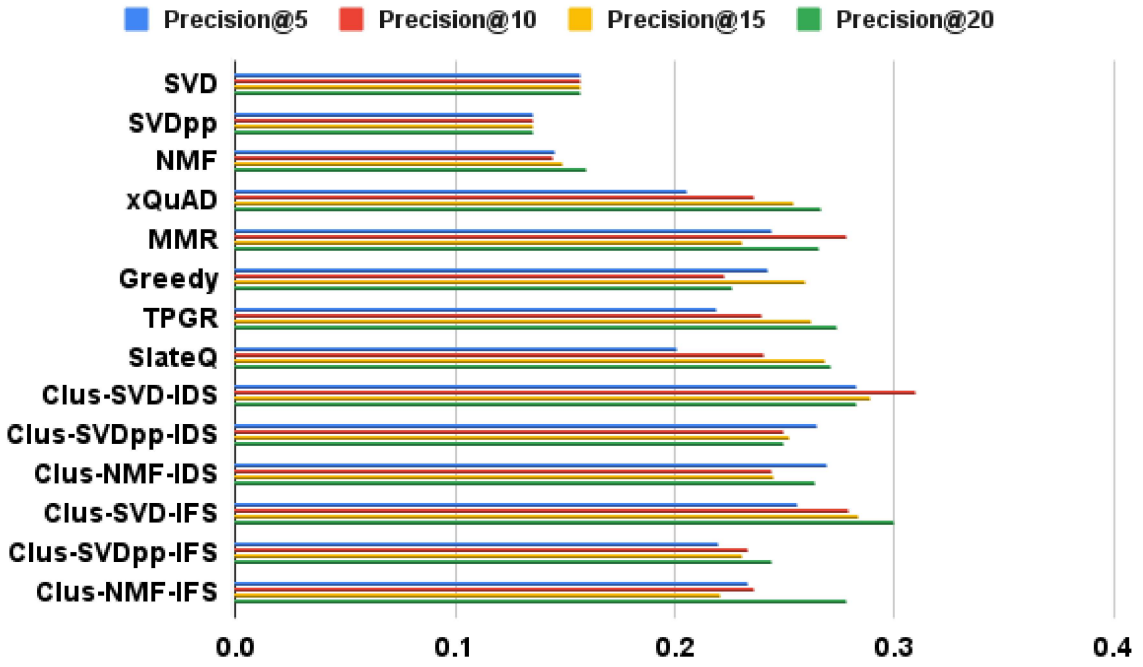


Figure 4.10: A demonstration for top-k precision analysis for ML-1M dataset for baselines and proposed Clus-DR model

4.4.4 Accuracy-Diversity Trade-off

We first study the trade-off when the models are trained for rating prediction using the ranking technique. The descending order of predicted ratings gives a ranking of the recommendation list generated for the user. Now we evaluate the trade-off between accuracy and diversity. The trade-off is plotted for the top-k recommendation list evaluation using nDCG and intralist diversity in Figure 5.11c. The higher value of nDCG leads to a decrease in IL-D and vice-versa. Similarly, we plotted a trade-off between nDCG and coverage in Figure 4.13b, showing similar behaviour to IL-D, where coverage increases while nDCG decreases.

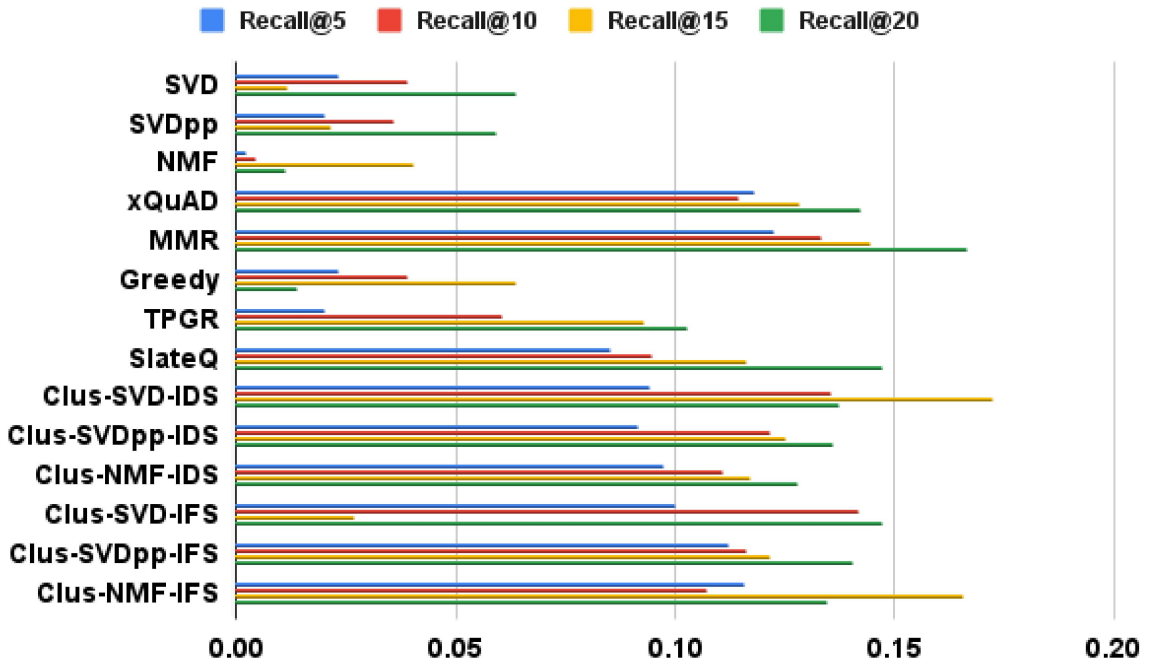


Figure 4.11: A demonstration for top-k recall analysis for ML-1M dataset for baselines and proposed Clus-DR model

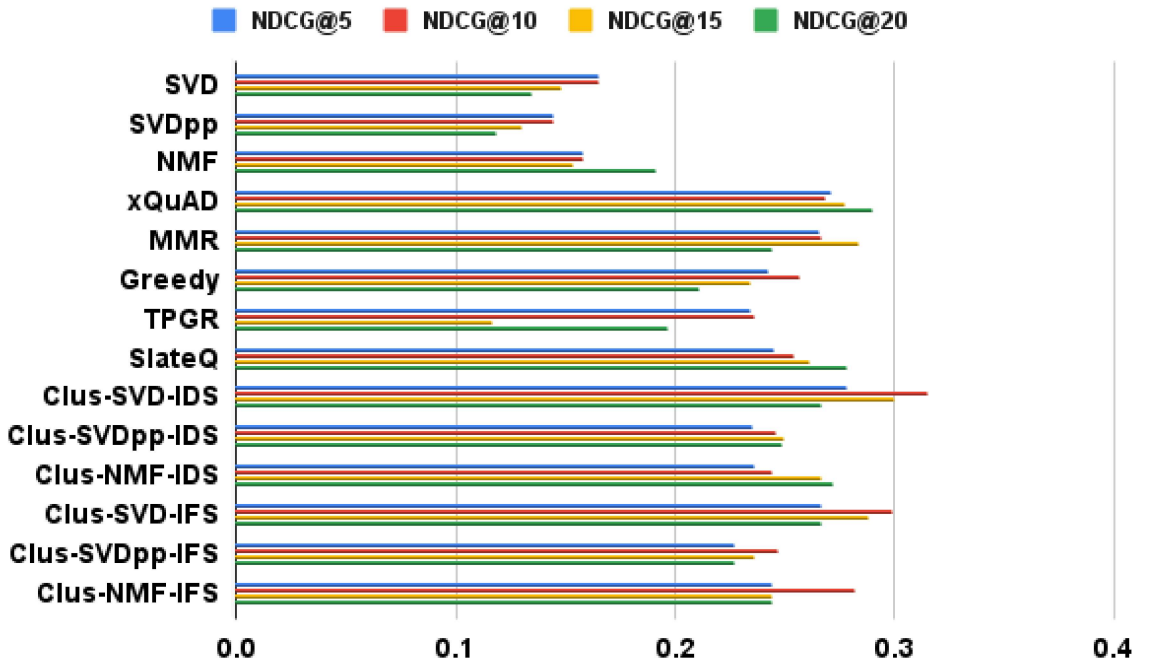
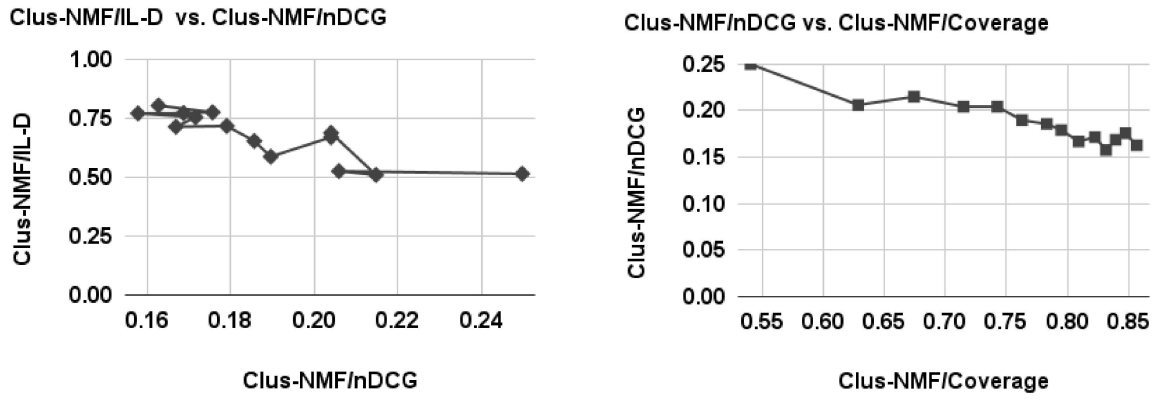


Figure 4.12: A demonstration for top-k nDCG analysis for ML-1M dataset for baselines and proposed Clus-DR model



(a) nDCG vs. intralist diversity

(b) nDCG vs. coverage

Figure 4.13: Accuracy-diversity trade-off for a top-k recommendation for Clus-NMF algorithm

4.5 Ablation Study

- **Effect of various clustering algorithms**: In our approach, we prove the flexibility of the approach by changing various clustering algorithms. In Table 4.8, we show the result obtained from $k = 10$ using various clustering algorithms. Similarly, we shift the clustering algorithm from k-means to MiniBatchk-means, Hierarchical clustering and Spectral clustering. All the comparative results are shown in Table 4.8. The coverage result for other clustering algorithms (Minibatchk-means, Hierarchical clustering and Spectral clustering) are shown in Table 4.9. The results are improved by changing spectral clustering in our proposed Clus-DR model. The diversity, as well as the coverage in spectral clustering, are improved significantly. In future, we will extend this methodology with the graph convolution network to achieve an accuracy-diversity tradeoff in the recommendation system.
- **Effect of user and item latent vector**: We obtained the best result for the individual diversity score for the Clus-NMF approach, where the rating prediction is based on the user and item latent factor size. We performed an analysis

Table 4.8: Comparison of various results for Clus-DR model using different clustering algorithm

Clustering Algorithm		**Clus-SVD**	**Clus-SVDpp**	**Clus-NMF**
minibatachk- means	RMSE	1.2141	1.1281	1.0529
	MAE	0.9765	0.8987	0.8281
	MAP@20	0.0131	0.0145	0.0197
	nDCG@20	0.23	0.2245	0.2522
	Precision@20	0.2198	0.2226	0.2489
	Recall@20	0.033	0.0379	0.0473
	IL-D Top-20	0.7985	0.7861	0.7735
Hierarchical Clustering	RMSE	1.3321	1.3221	1.1665
	MAE	0.9822	0.8987	0.8112
	MAP@20	0.0221	0.0145	0.0213
	nDCG@20	0.212	0.2322	0.2775
	Precision@20	0.2322	0.2298	0.2554
	Recall@20	0.0831	0.0355	0.1081
	IL-D Top-20	0.5781	0.6781	0.7233
Spectral Clustering	RMSE	1.1141	1.1075	**1.0213**
	MAE	0.9905	0.8944	**0.8358**
	MAP@20	0.0751	0.0605	**0.0628**
	nDCG@20	0.3775	0.2992	**0.3487**
	Precision@20	0.3665	0.3456	**0.2701**
	Recall@20	0.1281	0.1043	**0.1081**
	IL-D Top-20	0.5985	0.5288	**0.7861**

Table 4.9: Comparison of coverage results (%) for the proposed Clus-DR model for cluster size (k)= 10 and for Minibatachk-means algorithm.

	Clus-SVD	**Clus-SVDpp**	**Clus-NMF**
K-means	46.47	48.57	49.20
Minibatchk-means	15.59	51.57	52.35
Hierarchical Clustering	41.21	44.92	49.13
Spectral Clustering	44.12	39.67	**51.92**

for accuracy using various latent factor sizes (20,50,100,150,200,250). The accuracy results obtained from Clus-NMF are shown using a line graph in Figure 4.14, where we conclude that for latent factor size 100 we obtained the best results.

- **Online Experiment**: The online evaluation of the recommendation system is a more realistic experiment. Our proposed model focuses on the top-n recommendation generation for each user, so we need to find out models efficiency for the new user and unseen items. For new user and unseen item, we experiment with leaving one out cross-validation where we remove one data from each user's top-n training data list and use that data to evaluate our model's performance. For cold-start prediction, we followed the experimental analysis followed by [115], where they divided users into three groups according to their rating records $(0, 5)$, $(5, 15)$ and $(15, 30)$. We compare our results with traditional SVD, SVDpp and NMF approaches. The comparative performance of our proposed Clus-DR model is shown in Table 4.10.

4.6 Statistical Significance Test

We perform a statistical significance test for a performance comparison of our proposed approach Clus-DR and the second-best performing approach. A statistical significance test is based on a few simple ideas: hypothesis testing, normal distribution, and p

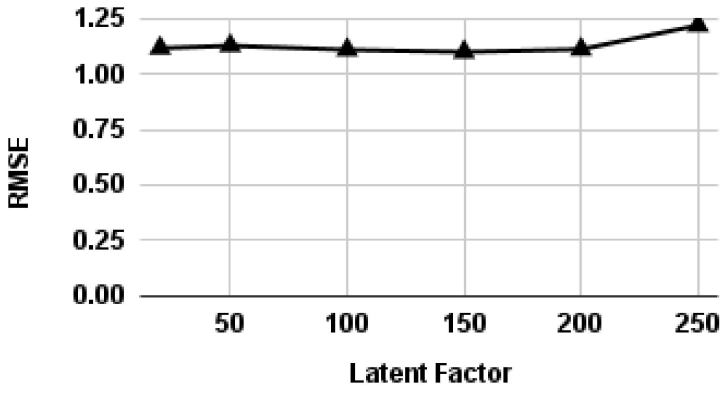


Figure 4.14: Performance analysis for various latent factors for our proposed approach

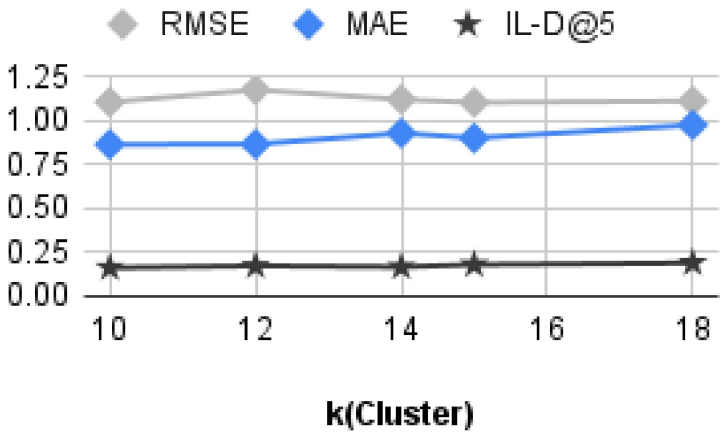


Figure 4.15: Effect of cluster value (k) on RMSE, MAE and diversity for top 5 recommendation

Table 4.10: Cold-start prediction performance for the Clus-DR model

Group	Approach	RMSE	MAE
Group 1 (0-5)	SVD	1.9662	0.9985
Group 2 (5-15)	SVD	1.8112	0.9887
Group 3 (15-30)	SVD	1.7665	0.9668
Group 1 (0-5)	Clus-DR	1.1985	0.9089
Group 2 (5-15)	Clus-DR	1.1507	0.9205
Group 3 (15-30)	Clus-DR	1.0797	0.9021

values. We performed a significance test for analysis of results obtained from Clus-DR to check whether they are significant or not. We did a paired t-test as a shred of statistical evidence, which shows that the mean difference between different paired results on a distinct outcome is significantly different from zero.

The parametric t-test is defined as:

$$t = \frac{m}{s/\sqrt{n}} \tag{4.12}$$

where m is mean s is standard deviation of the difference between all pairs and n is the total number of samples we taken for testing. We test our hypothesis using a one-sided t-test where our hypothesis says that the difference between the mean of two different samples is greater than zero. For this test, we considered precision@k and nDCG@k values for top 5, 10, 15, 20, respectively. We also use the actual and predicted rating mean for the Clus-DR for the same hypothesis to check the stability and prove the significance of the proposed Clus-DR against other state-of-the-art techniques. For the statistical analysis of the result, we have used the t-distribution with n-1 degree. The p-value should be less than the x (x = 0.01, 0.05). We reject the null hypothesis for nDCG@k value paired samples. Where the first group medium value of the observed effect is $d = 0.4$and $SD(standard\ deviation) = 0.017221$ and for the second group $mean = 0.34080$ and $SD = 0.00509$. We reject the null hypothesis with a 95% confidence interval of this difference from -0.01121 to 0.01121. Similarly, we rejected

the hypothesis for precision with 95% confidence interval with t value = 0.007638.

4.7 Summary

In order to solve the diversity problem in the recommendation system, this work proposes a model based on user clustering that includes a wide variety of reliable recommendations. The diversity significantly improved for our suggested model, Clus-DR. However, it reduces the accuracy of our proposed model because of the accuracy-diversity trade-off. To compare and evaluate our methodology, we make use of four datasets from the movie, music, and book categories. We will learn about employing graph neural networks in our future study, where we will attempt to balance accuracy-diversity trade-offs after learning from numerous experimental and ablation studies.