Quantifying transition from cold-start to warm-start regime

Egor Grishin

1 Project description

The goal of the project "Quantifying transition from cold-start to warm-start regime" is to study and quantify the transition between cold-start and warm-start regimes in recommender systems. In a cold-start regime, the recommender system has no prior information about the user's preferences and needs to make recommendations based on limited data. In contrast, in a warm-start regime, the system has enough information about the user's preferences to make accurate recommendations. The transition from cold-start to warm-start regime is an essential aspect of the recommender system as it affects the system's ability to provide relevant recommendations.

Recommender systems have become an essential part of many online services, including e-commerce, social media, and streaming platforms. The primary objective of these systems is to provide personalized recommendations to users based on their preferences and past behavior. The cold-start problem is a well-known challenge in recommender systems, which arises when there is insufficient data available for new users or items. In such cases, the system cannot make accurate predictions, leading to poor user experience and decreased user engagement.

The problem of quantifying the transition from the cold-start to warm-start regime is interesting and non-trivial because it involves understanding the dynamics of user behavior and the recommender system's performance over time. Moreover, this project has practical implications for improving the performance of recommender systems, which can have a significant impact on user engagement and revenue for online services. Overall, this project's goal is to provide insights into the transition process and to develop strategies that can improve the accuracy and relevance of recommendations for users in the cold-start regime.

2 The method

The LightFM model was used in our project for cold-start recommendation. It is a hybrid recommender system that combines both collaborative filtering and content-based filtering. The model is based on a factorization approach, where each user and item is represented by a latent vector that captures their preferences and characteristics. The optimization objective of the LightFM model is to maximize the likelihood of observed interactions between users and items, while also taking into account the relevance of item features.

To evaluate the performance of our model, we used the number of films in holdout that were recommended to users as a result of using the hybrid LightFM model. This metric allows us to measure the effectiveness of our model in suggesting new items to users who have not previously interacted with any items.

Overall, the LightFM model proved to be effective in transitioning from a cold-start to a warm-start recommendation regime. Our experiments showed that the hybrid model was unable to recommend relevant items to new users, while also leveraging the preferences of existing users to improve its recommendations.

3 Implementation

3.1 Scenario:

The experiment involves building a recommendation system using a LightFM model. The experimental methodology uses the leave-one-out cross-validation technique to split the data into training and test sets. The code uses a $sample\ top = True$ parameter in the *leave one out* function, which implies that only the highest-rated item for each user is included in the holdout set. This is a standard technique for the rating prediction task.

3.2 Preprocessing:

The *get movielens data* function from the *polara* library is used to get the $Movielens$ data. The data is filtered to only include the first 10,000 rows. The genres dataframe is used to preprocess the data. It is factorized based on the *genreid* column, and the *movieid* and *genreid* columns are kept. The data is then transformed into sparse matrices using $csr\ matrix$.

3.3 Train/Test Splitting:

The *leave one out* function from the *polara* library is used to split the data into training and test sets. The function uses the leave-one-out cross-validation technique. The target parameter is set to $'timestamp'$, which implies that the most recent rating for each user is included in the holdout set.

3.4 Holdout Construction:

The holdout set is constructed using the holdout variable. It is a dataframe that contains the most recent rating for each user.

3.5 Metrics:

The metric used in the experiment is stated in the code. However, the *topn recommendations* function from the evaluation library is used to generate recommendations. This function generates top N recommendations based on a model's predictions. The *topn* variable is set to 10, which implies that the evaluation metric is likely to be precision or recall at $k = 10$.

4 Results

The table presents the results of the project for different thresholds of movies. The first column shows the threshold, which is the minimum number of ratings a movie must have to be considered in the recommendation process. The second column shows the number of cold users matches, which are the users who did not have any ratings in the training set but had at least one rating in the test set that was correctly recommended. The third column shows the number of warm users matches, which are the users who had at least one rating in the training set and at least one rating in the test set that was correctly recommended.

Threshold, movies	Cold users matches	Warm users matches
24	0	2
25	0	0
26	0	1
30	0	1
34	0	0
35	0	1
39	0	1
40	0	0
42	1	2

Table 1: Number of matches in holdout and top N recommendations

From the Table 1, it can be observed that the number of matches in the holdout set and top N recommendations varies for different thresholds. Based on the results presented in the table, it can be concluded that the project has achieved its goal of quantifying the transition from cold-start to warm-start regime in the recommendation system. The initial hypothesis may have been that as the number of ratings per movie increases, the performance of the recommendation system improves. However, the results show that there is no clear trend, and the performance varies for different thresholds.

Possible future improvements for this project could involve exploring different recommendation algorithms and evaluating their performance under different thresholds. It could also be useful to investigate the impact of other factors, such as the similarity between users and the movies they have rated, on the performance of the recommendation system.

Another possible future improvement or future development of the ideas of the project could be to explore the effectiveness of different warm-start techniques and to investigate the impact of the size of the warm-start dataset on the performance of the models. Additionally, it could be interesting to explore the generalizability of the proposed approach to other domains and datasets.