Table of Contents

TASK 1: RECOMMENDER SYSTEM CHALLENGE 3

INTRODUCTION 3

ALTERNATING LEAST SQUARES MODEL 3

MATRIX FACTORIZATION MODEL 4

NEURAL MATRIX FACTORIZATION 4

COMPARE MODEL AND RESULT CONCLUSION 5

REFERENCE 6

INTRODUCTION

In this report, we will focus on two main tasks. The first is Recommender System Challenge. The second is Node Classification in Graphs.

In task of Recommender System Challenge, I will build a recommender system to recommend a list of items to each user by using the dataset contains a set of interactions between users and items. Finally, upload the result to Kaggle to test the NDCG.

In task of Node Classification in Graphs, I will perform the node classification by using the graph dataset of citation network to classify the nodes in the network into several categories. Finally, I will evaluate the performance of different classification algorithms.

TASK 1: RECOMMENDER SYSTEM CHALLENGE

Introduction:

In the task 1, I will build a recommender system to recommend a list of items to each user. For this task, the dataset I will use is collected from an online social network platform. This dataset records the information that a set of interactions between user and items. If a user interacts with an item, then there will record in the dataset. In addition, the ratings in this dataset only contain 0 or 1. It means if the rating is 1, the user interacted with the item. On the other hand, if the rating is 0, it means no observation of interaction for the user and the item.

To finish this task, I will use three models which are alternating least squares model, matrix factorization model (with bias), and neural matrix factorization model to build the recommender system to recommend the top 10 items for each user.

Alternating least squares (ALS) model:

The Alternating Least Squares model is a form of matrix factorization which could reduce user-item matrix into a very smaller amount of dimension called hidden or latent features. In addition, the train dataset only contains rating of 1. Therefore, I combine the validation dataset with original train dataset to the new train dataset. Finally, here are the steps of build alternating least squares model.

Steps of Alternating least squares model:

1. Preparing Data:
    a. Combine the original train dataset and validation dataset into new train dataset.
    b. Drop out the duplicate data

    c. Create two sparse matrices. One is the sparse user-item matrix which is used for recommendations. Another one is the sparse item-user matrix which is used for fitting the model.
2. Initialize the ALS recommendation model
3. Fit the model using the sparse item-user matrix.
4. Check the candidate list in test dataset and recommend the top 10 items for each user.

Matrix factorization Model (with bias):

The matrix factorization algorithm is decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. In addition, due to variation in rating among different users, I add item bias and user bias to this matrix factorization model. Here are the steps of build matrix factorization model.

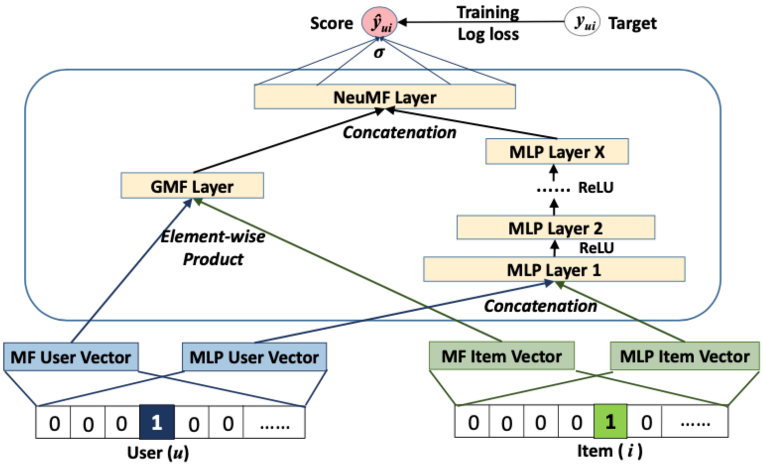
Steps of Matrix factorization model:

1. Preparing Data:
    a. Combine the original train dataset and validation dataset into new train dataset.
    b. Drop out the duplicate.
2. Build the Model that contains the user bias and item bias.
3. Train the Model
4. Fit the Model
5. Check the candidate list in test dataset and recommend the top 10 items for each user.

Neural matrix factorization Model:

According the Neural Collaborative Filtering (He, X., Liao et al., 2017), the neural matrix factorization model is combining the generalized matrix factorization (GMF), a generalized version of matrix factorization that under neural collaborative filtering that uses the sigmoid function to output, and multi-layer perceptron model (MLP) which using a standard multi-layer perceptron to learn the interaction between user and item latent features by adding hidden layers on the concatenated vector to reinforce each other to better model the complex user-item interactions.

In addition, due to the objective function that uses gradient-based optimization method of Neural matrix factorization only find locally optimal solutions. I initialize Neural matrix factorization by using the pre-trained models of generalized matrix factorization model and multi-layer perceptron model.



Neural matrix factorization model

Steps of Neural matrix factorization model:

1. Preparing Data:
   a) Combine the original train dataset and validation dataset into new train dataset.
   b) Drop out the duplicate.
2. Pre-training:
   a. Build the Generalized Matrix Factorization (GMF) model.
   b. Train the GMF model with initializations until convergence.
   c. Save the model parameters of GMF model.
   d. Build the Multi-Layer Perceptron (MLP) model.
   e. Train the MLP model with initializations until convergence.
   f. Save the model parameters of MLP model.
3. Build the Neural matrix factorization model.
4. Use the GMF and MLP model parameters as the initialization for the corresponding parts of Neural matrix factorization's parameters.
5. Train the NeuMF model.
6. Fit the model.
7. Check the candidate list in test dataset and recommend the top 10 items for each user.

Compare model and result conclusion:

Model	NDCG on Kaggle
Alternating Least Squares	0.22
Matrix Factorization (With Bias)	0.15
Neural Matrix Factorization	0.14

According to Normalized Discounted Cumulative Gain (NDGC):

1. The Alternating Least Squares has highest NDGC around 0.22. It means the ALS model is the most suitable model in this case.
2. The Matrix Factorization (With Bias) and Neural Matrix Factorization have similar NDCG around 0.15.
3. The reason may because in this case, the ratings are only 0 and 1, and it has small data size. Therefore, if we use complex model to build the model, it will cause high test error.
4. Finally, I choice the ALS model as my model to submit to Kaggle.

Reference:

He, X., Liao, L., Zhang, H., Nie, L., & Hu, X. (2017). Neural Collaborative Filtering. arXiv.org. Retrieved from http://search.proquest.com/docview/2075684171/