CSE 258- Assignment 1

Name: Ling Jiang
Student ID: A53321363
Kaggle username: lingjiangj
Leaderboard display name: LJ

Part 1 Reading Prediction

Accuracy: 0.74707
Leaderboard Rank: 41

The model I chose to make the prediction included Jaccard similarity and book popularity (the number of users who have read the specific book). And I used the whole dataset to calculate the book popularity and to find the similar user and book for the user and book we want to predict. This model ended up with an accuracy of 0.74707 and ranked 41 on the leaderboard.

Prediction Approach

Because the test set has been constructed such that exactly 50% of the pairs correspond to read books and the other 50% do not, the prediction results should be "1" (read) for the half of the user-book pairs with the highest multiplication results of Jaccard similarity and book popularity, and be "0" (unread) for the other half of the pairs.

For each user(u) in the test set, I first got the Jaccard similarity for each pair (u, book(b)). After getting the similarities of a list of books which the user has read, I calculated ***the average similarity***. This is because when using the maximum similarity of the list, if only one book in the book(b') list has a high similarity and others' similarities are 0, this book would get a high similarity and prediction result would probably be wrong. Thus, using average similarities can get a more accurate result compare to the maximum similarity of the book list.

Then I multiplied the average Jaccard similarity with the popularity of b. I calculated the number of users who have read the book to represent the book popularity. If a book is popular, people are more likely to read it, so including this feature in the model improved the prediction performance a lot.

After that, I sorted the multiplication results of these pairs for each user. To achieve this, I built a dictionary with all of the users in test set as keys, and values are similarities and their corresponding user-book pairs. Finally, I got the prediction results based on the approach mentioned above. Jaccard similarities and book popularities are all calculated from the whole dataset.

In this task, I have tried other methods to improve the prediction performance, such as cosine similarity and involving thresholds for both Jaccard similarity and book popularity, but they all fail to reach a better performance.

Part 2 Rating Prediction

MSE: 1.09915
Leaderboard Rank: 14

I used the Simple (bias only) Latent Factor Model to get the best result in rating prediction. I involved 2 different lambdas for user Bias (beta_user) and book Bias(beta_book) respectively. And I also used the whole dataset to train the model. The Mean Squared Error (MSE) was 1.09915 and ranked 14 on the leaderboard.

Latent factor model

Prediction = alpha + beta_user + beta_book
alpha: the average value of rating for all the training data
beta_user: the user rating bias between personal rating tendency and average rating value
beta_item: the item rating bias between books received rating and average rating value

1. Initialized the alpha, beta_user and beta_book to be 0. Calculated the global average rating.
2. Trained the Latent Factor Model without gamma, get the alpha, beta_user and beta_book.
3. Fixed the alpha, beta_user and beta_book using the following procedure until convergence. For beta_user and beta_book, I used two different lambdas to convergence. The differentiate procedure included both MSE and Regularizer.

$$\alpha = \frac{\sum_{u,i \in \text{train}} (R_{u,i} - (\beta_u + \beta_i))}{N_{\text{train}}}$$
$$\beta_u = \frac{\sum_{i \in I_u} R_{u,i} - (\alpha + \beta_i)}{\lambda + |I_u|}$$
$$\beta_i = \frac{\sum_{u \in U_i} R_{u,i} - (\alpha + \beta_u)}{\lambda + |U_i|}$$

Parameter selection

Because user biases should be valued more than book biases in this task, the constrain for book biases should be bigger. So, I used two different lambdas for beta_user and beta_book seperately. To find the best lambdas, I did several experiments on it. Finally, the best lambdas for converging beta_user and beta_book are 2 and 13 respectively.

In this task, I have also tried to include gamma to train the model, as well as to use Pearson Similarity. But since I did not have other features about users or books, the methods failed to get a good performance.