CSE 158/258, Fall 2023: Homework 3

Instructions

Please submit your solution by Monday, Nov 13. Submissions should be made on gradescope. Please complete homework individually.

These homework exercises are intended to help you get started on potential solutions to Assignment 1. We’ll work directly with the Assignment 1 dataset to complete them, which is available from:

http://cseweb.ucsd.edu/classes/fa23/cse258-a/files/assignment1.tar.gz

You’ll probably want to implement your solution by modifying the baseline code provided in the assignment directory.

You should submit two files:

answers_hw3.txt should contain a python dictionary containing your answers to each question. Its format should be like the following:

```python
{ "Q1": 1.5, "Q2": [3,5,17,8], "Q2": "b", (etc.) }
```

The provided code stub demonstrates how to prepare your answers and includes an answer template for each question.

homework3.py A python file containing working code for your solutions. The autograder will not execute your code; this file is required so that we can assign partial grades in the event of incorrect solutions, check for plagiarism, etc. Your solution should clearly document which sections correspond to each question and answer. We may occasionally run code to confirm that your outputs match submitted answers, so please ensure that your code generates the submitted answers.

You may build your solution on top of the provided stub:

Homework 3 stub: https://cseweb.ucsd.edu/classes/fa23/cse258-a/stubs/

Each question is worth 1 mark.

Play prediction

Since we don’t have access to the test labels, we’ll need to simulate validation/test sets of our own. So, let’s split the training data (‘train.json.gz’) as follows:

1. Reviews 1-165,000 for training
2. Reviews 165,001-175,000 for validation
3. Upload to gradescope for testing only when you have a good model on the validation set.

1. Although we have built a validation set, it only consists of positive samples. For this task we also need examples of user/item pairs that weren’t played. For each entry (user,game) in the validation set, sample a negative entry by randomly choosing a game that user hasn’t played. Evaluate the performance (accuracy) of the baseline model on the validation set you have built (1 mark).

2. The existing ‘played prediction’ baseline just returns True if the item in question is ‘popular,’ using a threshold of the 50th percentile of popularity (totalPlayed/2). Assuming that the ‘non-played’ test examples are a random sample of user-game pairs, this threshold may not be the best one. See if you can find a better threshold and report its performance on your validation set (1 mark).

3. A stronger baseline than the one provided might make use of the Jaccard similarity (or another similarity metric). Given a pair (u,g) in the validation set, consider all training items g’ that user u has played. For each, compute the Jaccard similarity between g and g’, i.e., users (in the training set) who have played g and users who have played g’. Predict as ‘played’ if the maximum of these Jaccard similarities exceeds a threshold (you may choose the threshold that works best). Report the performance on your validation set (1 mark).

---

1This is how I constructed the test set; a good solution should mimic this procedure as closely as possible so that your gradescope performance is close to their validation performance.
4. Improve the above predictor by incorporating both a Jaccard-based threshold and a popularity based threshold. Report the performance on your validation set (1 mark).

5. To run our model on the test set, we’ll have to use the files ‘pairsPlayed.txt’ to find the reviewerID/itemID pairs about which we have to make predictions. Using that data, run the above model and upload your solution to gradescope. If you’ve already uploaded a better solution to gradescope, that’s fine too!

Time played prediction

Let’s start by building our training/validation sets much as we did for the first task. This time building a validation set is more straightforward: you can simply use part of the data for validation, and do not need to randomly sample non-played users/games.

Note that you should use the time transformed field, which is computed as $\log_2(\text{time played} + 1)$. This is the quantity we are trying to predict.

6. Fit a predictor of the form
   
   $$\text{time}(user, item) \simeq \alpha + \beta_{user} + \beta_{item},$$

   by fitting the mean and the two bias terms as described in the lecture notes. Use a regularization parameter of $\lambda = 1$. Report the MSE on the validation set (1 mark).

7. Report the user and game IDs that have the largest and smallest values of $\beta$ (1 mark).

8. Find a better value of $\lambda$ using your validation set. Report the value you chose, its MSE, and upload your solution to Kaggle by running it on the test data (1 mark).

---

2This could be further improved by treating the two values as features in a classifier — the classifier would then determine the thresholds for you!