CSE 158/258, Fall 2022: Homework 3

Instructions

Please submit your solution **by Monday, Nov 7.** 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/fa22/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:

```
{ "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/fa22/cse258-a/stubs/`

Each question is worth 1 mark.

Tasks (Read 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_Interactions.csv.gz') as follows:
(1) Reviews 1-190,000 for training
(2) Reviews 190,001-200,000 for validation
(3) Upload to Gradescope for testing only when you have a good model on the validation set. If you can build such a validation set correctly, it will significantly speed up your testing and development time.

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* read. For each (user,book) entry in the validation set, sample a negative entry by randomly choosing a book that user *hasn't* read.[1] Evaluate the performance (accuracy) of the baseline model on the validation set you have built.

2. The existing 'read prediction' baseline just returns *True* if the item in question is 'popular,' using a threshold of the 50th percentile of popularity (`totalRead/2`). Assuming that the 'non-read' test examples are a random sample of user-book pairs, this threshold may not be the best one. See if you can find a better threshold (or otherwise modify the thresholding strategy); report the new threshold and its performance of your improved model on your validation set.

3. A stronger baseline than the one provided might make use of the Jaccard similarity (or another similarity metric). Given a pair (u, b) in the validation set, consider all training items b' that user u has read. For each, compute the Jaccard similarity between b and b', i.e., users (in the training set) who have read b and users who have read b'. Predict as 'read' 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] This is how I constructed the test set; a good solution should mimic this procedure as closely as possible so that your leaderboard 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.[2]

5. To run our model on the *test* set, we'll have to use the files 'pairs_Read.csv' to find the userID/bookID 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, that's fine too! Your answer should be the string *"I confirm that I have uploaded an assignment submission to gradescope"*.

(CSE 158 only) Tasks (Category prediction)

Please try to run these experiments using the specified training data and dictionary size; an efficient solution should take only a few seconds to run. Please only use a smaller dictionary size (i.e., fewer words), or a smaller training set size, if the experiments are taking too long to run.

6. Using the review data (train_Category.json.gz), build training/validation sets consisting of 190,000/10,000 reviews. We'll start by building features to represent common words. Start by removing punctuation and capitalization, and finding the 1,000 most common words across all reviews ('review_text' field) in the training set. See the 'text mining' lectures for code for this process. Report the 10 most common words, along with their frequencies (your answer should be a list of 10 `(frequency, word)` tuples).

7. Build bag-of-words feature vectors by counting the instances of these 1,000 words in each review. Set the labels (y) to be the 'genreID' column for the training instances. You may use these labels directly with sklearn's *LogisticRegression* model, which will automatically perform multiclass classification. Report performance (accuracy) on your validation set.

8. Try to improve upon the performance of the above classifier by using different dictionary sizes, or changing the regularization constant C passed to the logistic regression model. Report the performance of your solution, and upload your prediction file to the Assignment 1 gradescope.

(CSE 258 only) Tasks (Rating 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-read users/books.

9. Fit a predictor of the form

$$\text{rating}(\text{user}, \text{item}) \simeq \alpha + \beta_{\text{user}} + \beta_{\text{item}},$$

   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.

10. Report the user IDs that have the largest and smallest (i.e., largest negative) values of β_u, along with the beta values.

11. Find a better value of λ using your validation set. Report the value you chose, its MSE, and upload your solution to gradescope by running it on the test data.

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