CSE 258, Fall 2017: Assignment 1

Instructions

In this assignment you will build **recommender systems** to make predictions related to reviews of businesses from *Google Local*.

Solutions will be graded on Kaggle (see below), with the competition closing at **5pm, Monday November 20** (note that the time reported on the competition webpage is in UTC!).

You will also be graded on a brief report, to be submitted electronically on gradescope by the following day. Your grades will be determined by your performance on the predictive tasks as well as your written report about the approaches you took.

This assignment should be completed **individually**.

To begin, download the files for this assignment from:
`http://jmcauley.ucsd.edu/data/assignment1.tar.gz`

Files

train.json.gz 200,000 reviews to be used for training. It is not necessary to use *all* reviews for training, for example if doing so proves too computationally intensive. While these files are one-json-per-line (much as we have seen so far in class), you may find it useful to represent them more concisely in order to produce a more efficient solution. The fields in this file are:

- **businessID** The ID of the business. This is a hashed product identifier from Google.
- **userID** The ID of the reviewer. This is a hashed user identifier from Google.
- **rating** The star rating of the user's review.
- **reviewText** The text of the review. It should be possible to successfully complete this assignment *without* making use of the review data, though an effective solution to the category prediction task will presumably make use of it.
- **reviewHash** Hash of the review (essentially a unique identifier for the review).
- **unixReviewTime** Time of the review in seconds since 1970.
- **reviewTime** Plain-text representation of the review time.
- **categories** Category labels of the product being reviewed.

pairs_Visit.txt Pairs on which you are to predict whether a user would visit a business.

pairs_Category.txt Pairs (userID and reviewHash) on which you are to predict the category of an item. **(Not relevant for CSE258)**

pairs_Rating.txt Pairs (userIDs and businessIDs) on which you are to predict ratings.

test_Category.json.gz The review data associated with the category prediction *test* set. Again, the field that you are trying to predict has been removed.

baselines.py A simple baseline for each task, described below.

Please do not try to crawl these products from Google, or to reverse-engineer the hashing function I used to anonymize the data. I assure you that doing so will not be easier than successfully completing the assignment.

Tasks

You are expected to complete the following tasks:

Visit prediction Predict given a (user,business) pair from 'pairs_Visit.txt' whether the user visited the business (really, whether it was one of the business they reviewed). Accuracy will be measured in terms of the *categorization accuracy* (1 minus the Hamming loss). The test set has been constructed such that exactly 50% of the pairs correspond to visited business and the other 50% do not. Performance will be measured in terms of the fraction of correct classifications.

Rating prediction Predict people's star ratings as accurately as possible, for those (user,item) pairs in 'pairs_Rating.txt'. Accuracy will be measured in terms of the *(root) mean-squared error* (RMSE).

These error measures are described on *Kaggle*:

Classification accuracy Is equivalent to 1 minus the Hamming Loss: `https://www.kaggle.com/wiki/HammingLoss`

RMSE `https://www.kaggle.com/wiki/RootMeanSquaredError`

A competition page has been set up on Kaggle to keep track of your results compared to those of other members of the class. The leaderboard will show your results on *half of* the test data, but your ultimate score will depend on your predictions across the *whole* dataset.

Grading and Evaluation

This assignment is worth 25% of your grade. You will be graded on the following aspects. Each of the two tasks is worth 10 marks (i.e., 10% of your grade), plus 5 marks for the written report.

- Your ability to obtain a solution which outperforms the baselines on *the unseen portion of* the test data (5 marks for each task). Obtaining full marks requires a solution which is substantially better (i.e., at least several percent) than baseline performance.
- Your ranking for each of the tasks compared to other students in the class (3 marks for each task).
- Obtain a solution which outperforms the baselines on *the seen portion of* the test data (i.e., the leaderboard). This is a consolation prize in case you overfit to the leaderboard. (2 mark for each task).

Finally, your written report should describe the approaches you took to each of the tasks. To obtain good performance, you should not need to invent new approaches (though you are more than welcome to!) but rather you will be graded based on your desicion to apply reasonable approaches to each of the given tasks (5 marks total).

Baselines

Simple baselines have been provided for each of the tasks. These are included in 'baselines.py' among the files above. These baselines operate as follows:

Visit prediction Find the most popular businesses that account for 50% of visits in the training data. Return '1' whenever such a business is seen at test time, '0' otherwise.

Rating prediction Return the global average rating, or the user's average if we have seen them before in the training data.

Running 'baselines.py' produces files containing predicted outputs. Your submission files should have the same format.

Kaggle

We have set up a Kaggle page to help you evaluate your solution. You should be able to access the competition via your UCSD address. The Kaggle pages for each of the tasks are:
`https://inclass.kaggle.com/c/cse158-258-fa17-visit-prediction`
`https://inclass.kaggle.com/c/cse258-fa17-rating-prediction`

You may need to follow the private links to access the competition pages:
visit prediction: `https://www.kaggle.com/t/f5cb34fc0265469eb7c4d4e86618e650`
rating prediction: `https://www.kaggle.com/t/a9061008f33740a6a0b8e120b775bc7b`