Final

DSE 220: Machine Learning

Due Date: 06/12 11:59 PM

1 Instructions

The deadline for the Kaggle competition is June 12th (11:59 PM). The report and the code for the final should be submitted on Gradescope before the deadline. To secure full marks both the report and the code should be in sync and logically correct. Please only submit relevant and legible code. **Please complete the final individually**.

2 Overview

For the take-home final of DSE 220: Machine Learning, we combine all the concepts we have learned so far and apply them to a real world problem. The problem, in very broad terms, is to predict how useful an Amazon product review is going to be. The problem is open-ended and you can use any method you like.

Amazon is a leading e-commerce website and uses Machine Learning applications extensively in their products. For example, they recommend products based on your browsing history and previous purchases so as to maximize their profit (by selling you another product!). For your final, you are given Amazon data. Given a review, the item for which the review is and the user who has written the review, you need to predict how helpful the review will be.

Solutions will be graded on Kaggle. Please follow the link -
https://www.kaggle.com/t/4ce48936219a46a5a278782bd44eaaaf

to view the webpage and signup using your UCSD email id (@ucsd.edu and not @eng.ucsd.edu). **Note that the time reported on Kaggle are in UTC and not PST**.

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

3 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, you may find it useful to represent them more concisely in order to produce a more efficient solution. The fields in this file are:

- **itemID** The ID of the item. This is a hashed product identifier from Amazon.
- **reviewerID** The ID of the reviewer. This is a hashed user identifier from Amazon.
- **helpful** Helpfulness votes for the review. This has two subfields, 'nHelpful' and 'outOf'. The latter is the total number of votes this review received, the former is the number of those that considered the review to be helpful.
- **reviewText** The text of the review. It should be possible to successfully complete this task without making use of the review data, though an effective solution to the helpfulness prediction task will presumably make use of it.
- **summary** Summary of the review.
- **price** Price of the item.
- **reviewHash** Hash of the review (essentially a unique identifier for the review).
- **unixReviewTime** Time of the review, i.e, time elapsed in seconds since 1970. For example - 1400544000
- **reviewTime** Plain-text representation of the review time in human readable format. For example - '05 20, 2014'
- **category** Category labels of the product being reviewed.
- **rating** Rating given by the reviewer.

pairs_Helpful.txt Pairs on which you are to predict helpfulness votes. A third column in this file is the total number of votes, from which you should predict how many were helpful.

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.

Baseline.ipynb A simple baseline for the task, described later.

Please do not try to crawl these products from Amazon/any third-party website, to reverse-engineer the hashing function used to hide the data, or to copy someone else's code off the internet. We assure you that doing so will not be easier than successfully completing the final. Since we are asking for your code, we will know whether or not you parsed the data, and we will check for the originality of code. Any unfair means used could end up earning you a zero on your final exam.

4 Task

Helpfulness prediction - Predict whether a user's review of an item will be considered helpful. The file 'pairs_Helpful.txt' contains (user,item) pairs, with a third column containing the number of votes the user's review of the item received. You must predict how many of them were helpful. Accuracy will be measured in terms of the mean absolute error, i.e., you are penalized one according to the **absolute difference of nHelpful and prediction**, where 'nHelpful' is the number of of helpful votes the review actually received, and 'prediction' is your prediction of this quantity.

The error measure is described on Kaggle:
Mean Absolute error

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**.

5 Grading and Evaluation

You will be graded on the following aspects.

- Your ability to obtain a solution which outperforms the baselines on the total test data (including the unseen portion). Obtaining full marks requires a solution which is substantially better than baseline performance. Following are the marks you'll get for beating every benchmark.
  Baseline: 0.261
  Benchmark I: $\leq$ 0.190 15 marks
  Benchmark II: $\leq$ 0.185 20 marks
  Benchmark III: $\leq$ 0.179 25 marks
  Benchmark IV: $\leq$ 0.176 30 marks
  Benchmark V: $\leq$ 0.173 35 marks
  Benchmark VI: $\leq$ 0.170 40 marks
  Benchmark VII: $\leq$ 0.166 45 marks
  Benchmark VIII: $\leq$ 0.162 50 marks

- Obtain a solution which outperforms the baselines on the seen portion of the test data (i.e., the public leaderboard) to obtain 20 marks. This is a consolation prize in case you overfit to the leaderboard.
- The report accounts for 30 marks. It should describe the approaches you took to perform the task. Make sure that the methods you describe in the report include all the aspects of your final model including pre-processing, feature engineering etc. The aim is that anyone with your report should be able to recreate your results. Even if your model doesn't perform well, you can obtain marks in this section for the comprehensiveness of your analysis.

You can obtain a maximum of 100 marks in this project, which will the scaled down to 40% of the total course assessment. To obtain good performance, you should not need to invent new approaches (though you are more than welcome to!).

6 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(@ucsd.edu, not @eng.ucsd.edu).

https://www.kaggle.com/t/4ce48936219a46a5a278782bd44eaaaf

You can submit only 5 submissions per day to Kaggle. This is to ensure that you don't learn from the test data. Please ensure that you use a good validation set for measuring the performance of your model. **Tuning your models based on the leaderboard position can lead to overfitting and you might end up losing all the marks based on the private leaderboard.**
You need to select two top submissions on which you want us to evaluate you at the end of the competition. If top two submissions are not selected, we pick the top two submissions according to the public leaderboard.

7 Baselines

Simple baseline solution has been provided for the task. These are included in 'Baseline.ipynb' among the files above. The baseline operate as follows:

Multiply the number of votes by the global average helpfulness ratio (nHelpful/outOf), or the user's average helpfulness ratio if we saw the same user in the training data.
Running the given iPython notebook produces files containing predicted outputs. Your submission files should have the same format. Hints will be provided to you on Canvas by the TAs from time to time depending upon the class performance. Please monitor Canvas posts for updates.