Homework One

MSDS Summer 2022

- Submit code via Canvas using Markdown Cells to **clearly** indicate which code answers which question and to answer short answer questions.
- Use `book_ratings_test_v2.csv` to evaluate performance of your model. It contains at least one review from each user and two reviews from each user with more than 10 reviews. (original data is from `http://www2.informatik.uni-freiburg.de/~cziegler/BX/`)
- This will be graded for correctness, but **not** model performance.

You will create three models to predict whether a user will give a "good" or "bad" rating on a certain book based on constructing embeddings of users and books (Matrix Factorization, FF NN with embedding layers, and a FF NN with embedding layers along with other inputs). To do so you will need to prepare the data for input into a PyTorch embedding layer.

1. First create a dictionary for users and books, then do the following.

- Create a DataSet class that outputs the user and book indices in a single tensor (in preparation for input into an `nn.Embedding` layer) and another tensor with 1 corresponding to a good rating (above 4) and 0 corresponding to a bad rating (4 or below).
- Create a DataLoader for the training data and the test data.

2. Create three different classes of models using `nn.Module`. You will need a second DataSet class for the third model.

- A model which predicts the rating a user will give to a book using Matrix Factorization (similar to what you did before in Dr. Interian's course)
- A model which predicts the rating a user will give to a book by embedding both the book and the user as 40-dimensional features, followed by a linear layer (Hint: it will look like `nn.Linear(80, 1)`).
- A model which predicts the rating a user will give to a book by embedding both the book and the user in some feature space (dimension up to you) and by including **at least two other features** from the dataset (such as age, location, year of publication, etc.). Note that for categorical variables you will need to use more embedding layers! Feel free to use any techniques we learned last week in this model.

3. Initialize each of the three models and pass one batch through them to make sure they are working properly.

4. Train each of the models for this classification task using an appropriate loss function for at least two epochs. At the end of each epoch, print the accuracy of your model in predicting whether a user will rate a book as "good" or as "bad" for both the training and test sets.

- For context, I achieved around 70% percent accuracy on the test set after 5 epochs and 20 minutes on my laptop using the second model. I used a batch size of 10000 and Adam optimization with a learning rate of 0.01.
- You will not be graded on model performance, just being able to train the model and print the accuracy. The dataset is rather large, so if you are interested in pushing the performance and trying other methods I suggest using Google Colab or Kaggle GPUs.

5. Take the second model from Q2. Of the following techniques to improve performance which has the biggest impact? Can you give evidence of this?

- Dropout
- Batch Normalization
- Weight decay

Bonus: Investigate the book embedding. Does it seem like "similar" books are "close"?