Predicting Implicit Ratings

Kaggle - Final Project

Group - SageMaker

Jih-Chin Chen, Ronica Gupta

Data

Columns:

- User_id
- Item_id
- Context_feature_id : Related to user_id
- Item_feature_id: Related to item_id

Context:

- The data contains implicit rating for users and items
- Each user could have liked multiple items

Negative Sampling Techniques

1. Random Sampling: We initially tried with randomly sampling the item_ids to populate the user-item negative sample. We tried with multiple frequencies:
    a. Number of negative samples per user = number of positive samples per user
    b. Number of negative samples per user = 2*number of positive samples per user
    c. 5 negative samples per user

2. Probabilistic Sampling:
    a. User Based: The user with maximum liked items gets minimum negative samples
    b. Item Based: The item with maximum frequency gets picked minimum for random sampling

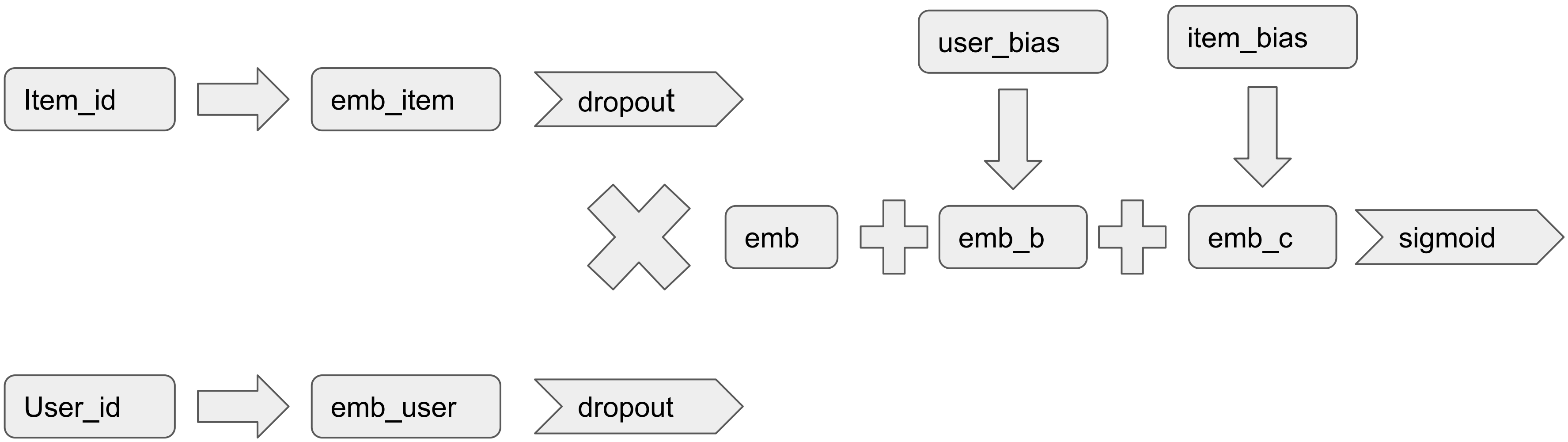
Model 1: Matrix Factorization

Features

- In the MF model, our group only adapted two features: User_id and Item_id.
- Size of user embedding corpus was max(user_id). So missing users in test were encoded with random embeddings.
- Size of item embedding corpus was max(item_id). So missing items in test were encoded with random embeddings.

Model 1: Matrix Factorization

Model Structure



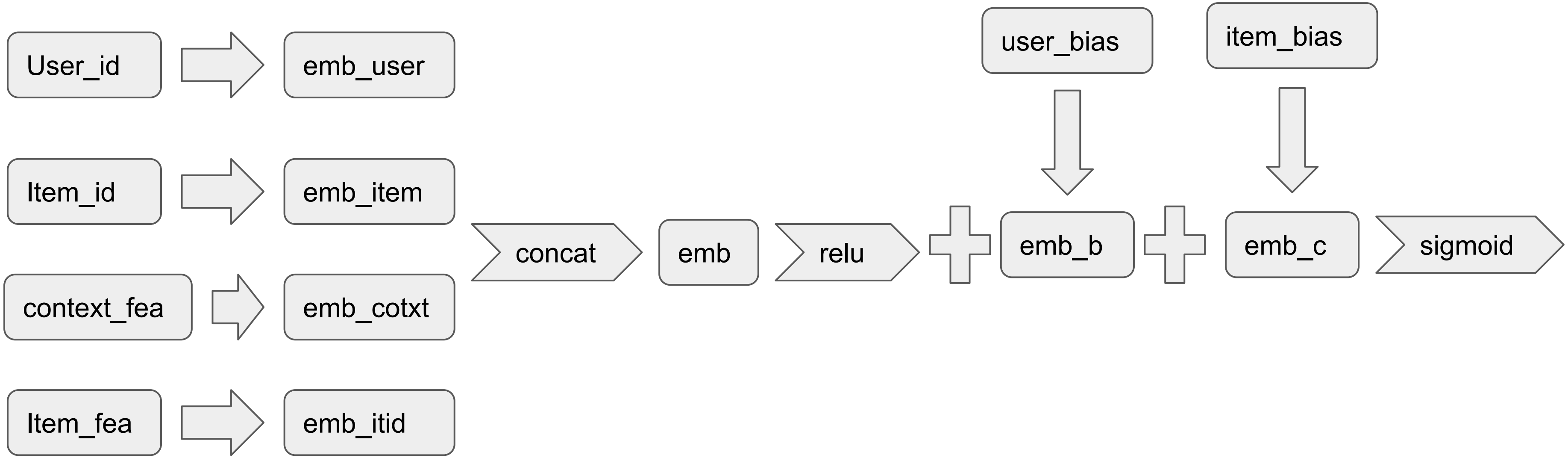
Model 2: Neural Networks

Features

- In the neural networks model, our group adapted all four features: User_id, Item_id, context_feature_id, and Item_feature_id.
- Add a data loader and dataset training in 10000 batch size, to improve hyperparameters.
- Size of user embedding corpus was len(user_encoding)+1. So all missing users in test were encoded with the extra embedding to reduce randomness.
- Size of item embedding corpus was len(item_encoding)+1. So all missing items in test were encoded with the extra embedding to reduce randomness.

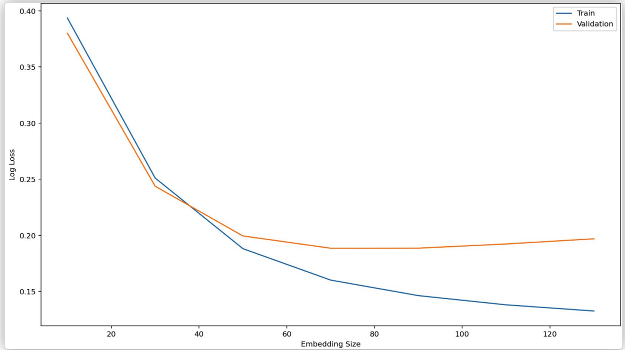
Model 2: Neural Networks

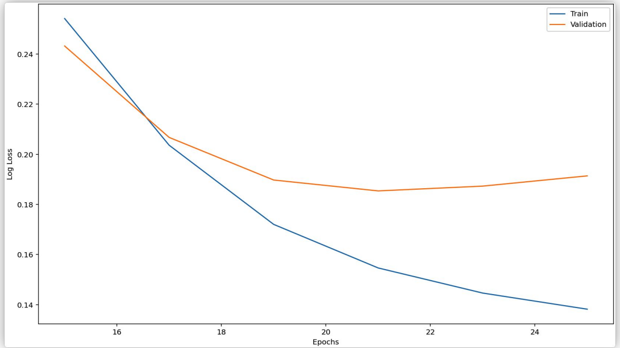
Model Structure

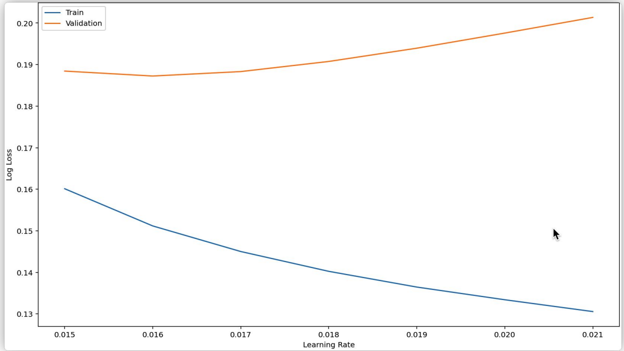


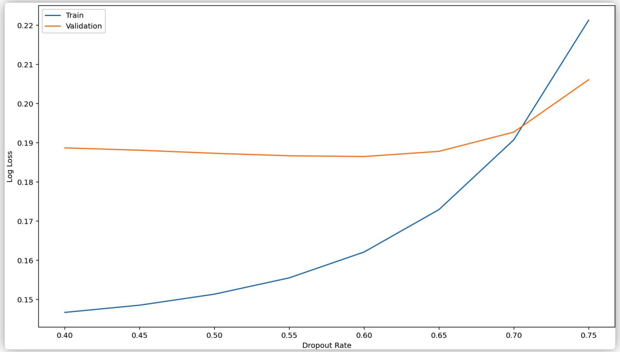
Experiments and Hyperparameter Tuning

1. Learning Rate
2. Epochs
3. Embedding Size
4. Dropout Rate
5. Weight decay









Tracking the training & validation loss with respect to these hyperparameters.

Experiments and Hyperparameter Tuning

1. Best Hyperparameters for Model 1:
   emb_size=70
   drop=0.6
   epochs=20
   learning rate =0.015
   weight decay= 1e-6

   Performance in test: 0.545

2. Best Hyperparameters for Model 2:
   emb_size=50
   drop=0.4
   epochs=20
   learning rate =0.002
   weight decay= 1e-6

   Performance in test: 0.410

Final Thoughts

- Using all features in a neural network, gave much more robust predictions.
- Giving all missing users/items a constant embedding to train on , greatly improved results.
- Negative sampling methods greatly affect the model learning and thus model performance.
- We still are unsure of how to pick a negative sampling method and avoid randomness in the model due to the same.