Predicting Implicit Rating

Addressing Cold Start Problem

- Shubham Thakur, Zihao Ren

INTRODUCTION

GOAL: Build a Implicit feedback based recommender to predict if the user is going to like a particular item

DATA DESCRIPTION

Dataset: Training data has 970245 observations having information including user_id and item_id for the users who liked particular item. In addition, its has context feature id which can viewed as feature related to users eg. Location, Nationality etc.

The second dataset has item_id and item_features id. Item features shows the category in which the particular item belongs. Ex. Genre of the item.

user_id	item_id	context_feature_id
0	28366	2
0	16109	2
0	11500	3

item_id	item_feature_id
0	139
1	55
2	11
3	138

IMPLICIT RATING & NEGATIVE SAMPLING

Problem:

- We don't have negative feedback(positive-only data)
- No level of Preference(Not like a 1-5 explicit rating)

Solution:

- Implemented User-Oriented and Popularity-Based negative sampling
- Sampled same number of items for a particular user as available in the initial training data

Before

	item1	item2	item 3	item4	item5
user1	1			1	
user2		1			
user3					1
user4			1		

After

	item1	item2	item3	item4	item5
user1	1	0		1	0
user2		1	0		
user3			0		1
user4		0	1		

MATRIX FACTORIZATION

Matrix factorization is a way to generate latent features when multiplying two different kinds of entities

LOSS FUNCTION: $$\frac{1}{N}\sum_{(ij),r_{ij}=1}^{N} y_{ij}\log u_i v_j + (1-y_{ij})\log\left(1-u_i v_j\right)$$

UTILITY MATRIX(Y)

	item1	item2	item3	item4	item5
user1	1	0		1	0
user2		1	0		
user3			0		1
user4		0	1		

N

M

USER VECTOR(U)

0.1	-0.03
0.4	0.9
0	0.03
-0.7	0.6

K

ITEM VECTOR(V)

0.3	0.01	-0.1	0	2
0.01	0.7	0.06	-0.01	0.3

FEATURE ENGINEERING

Item_features ID can be considered as the Genre of the Item(Movie). And Context feature ID can be looked as Nationality of the Users.

Item Feature ID

	ACTION	COMEDY	ROMANCE	THRILLER	DRAMA	
Item_feature_1	0.1	0.7	0.1	0.05	0.05	⇨ **Rom-Com**
Item_feature_2	0.05	0.2	0.05	0.6	0.1	⇨ **Com-Thrill**

Context Feature ID

	ASIAN	AFRICAN	EUROPEAN
Context_feature_id_1	0.1	0.7	0.2
Context_feature_id_2	0.0	0.2	0.8

DROPOUT FOR COLD START USER

Problem: More that 80% of the test observations are cold start users.

Solution: We trained one embedding vector specifically for cold start users. We did that by using technique of dropout. This can be achieved by randomly dropping users in every Batch and putting the user_id as 0. This made the model learn the average embedding for cold start user.

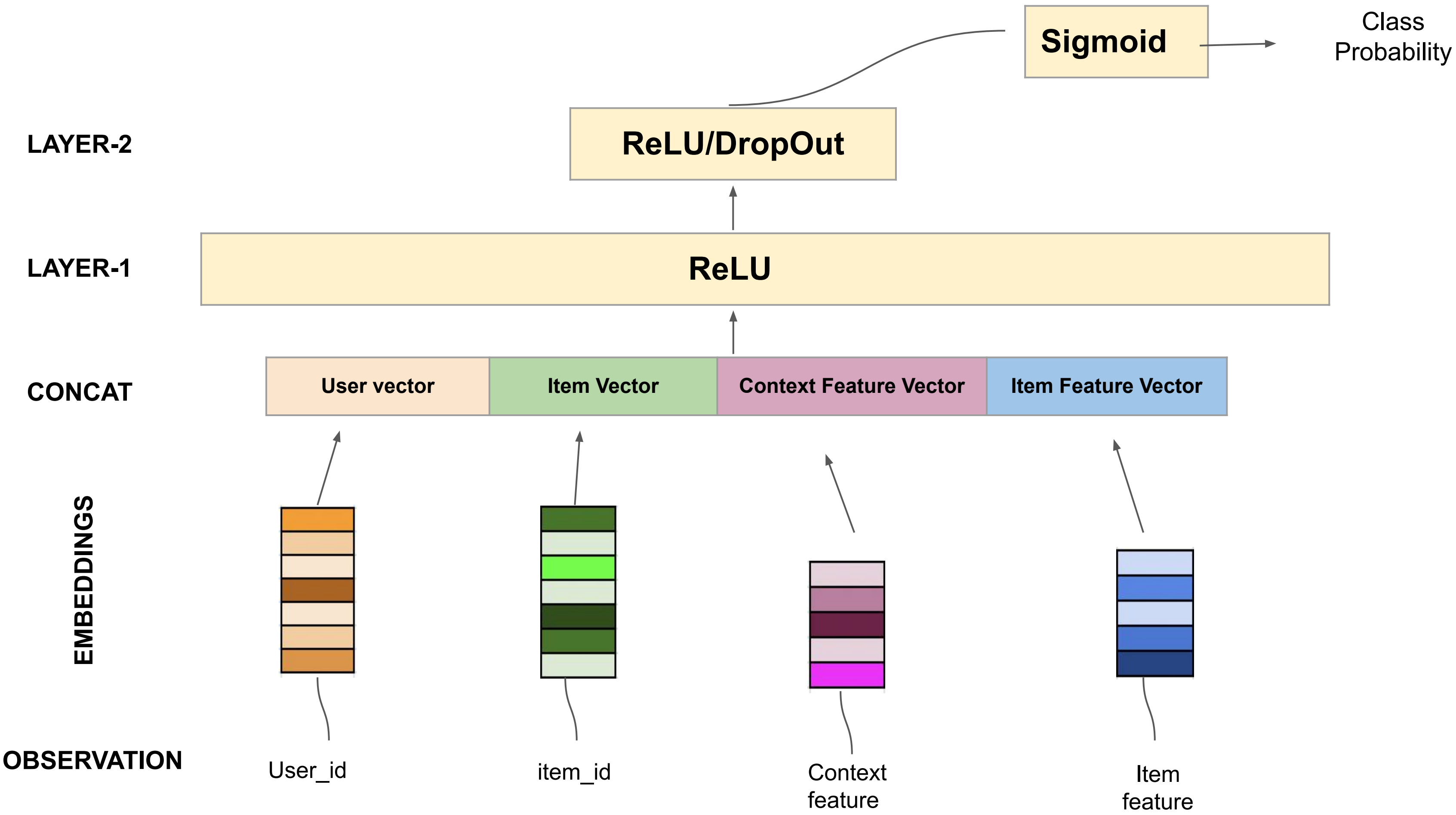
Pseudocode

Initialize: user emb f_U, item emb f_v
Repeat {NN/MF optimization**}**
    minibatch B = $\{(u_1, v_1) \ldots (u_k, v_k)\}$
    foreach $(u,v) \in B$ **do**
        do for randomly 0.2 frac i in 1..k
        $(u_i, v_i) \rightarrow (u_0, v_i)$
    endfor
    Update f_U and f_v using B
until convergence
output f_U and f_v

Cold start emb

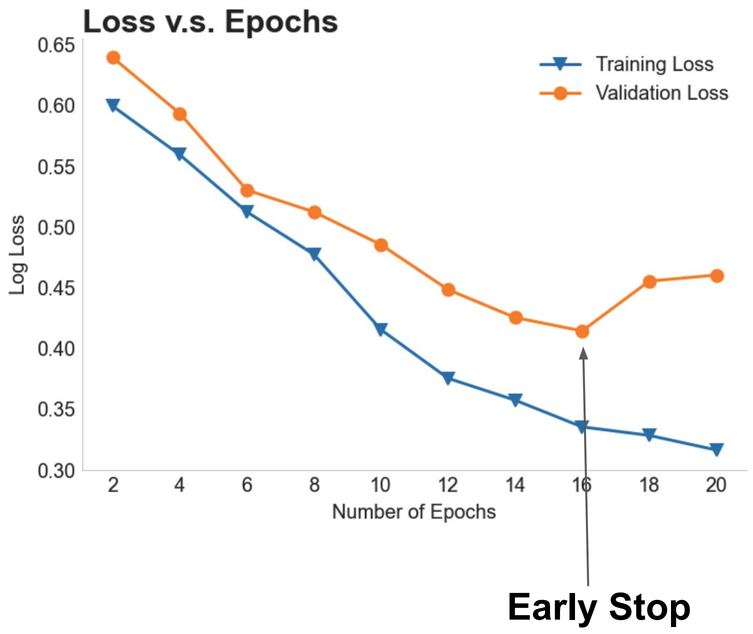
User_0	0.02	-0.123	0.45	0.557	0.3
User_1	0.1	0	0.4	-0.4	0.34
User_2		0.03		0.87	
. . User_k . .	. . .	. . .	. . .	. . .	. . .
User_n	0.3	-0.34	0.354	0.2	-0.09

MODEL ARCHITECTURE



HYPERPARAMETER TUNING-1

Early Stopping: a form of regularization used to avoid overfitting when training a learner with an iterative method. This drastically helped us improve our test accuracy.



- Training loss keeps on decreasing however, validation loss started to increase.
- Thus optimal epoch size in this case would be 16

HYPERPARAMETER TUNING-2

We performed multiple experimentation by performing grid-search on the following parameters

Dropout Rate(Cold Start Users): This shows the dropout rate while model training to train embedding for cold start users.
Learning Rate: determines the step size at each iteration while moving toward a minimum of a loss function.
Layer Size: determines the neuron in the intermediate layers in Neural Network
Embedding Size: Size of embedding layer in Neural Network

Hyperparameter	Optimal Value
Dropout Rate	0.3
Learning Rate	0.001
Layer Size	50, 5
Embedding Size	100

Based on Validation loss

Optimal Values are based on the results on Validation Dataset

Lessons Learned

- We learned how to build a Implicit feedback based recommender system from scratch in pytorch
- We explored various negative sample generation techniques like user-oriented, popularity-based and compared their performance
- Solving cold start user problem with the user dropout method
- Building Neural Network model based on features generated from matrix factorization
- Regularizing the model using techniques like early stopping and dropout.
- Hyperparameter tuning to generate the appropriate fitting model.