Your Name: Zihao ZHOU
Your Andrew ID: zihaozho
Your Nickname on Leaderboard: Jack

Homework 2

0. Statement of Assurance

1. Did you receive any help whatsoever from anyone in solving this assignment? Yes / No.

No.
If you answered 'yes', give full details? (e.g."Jane explained to me what is asked in Question 3.4").

2. Did you give any help whatsoever to anyone in solving this assignment? Yes / No.
If you answered 'yes', give full details? (e.g. "I pointed Joe to section 2.3 to help him with Question 2").

No.

3. Did you find or come across code that implements any part of this assignment? Yes / No.
If you answered 'yes', give full details? (e.g. book & page, URL & location within the page, etc)

No.

1. Corpus Exploration (10)

Please perform your exploration of the training set.

1.1 Basic statistics (5)

Statistics	
the total number of movies	5353
the total number of users	10858
the number of times any movie was rated '1'	53852
the number of times any movie was rated '3'	260055
the number of times any movie was rated '5'	139429

the average movie rating across all users and movies	3.3806

For user ID **4321**	
the number of movies rated	73
the number of times the user gave a '1' rating	4
the number of times the user gave a '3' rating	28
the number of times the user gave a '5' rating	8
the average movie rating for this user	3.1507

For movie ID **3**	
the number of users rating this movie	84
the number of times the user gave a '1' rating	10
the number of times the user gave a '3' rating	29
the number of times the user gave a '5' rating	1
the average rating for this movie	2.5238

1.2 Nearest Neighbors (5)

	Nearest Neighbors
Top 5 NNs of user 4321 in terms of dot product similarity	Userid: [980, 551, 3760, 2586, 90]
Top 5 NNs of user 4321 in terms of cosine similarity	Userid: [8497, 9873, 7700, 8202, 3635]
Top 5 NNs of movie 3 in terms of dot product similarity	Movieid: [1466, 3688, 3835, 4927, 2292]
Top 5 NNs of movie 3 in terms of cosine similarity	Movieid: [5370, 4857, 5391, 4324, 5065]

2. Rating Algorithms (50)

2.1 User-user similarity (10)

Rating Method	Similarity Metric	K	RMSE	Runtime(sec)
Mean	Dot product	10	1.0023961292822365	200.982228
Mean	Dot product	100	1.0066658184323152	2367.347831
Mean	Dot product	500	1.0429594605096932	11522.22382
Mean	Cosine	10	1.0630384126016648	210.8129843
Mean	Cosine	100	1.061981875237686	2568.374920
Mean	Cosine	500	1.075302048728639	10232.456660
Weighted	Cosine	10	1.0627279267794048	206.946684
Weighted	Cosine	100	1.0614932641773087	2096.644761
Weighted	Cosine	500	1.075302048728639	10422.22382

2.2 Movie-movie similarity (10)

Rating Method	Similarity Metric	K	RMSE	Runtime(sec)
Mean	Dot product	10	1.0204195215694447	152.101482
Mean	Dot product	100	1.0468579750854508	1503.688235
Mean	Dot product	500	1.1109235656275651	7455.823064
Mean	Cosine	10	1.0249396730214657	153.639574
Mean	Cosine	100	1.0497811613220471	1508.089965
Mean	Cosine	500	1.112130368317191	7467.099798
Weighted	Cosine	10	1.0228981033875464	150.639574
Weighted	Cosine	100	1.0130507784876977	1518.089023
Weighted	Cosine	500	1.112130368317191	7401.045988

2.3 Movie-rating/user-rating normalization (10)

Rating Method	Similarity Metric	K	RMSE	Runtime(sec)
Mean	Dot product	10	1.0110227494967678	987.381695
Mean	Dot product	100	1.057937710831789	9093.296447
Mean	Dot product	500	1.1243744512690956	39835.046337
Weighted	Dot product	10	1.0068749414366942	753.628285
Weighted	Dot product	100	1.0322696581835096	738.145401
Weighted	Dot product	500	1.1028121434348908	7434.348387

Clarification: I use AWS servers and my own laptop to compute the results in the above three tables. The use of CPU in the different tables is not the same and there is a significant difference in the computational capability of three types of CPU. It is meaningless to compare Runtime among different table but in one table the type of CPU is the same so it is reasonable to compare results within one table.

Add a detailed description of your normalization algorithm.

I used Vector Standardization method in Page 10 of '6,7-CF.pdf' TO normalize raw data, i.e.:

Initial Vector: $\vec{x} = (x_1, \cdots, x_n)$

Centering: $\bar{x} := \frac{1}{n}\sum_{j=1}^{n} x_j, \quad x_j^{'} := x_j - \bar{x}, \forall j$

Normalization: $\| x' \| := \sqrt{\sum_{j=1}^{n} \left(x_j^{'}\right)^2}, \quad \vec{z} := \frac{x'}{\| x' \|}$

Equivalently: $z_j := \frac{x_j - \bar{x}}{\sqrt{\sum_{j'=1}^{n} \left(x_{j'} - \bar{x}\right)^2}}, \forall j$

Clearly: $\bar{z} = 0, \quad \| \vec{z} \|^2 = \sum_{j=1}^{n} z_j^{\,2} = 1$

Vector Standardization

$x \to z_x$

$y \to z_y$

Similarity

$z_x \cdot z_y = \cos(z_x, z_y) = \mathrm{PCC}(x, y)$

PCC: the Pearson's Pearson's Correlation Coefficient

I choose to implement PCC on Movie-movie similarity calculation.

2.4 Matrix Factorization (20)

a. Briefly outline your optimization algorithm for PMF

In order to find good hyperparameter Lambda _U and Lambda _V, in the equation in the designated paper:

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}\left(R_{ij} - U_i^T V_j\right)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N} \| U_i \|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M} \| V_j \|_{Fro}^2, \quad (4)$$

I perform the following experiments and try to find a good choice of Lambda _U and Lambda _V to get a better result.

Several Experiments:

1) **Choose Lambda _U = 0.01and Lambda _V = 0.01**

Num of Latent Factors	RMSE	Runtime(sec)*	Num of Iterations
2	1.1784662112298945	658.115758	10300
5	1.1818282411420684	649.201816	10150
10	1.1852095142307766	694.127313	10850,
20	1.1851290071477092	729.131297	11400
50	1.1763543367258331	698.053141	10550

2) **Choose Lambda _U = 0.5 and Lambda _V = 0.5**

Num of Latent Factors	RMSE	Runtime(sec)*	Num of Iterations
2	1.1069990010644566	808.735603	11550
5	1.0849444647338247	756.7786675	10800
10	1.0933884174055906	689.956895	9850
20	1.1206563409452117	663.307155	9400
50	1.1421747564121638	751.094584	13200

3) **Choose Lambda _U = 1and Lambda _V = 1**

Num of Latent Factors	RMSE	Runtime(sec)*	Num of Iterations
2	1.1072320385344652	782.7085435	11400,
5	1.082203837762797	659.719021	9400
10	1.0919436405279466	653.757856	9350
20	1.113940593566384	631.490992	9000
50	1.140205172856903	871.9574205	12000

b. Describe your stopping criterion

I keep track of the loss and previous loss, once if the absolute value of loss and previous loss is less than 0.01, training iteration will stop. I even try to let all 5 training with different latent

factors to iterate until 50000 iterations. The result is not better than the former stopping criterion. Therefore, I think my stopping criterion is reasonable.

3. Analysis of results (15)

Discuss the complete set of experimental results, comparing the algorithms to each other. Discuss your observations about the various algorithms, i.e., differences in how they performed, what worked well and didn't, patterns/trends you observed across the set of experiments, etc. Try to explain why certain algorithms or approaches behaved the way they did.

1) For the first two experiments, the RMSE of User-user similarity is better than Movie-movie similarities. It can be explained that the number of users in training data is 10858, which is larger than the movie number 5353. Therefore, it may be more accurate to choose a KNN neighbor from a larger example data to predict rating.

2) For comparison the RMSE of the 'Mean' Rating method among different K value, the less K value it is, the less RMSE it gets from its prediction. It can be explained that the further neighbor is inaccurate not so useful to predict rating.

3) For comparison the RMSE of the 'Weighted' Rating method among different K value in the first two experiments, K=100 is better than K=10 and K=500. That is to say, increasing a reasonable size of data used for KNN with the 'Weighted' Rating method can improve the result. However, if K is too large, it will still lead to bad performance.

4) For comparison the PCC Movie-movie similarity with experiment 2. The result of PCC is slightly better than experiment 2. Eliminating the bias of the user can improve the accuracy of prediction.

5) For PMF, with the use of GPU, the speed of training is much faster than the above methods, especially with the as data volumes grow. It is much more suitable for large size search engine. However, I can't use this method to get a result which is better than results in experiments 1,2 and 3.

4. The software implementation (5)

Add detailed descriptions of software implementation & data preprocessing, including:

1. A description of what you did to preprocess the dataset to make your implementations easier or more efficient.

1) I use package pandas to read the Netflix data and fill the blank with 0 if there is any blank of the corresponding user-movie matrix to avoid 'nan' in the following calculation.

2) Using scipy sparse csc matrix to build the user-movie matrix and its sparse matrix multiplication method to speed up the training process.

'Corpus_Exploration.py' directly prints out the results of the Corpus Exploration session in this report.

'Experiment.py' is coded for Experiment 1. User-user Similarity.

'Experiment2.py' is coded for Experiment 2. Movie-movie Similarity.

'PCC.py' is coded for Experiment 3. PCC-based methods.

'PMF.py' is coded for PMF Matrix Factorization.

The matrix used in the above programs is from 'Create_matrix.py'.

2. A description of major data structures (if any); any programming tools or libraries that you used

I use package pandas to read the Netflix data(train, dev, test). This makes it easy for me to create the corresponding user-movie matrix.

For the PMF Matrix Factorization experiment, I used the GPU version Pytorch package to speed up training calculation.

3. Strengths and weaknesses of your design, and any problems that your system encountered;

Strength: This program is a modularity type. 5 programs are corresponding to 5 experiments respectively.
Weaknesses: I hard-code all 5 programs because of the limited time. Additionally, there is some duplicated calculation in the first three experiments and there is space for optimization.