# Your Name: Changhyun, Lee Your Andrew ID: changhyl Your Nickname on Leaderboard: CH

# Homework 2

**0. Statement of Assurance**

1. Did you receive any help whatsoever from anyone in solving this assignment? 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? No.  
If you answered ‘yes’, give full details? (e.g. “I pointed Joe to section 2.3 to help him with Question 2").

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

# 1. Corpus Exploration (10)

Please perform your exploration on 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 | 3.3805 |

|  |  |
| --- | --- |
| 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 | 3.1506 |

|  |  |
| --- | --- |
| 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 | 90, 551, 980, 2586, 3760 |
| Top 5 NNs of user 4321 in terms of cosine similarity | 3635, 7700, 8202, 8497, 9873 |
| Top 5 NNs of movie 3 in terms of dot product similarity | 1466, 2292, 3688, 3835, 4927 |
| Top 5 NNs of movie 3 in terms of cosine similarity | 4324, 4857, 5065, 5370, 5391 |

# 2. Rating Algorithms (50)

## 2.1 User-user similarity (10)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rating Method** | **Similarity Metric** | **K** | **RMSE** | **Runtime(sec)** |
| Mean | Dot product | 10 | 1.0032 | 56.22 |
| Mean | Dot product | 100 | 1.0086 | 135.81 |
| Mean | Dot product | 500 | 1.0459 | 491.40 |
| Mean | Cosine | 10 | 1.0634 | 54.67 |
| Mean | Cosine | 100 | 1.0622 | 135.81 |
| Mean | Cosine | 500 | 1.0755 | 491.40 |
| Weighted | Cosine | 10 | 1.0631 | 54.67 |
| Weighted | Cosine | 100 | 1.0617 | 141.40 |
| Weighted | Cosine | 500 | 1.0740 | 496.28 |

## 2.2 Movie-movie similarity (10)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rating Method** | **Similarity Metric** | **K** | **RMSE** | **Runtime(sec)** |
| Mean | Dot product | 10 | 1.0203 | 39.72 |
| Mean | Dot product | 100 | 1.0473 | 183.62 |
| Mean | Dot product | 500 | 1.1118 | 572.83 |
| Mean | Cosine | 10 | 1.0177 | 37.69 |
| Mean | Cosine | 100 | 1.0642 | 216.91 |
| Mean | Cosine | 500 | 1.1185 | 662.62 |
| Weighted | Cosine | 10 | 1.0152 | 37.26 |
| Weighted | Cosine | 100 | 1.0571 | 131.46 |
| Weighted | Cosine | 500 | 1.1024 | 552.92 |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rating Method** | **Similarity Metric** | **K** | **RMSE** | **Runtime(sec)** |
| Mean | Dot product | 10 | 1.0028 | 84.13 |
| Mean | Dot product | 100 | 1.0082 | 189.65 |
| Mean | Dot product | 500 | 1.0445 | 615.15 |
| Mean | Cosine | 10 | 1.0648 | 62.77 |
| Mean | Cosine | 100 | 1.0635 | 165.33 |
| Mean | Cosine | 500 | 1.0766 | 576.94 |
| Weighted | Cosine | 10 | 1.0645 | 61.03 |
| Weighted | Cosine | 100 | 1.0630 | 145.16 |
| Weighted | Cosine | 500 | 1.0752 | 545.40 |

Add a detailed description of your normalization algorithm.

* The user-movie matrix is centered from -2 to 2.
* To calculate the cosine similarity, the dot product similarity is divided by L2 norm of each user/movie vectors.
* Pearson Correlation Coefficient is calculated by replacing the user/movie vector X to (X - X\_mean). After that, the cosine similarity is applied to get PCC similarity.

## 2.4 Matrix Factorization (20)

1. Briefly outline your optimization algorithm for PMF
2. Describe your stopping criterion

|  |  |  |  |
| --- | --- | --- | --- |
| **Num of Latent Factors** | **RMSE** | **Runtime(sec)\*** | **Num of Iterations** |
| 2 |  |  |  |
| 5 |  |  |  |
| 10 |  |  |  |
| 20 |  |  |  |
| 50 |  |  |  |

# 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.

* Cosine similarity seems like it does not affect the performance of rating algorithm because the RMSEs according to the similarity matrix are pretty much similar.
* Weighted rating method reveals that has higher RMSE, except for the case of movie-movie similarity.
* The more K larger, the more the RMSE is increased.
* The running time is increased when K is getting larger
* Pearson Correlation Coefficient is calculated by replacing the user/movie vector X to (X - X\_mean). After that, the cosine similarity is applied to get PCC similarity.
* When I tried to calculate the Pearson Correlation Coefficient, I found that the RMSE is decreased if the X\_mean has lower value. For example, we can choose the X\_mean by averaging 10916 values with assuming missing values also have ratings of 3, not by averaging only existing rating values.

# 4. The software implementation (5)

# Add detailed descriptions about software implementation & data preprocessing, including:

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

- Similarity matrix preprocessed, by using user-item matrix multiplication. Through this preprocessing, we can get more efficient algorithm.

- The csr matrix from scipy is used to resolve computational cost, especially in user-item matrix multiplication that I mentioned above.

- Modularization of algorithm such as User\_user\_similarity and Movie\_movie\_similarity, knn, and so forth. We reduced the redundancy of codes by using these kinds of modules. Simply we can choose the parameters such as K, metric(dot product or cosine), rating method(mean or weighted).

- Exceptional input is handled with a notification of ‘invalid input’.

- PCC is efficiently calculated by replacing the user/movie vector X to (X - X\_mean), using the user-user similarity function.

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

- Scipy csr\_matrix is used to represent sparse matrix

- Numpy library is used to efficient matrix multiplications.

- Tools : Pycharm Community Edition (2019.3.2.)

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

- This program can automatically run and implement all the experiments.

- We used sparse matrix, by using the library of scipy.csr\_matrix.

- We calculated normalization factor such as L2 norm for the cosine similarity in advance so that to improve computational efficiency.

- However, we need to develop this program further, to reduce redundancy of codes used to make output of all experiments.