HW4: Recommender Systems

Name: Yaswanth Reddy Manukonda
Email: ymanukon@gmu.edu
Miner2 Id: ymanukon
Mason Id: ymanukon
RMSE: 0.90
Rank: 160

Problem Statement:
Develop a recommender system with the help of given datasets to predict the rating for the user-movie pairs in the test data

Solution:
1. Load the given datasets using pandas library
2. Pre-process the datasets to remove null values if any and create the user (user ID vs Movie ID) and movie (movie ID vs Genre name) matrices from the given datasets and
3. Perform collaborative filtering for calculating the similarity of the given user to all the other users available in the training dataset and measured the root mean squared error
4. Other way of approaching this problem is to calculate the average rating a user has given to all the movies in a particular genre and marked as 1 if none of the movies in that genre are rated. This may have a problem of user have not seen a movie for which by default a rating of 2.5 is given.

Explanation:
- During the pre-processing, I have removed the null values present in the data and created the user item and movie item matrices

	1	2	3	4	5	6	7	8	9	10	...	64986	64990	64993	64997	65011	65037	65088	65126	65130	65133
75	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
78	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
127	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
170	3.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.5	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
175	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
71487	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
71497	0.0	3.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
71509	4.0	0.0	0.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
71525	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
71529	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

2112 rows × 9936 columns

Movie_ID	Crime	Film-Noir	Thriller	IMAX	Musical	Mystery	Western	Action	Drama	Sci-Fi	Documentary	Short	Comedy	War	Romance	Horror	Adventure
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
5	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
65088	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
65091	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
65126	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
65130	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
65133	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0

10197 rows × 20 columns

  - These matrices are further used in the process of collaborative filtering to identify the users whose ratings to the movies are like the user X (Here, the user X is all the users from the test data who have rated movies)
  - To calculate the similarity between these users in the collaborative filtering I have tried using cosine similarity, and Euclidian distance but it turned out that these calculations are time consuming, and the root mean squared error is high.
    - I have used scikit learn library to calculate these distances
    - Euclidian distance: It measures the distance between the point in the two n dimensional arrays.
    - Cosine similarity: it is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.
    - Cosine Distance: is the ***1- (cosine similarity)***. It takes two n dimensional arrays as parameters.
  - After calculating these similarities, I have tried considering the average rating of the 7 nearest users as the rating of a movie for that user.
  - The same has been performed for the new or unseen movies of a user.
  - However, this entire method is not giving me a good, root mean squared error.
- So, I tried using the movie genre, train and test datasets which is giving me a root mean squared error of 0.90.
  - In this approach, I have calculated the average rating a user has given to all the movies in a particular genre.
  - By default, it is marked as 1 if none of the movies in that genre are rated by that user.
  - This approach has a problem of rating a user-movie pair for which the user has not seen a movie. for this by default a rating of 2.5 is given.

- As part of validating these two approaches and to calculate the root mean squared error, I have picked one movie for each user in the training dataset with help of *drop_duplicates* method in pandas data frames which returns 2113 records. These records are used to validate the predictions.
    - I have used *mean_squared_error* method to calculate the mean squared error rate. This method takes predicted values and actual values as two parameters. Square rooting the returned value gives us the root mean squared error rate

References:

- Prabhakaran, S. (2021, October 13). *Cosine similarity - understanding the math and how it works? (with python)*. Machine Learning Plus. Retrieved November 10, 2021, from https://www.machinelearningplus.com/nlp/cosine-similarity/.
- Wikimedia Foundation. (2021, October 29). *Collaborative filtering*. Wikipedia. Retrieved November 10, 2021, from https://en.wikipedia.org/wiki/Collaborative_filtering.
- *Lecture 43 — collaborative filtering - youtube*. (n.d.). Retrieved November 10, 2021, from https://www.youtube.com/watch?v=h9gpufJFF-0.
- Luo, S. (2019, February 6). *Intro to Recommender System: Collaborative filtering*. Medium. Retrieved November 10, 2021, from https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26.
- *SKLEARN.METRICS.MEAN_SQUARED_ERROR*. scikit. (n.d.). Retrieved November 10, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html.