Joel Nail
Alex Yu

Project 1 Report

<u>Features Used for Recommendation</u>

- Song ratings were calculated on a per-user basis – for any given user (x) with list of songs (s) and each song's number of plays (n):
    - $song\ rating = \frac{n}{\Sigma x(s(n))}$
    - $i.e., song\ rating = \frac{Number\ of\ Plays\ for\ Given\ Song\ Listened\ to\ by\ Given\ User}{Total\ \#\ of\ Plays\ for\ All\ Songs\ LIstened\ to\ by\ Given\ User}$
- In order to recommend five songs to a given user, we first look at the song that the given user has listened to the most
    - We chose user = ' 62543798a0eac1d057af421b188dc72d6fd48ff2' whose highest rated song = 338648 with rating = 0.192
- We then find all other users who have listened to the given song and then isolate only the rows relating to the given song. Using these rows, we find the user who has the highest rating for the given song
- We then analyze all songs listened to by the user selected in the prior step and suggest the top five songs for this user (not including the given song)
- This method might be improved by taking the top song (not including the given song) from the top five users who listened to the given song rather than taking the top five songs from the user who gave the given song the highest rating. This would introduce recommendations from a variety of users rather than a singular user while still ensuring that the given song's rating is considered

<u>Code Flow</u>

- **Block 3:** this code uses the song_to_index dictionary to create a new RDD after replacing all song ID's with their corresponding index value
- **Block 4:** this code creates an RDD that only contains songs with user listening history
- **Block 5:** this code creates an RDD of all songs without user listening history by taking an RDD of all songs and subtracting the RDD of all songs with listening history

- **Block 7:** this code creates an RDD grouped by each user ID coupled with a list of lists where each sublist contains a song the user has listened to and the number of plays for that song
- **Block 8:** this code calculates the relative rating for each song within each user's list of songs/ratings using the formula described in the above section
- **Block 10:** this code creates an RDD of all users who have listened to the given song (not including the given user)
- **Block 11:** this code reformats the RDD created in the last step so that each song/rating pair is in its own list combined with the user who gave that rating. This format is used to isolate only those ratings that are for the given song. This RDD is then sorted by rating so that the first row is the user who gave the given song the highest rating
- **Block 12:** this code find the user ID for the user who gave the given song the highest rating (highest rater) and then finds all songs listened to by that user, rated by popularity (rating)
- **Block 13:** this code removes the given song from the list of songs listened to by the highest rater and then recommends the remaining top five songs listened to by that user
- **Block 15:** this code creates lists from our RDDs for easier manipulation and also establishes the dot and cosine_similarity functions
- **Block 16:** this code creates a list of lists with the following format: [user [song1, song 2]] i.e., we remove the rating and put the songs into a single list
- **Block 30:** this code first takes a random sample of users and then defines the function for creating the user interest matrix (detailed explanation of this function can be found within our notebook). The code then calculates cosine similarity for each pair of users
- **Block 31:** this code finds the highest user similarities within the user_similarities matrix and combines it with the index values of the user and the user they were compared to
- **Block 34:** this code defines dictionaries to convert from user ID to user index which allows us to search for most similar users with user ID. Code was borrowed

from the RecommenderSystem notebook to find the most similar users for a given user. The code also ensures that only the top five most similar users are selected

- **Block 35:** for the top five most similar users, this code creates a flattened RDD of all songs that the users have listened to with a rating above a certain threshold. These are then collected and added to a list
- **Block 36:** this code sorts through the previously created list and then recommends the five songs with the highest rating