Introduction

This report is based on a song recommendation system. It includes two major datasets:

1. Triplets dataset: This includes user id, song name and play count
2. Songs dataset: This includes song name and song id

Computing Song Ratings

We find that the song ratings based on only play count gives low values. Hence, I decided to use the user data on how many times each song has been played by the user. To obtain this, I followed the below steps:

1. Total RDD - For each user, I found the total play count of all songs that the user had listened to.
2. I joined the total RDD to triplet RDD such that each row shows user ID, Song ID, play count for each song and total play count for all songs listened to by the user.
3. Rating RDD –
    a. Rating = Play count for each song listened to by each user / total play count for all songs listened to by each user
    b. I replaced the total play count column with calculated rating (as per above formula which gives value between 0 and 1)
4. Rating_Triplet RDD - I restructured the data format and named it rating_triplet

Top 5 Recommended Songs for a User

1. I collected given user id, given song id and rating as the input.
2. Define liked function, which checks for users who have liked the same given song id and given a rating higher than the input rating.
3. Users RDD: RDD of similar users who have liked the same song as input with higher rating
4. Joined the users RDD with the Rating Triplet RDD to obtain the list of songs, listened to by these users. I used the filter function to filter out the input song, so that it does not show up in list of recommended other songs.
5. To obtain top recommended songs, I aggregated the total rating across songs (Note: this will give a value higher than 1 for total aggregated rating: the purpose of this is only to get the top recommended songs)
6. I ran a TakeOrdered command to obtain top 5 recommended songs to the user.

Top 5 Most Similar User Pairs: Cosine Similarity

1. Rating_Triplet – In order to obtain a mapping of each user with all users, I joined the rating_triplet RDD with itself. I have used 1 as the key in both these RDD's so that every user is mapped to all other users.
2. To find similar users, I defined a checkSimilar function which finds other users for every user who have liked the same song.
3. User_pair RDD – This RDD contains pairs of users with their ratings for every song they have listened to.
4. Finding Cosine Similarity
    a. To find cosine similarity I need the dot product of ratings for all songs listened to by a user set. I also need the normalized value of ratings given by each user for all those songs.

b. Cosine Similarity = DotProduct(A,B) / Norm(A)*Norm(B)
c. I calculated the numerator and denominator values for cosine similarity and created a new RDD which has the user set with the calculated cosine similarity.
d. Cos_rdd – This RDD has each user set as the key and the corresponding cosine similarity as the value.
e. Cos_sorted RDD – I sorted cos_rdd in descending order to obtain user set with the highest similarities.
f. cos_temp RDD – I used this RDD to obtain distinct values of users in the first column of the user set.
g. Cos_temp1 RDD – I used this RDD to obtain distinct values of users in the second column of the user set.
h. Now I have an RDD which has distinct values of users in both columns of the user set. I want to ensure distinct values of users across both columns so that each user appears only once in the output.
i. Cos_temp2 RDD – This RDD contains user1 as the key and user2 & corresponding cosine similarity as the value pair.
j. Cos_temp3 RDD - This RDD contains user2 as the key and user1 & corresponding cosine similarity as the value pair.
k. L1 RDD – This RDD was used to obtain rows only present in cos_temp2
l. L2 RDD - This RDD was used to obtain rows only present in cos_temp3
m. I found that if a key is present in both RDD's, it is not present either in l1 or l2 RDD. This means I need to find the intersection of cos_temp2 & cos_temp3 to obtain common rows.
n. L3 RDD – This RDD is used to obtain rows which have keys common to both cos_temp2 & cos_temp3. I obtained distinct rows from this RDD (This gives us unique rows for keys which have the same user set. For e.g. ('User6','User3') & ('User3','User6').
o. Cos_Final RDD - I combined l1, l2 and l3 RDD using union function to obtain the desired output. Then, I used the takeOrdered function to obtain the top 5 unique user set with the highest cosine similarity.
p. Top_users RDD - I inputted a given user ID and filtered all user sets which have this user ID, to obtain top 5 similar users in the user sets.
q. I joined the top_users RDD with the original rating_triplet RDD to obtain the songs listened to by these similar users.
r. Finally, I used a distinct function to find unique songs and generated song recommendations for the given user.