Code ▾

A6 - Code Review for Ritvika-Xia

Futai-Shikha

Code Review:

- They did a good job while coding with scala. The code is easy to read. They understand operation of spark in scala well and performs each task well and get a good result, although there are still some mistakes when they coded.
- From our perspective, since the authors did filter function at the first line of every task to get rid of empty lines, they could have filtered the lines in the beginning that will avoid writing the same filter function at every single task.

```
val totalSongs = songInfo.filter(record => !record(TRACK_ID).isEmpty).
  map(record => record(TRACK_ID)).distinct().count()

val totalArtists = songInfo.filter(record => !record(ARTIST_ID).isEmpty).
  map(record => record(ARTIST_ID)).distinct().count()

val totalAlbums = songInfo.filter(record => !record(ALBUM_RELEASE).isEmpty).
  map(record => record(ALBUM_RELEASE)).distinct().count()

val top5Loudest = songInfo.
  filter(record => !record(TRACK_ID).isEmpty && Try(record(LOUNDNESS).toDouble).isSuccess).
  map(record => (record(TRACK_ID), record(LOUNDNESS).toDouble)).sortBy(_._2,false).take(5)

val top5Longest = songInfo.
  filter(record => !record(TRACK_ID).isEmpty && Try(record(DURATION).toDouble).isSuccess).
  map(record => (record(TRACK_ID), record(DURATION).toDouble)).sortBy(_._2,false).take(5)
```

- Moreover, they did a good job in reading file using textFile, and for some tasks, they perform parallelize function to parallelize data.
- The report we looked at, was committed at 3:11 pm. (CommitId: 493fbe8b75b307c78410902245ff3d19142f936b).

Commit details:

```
commit 493fbe8b75b307c78410902245ff3d19142f936b (HEAD -> master, origin/master, or
igin/HEAD)
Author: ritvikareddy18 <ritvikareddy18@ccs.neu.edu>
Date:   Fri Oct 27 15:11:13 2017 -0400

    Report



Last commit before 2pm:

commit b3aa32e9ee2d1a82bad7f1c84f68ce38089ec52a (HEAD)
Author: ritvikareddy18 <ritvikareddy18@ccs.neu.edu>
Date:   Fri Oct 27 13:55:03 2017 -0400

    Time counter
```

The previous version of the report (before 2:00pm) just mentioned the results for the top five common words. However, the results are incorrect since they have included the following words:

```
(ALBUM
(LP
&
```

- No results found for the number of distinct songs, albums and artists. Also, When they dealt with these (in the source code), it is more efficient if they wrote reduceByKey function rather than distinct function.
- They compare the efficiency of using persist and not using persist which dive more into the understanding of spark principle as persist might not improve in standalone mode but will improve efficiency if the data set is run in distributed system.
- The name of the .scala class file is “untitled”. It could be more descriptive. Also, there is no description or documentation in the file about it does. It also has commented code.
- Results for the following tasks are present in the output folder (code), but not mentioned in the report:

```
Top 5 most popular keys (must have confidence > 0.7)
Find top 5 hottest Genres
Top 5  familiar artists
```

- System specifications not mentioned in the report.
- Results for the common words are incorrect. The question also asked for only 5 common words. Some of the incorrect results from the results are:

```
VERSION),55507
(ALBUM,26075
-,13811
REMASTER),10049
&,9471
I'M,6846
/,4928
(LIVE,4243
(LIVE),7647
2,3837
```

Moreover, while filtering the common words, they did not consider the bracket which leads to the wrong final result. To solve this problem, they could have added more filtered elements in the object “ignored” like “)”and “(”, etc to retrieve an accurate result.

- Results for prolific artists are as follows in the output folder:

```
AR0O3AV1187FB4D030,9
ARPDVPJ1187B9ADBE9,9
ARI648V1187B9B5379,10
AR6PJ8R1187FB5AD70,11
ARNLO5S1187B9B80CC,9
AR0X3JJ1187FB3A9A7,9
AROF4LP1187FB41C51,10
ARDVZTE1187FB5A0A1,10
ARODBRG1187FB3FD99,11
AR9W3X91187FB3994C,12
AR65K7A1187FB4DAA4,9
ARVUN5F1187FB4CCC7,9
ARCII0J1187FB3A1B4,9
ARL3VGT1187FB40E8E,10
ARRXPRY1187B9A8B34,9
AREWQSE1187B9AEC6C,12
AR8BMEQ1187B9B4214,10
ARO0R521187B98A2D1,9
ARD3LXU1187B9ABFC5,9
AROIHOI122988FEB8E,13
AR051KA1187B98B2FF,10
ARY1P2B1187B9B78F5,9
ARD3ICK1187B9A56C9,9
ARAIABB1187B9AC6E2,9
ART3O5Z1187B9AB043,11
ARIRD6J1187FB5A98C,12
ARXPPEY1187FB51DF4,10
ARYDFJA1241B9CBFF2,10
ARGBGC71187FB3DD0B,9
ARH6W4X1187B99274F,11
ARZBYDY1187FB46DD7,9
AR78ZID1187B9B31ED,11
ARBB58Y1187B9B621B,9
ARSQDUN1187B98D7D7,9
ARVML4B1187FB52324,9
AR12F2S1187FB56EEF,12
ARCCRTI11F4C845308,10
ARJ9DSA1187B990E00,10
ARBEOHF1187B9B044D,9
ARRH63Y1187FB47783,9
ARE8GLF1187FB52532,10
ARD7U0O1187FB5531A,9
AREPD8D1187FB3F5BC,9
ARYF20K1187B9B76BD,9
ARX9YIP1187B98A656,12
ARMPRI31187FB4F7D9,9
ARVEJ9M1187FB4DC44,9
ARJIE2Y1187B994AB7,12
AREJ5K11187B993F5F,9
AR03BDP1187FB5B324,10
ARDY3451187B9A0226,9
AR7KA5V1187FB44E6B,10
ARVN9FZ1187FB393F1,12
AR6CLFP1187B9ACB94,9
ARN03F71187FB4E3F4,10
```

The question asked for only top 5 values. We matched it with our results and we believe that the correct values should be:

```
artist_id,count
AROIHOI122988FEB8E,13
AR9W3X91187FB3994C,12
AREWQSE1187B9AEC6C,12
ARIRD6J1187FB5A98C,12
AR12F2S1187FB56EEF,12
```

- Most of the results doesn’t define what value is in the output, that is, headers missing. For instance, the results for the fastest songs are:

```
TRBGYHC12903D0626A,262.828
TRBHZRN128F92EFF91,258.677
TRAGYJM128EF3466D2,253.357
TRBFXLH12903CBAFAD,248.079
TRAZFZS128F4272A31,246.593
```

It would be better if they could have mentioned the track_id of the songs is the id’s mentioned in the results.