Challenge 3: Million Song Dataset

The Million Song Dataset [1] contains metadata for one million commercially-available songs.

Your task is to inspect a subset of the MSD (consisting of 10000 songs) using Hadoop or Spark and do the following tasks:

TASK 1: The loudness war [2] is a well-known trend of releasing recorded material with increasingly higher audio levels. Your job is to expose this trend.

- Plot the evolution of the loudness war (average loudness as a function of the year of release).
- If done correctly, the the left portions of the graphs will be spiky. Why is that? Write your answer in the readme file. (Hint: it has nothing to do with the music itself.)
-

TASK 2: Minor keys have traditionally been reserved for songs which seek to convey sadness or melancholy. As such, we can reasonably expect minor keys to be correlated with low tempos and reduced loudness. *Is that true?*

- Back up your conclusion with graphs that show the evolution of the relevant characteristics side-by-side for major and minor keys.
- Hint: the song property that differentiates between the two is called "MODE" (0 means minor, 1 means major)

TASK 3: Sort the genres by:

- number of songs in the dataset
- average loudness
- average tempo

TASK 4: *Attempt* to quantify the tendencies for crossover between the genres. The result should be in the in the form of an N x N crossover matrix (where N is the number of genres). Cell (x, y) contains a numerical value denoting how much of a crossover there is between genre x and genre y. The main diagonal of the matrix (x = y) should contain the maximum possible value (a genre is always 100% "contaminated" by itself).

- It is up to you to come up with a crossover heuristic. Explain and motivate your choice in the readme file.
- A sensible armchair [3] solution is enough. Demonstrating the soundness of your heuristic would require some musical knowledge, as well as manually going through a large part of the dataset.
- Hint: Take a look at how GenreDetector was implemented.

ALTERNATIVE TO TASK 4: If you have an idea and would like to do something else with the data set, e-mail your TA. Your proposed alternative must be non-trivial; it should be in the same difficulty range as task 4 or higher.

About the data set:

- Get the subset in plain text from here: http://nets.cs.pub.ro/~vlad/atds/challenge3/MillionSongSubsetConverted.tar.gz (mirror: https://drive.google.com/file/d/0B-DbK_GRSgp7RnlweXR2VTE2d0k/view). Masochists can download the HDF5 version (http://static.echonest.com/millionsongsubset_full.tar.gz) and convert it to plain text (http://nets.cs.pub.ro/~vlad/atds/challenge3/hdf5-to-text.tar.gz, mirror: https://drive.google.com/file/d/0B-DbK_GRSgp7MlVnTHRFMWdlSDg/view)
- Use the following library to parse it: http://nets.cs.pub.ro/~vlad/atds/challenge3/util.tar.gz (mirror: https://drive.google.com/file/d/0B-DbK_GRSgp7aUtCUWIxMzR5VHM/view)
-

Implementation hints

- Imagine that you are going to process the entire dataset; as such, you need to be performance-oriented.
- You are expected to always use the most efficient key/value data types. For example, comparing integers (IntWritable) is much faster than comparing strings (Text) that depict the same numbers, even though the end result is the same.
- Enums have ordinals.
- Sorting and computing averages can (and should) be done outside of Hadoop. As seen in Lab 6, computing averages with Hadoop requires forgoing the use of a combiner or writing different code for the combiner and the reducer or complicating your data structures (neither of which desirable when the number of keys is small).
- For each of the first three tasks you're only allowed to read the dataset from the disk once. This does not apply to task 4.
    - For those who use Hadoop, this means one job per task.
    - For Spark users, this means that any RDD query (i .e. collect()) aside from the first one should only involve cached data.
    -

Submission

You should submit an archive containing the following:

- source code
- output files
- plots
- a readme file with an emphasis on task 4

[1] http://labrosa.ee.columbia.edu/millionsong/
[2] http://en.wikipedia.org/wiki/Loudness_war
[3] armchair = a pejorative modifier to refer to a person who experiences something vicariously rather than first-hand, or to a casual critic who lacks practical experience