Music Mining

Anonymous

1. OVERVIEW

For this project we are using the Million Song Dataset(MSD)[2] which will be in the music domain. Due to the vastness of the dataset, a subset of 10,000 random songs will be used. We plan on using MySQL to store our dataset and interface it with R for data mining. We would be making use of python scripting in order to read data from provided HDF5 files and put it into MySQL database. We are using making use of HDFView tool[3] provided by the HDF5 Group to view the data in the files.

There are many approaches to mining this dataset. In a broader sense, this dataset can be used to create groups of related artist or songs. This could be done with clustering of the data, and classification of 'new' data into the predefined clusters. This dataset also provides tags that users have associated with a particular song, tag prediction based on other attributes could be an application to this dataset.

This project's main scope is to provide song and artist recommendations. This would prove useful to music streaming sites to keep the listener engaged in their services.

2. DESIGN CONSIDERATIONS

Our first draft of the dataset's separation was with two main entities identified; Song and Artist. From the data provided in the dataset, this was the most intuitive way to categorize the data. The set contains song specific attributes and artist specific attributes. The song and artist entity holds a relationship, were the specific cardinality is assumed to be 1 artist to many song titles (where many contributing artists are identified as a separate artist).

Our current draft (described in section 3) of the dataset contains more entities. Due to the set of strings collected under some attributes making them non-atomic, this change was made from the two main tables; song and artist. To lessen the scope, some attributes were also eliminated for we deemed them as unnecessary. These attributes may be considered later in future analysis.

3. ARCHITECTURE

Figure 1 depicts the schema we have created for the million song dataset. We have created database tables using MySQL Workbench and then applied the built-in reverse engineering tool to generate this model. All the primary and foreign key relationships have been added. We haven't added any stored procedures or triggers as the data is going to be static for this project. We might add a few views based on requirement as we proceed further in the project.

4. IMPLEMENTATION

Once the data is loaded into R, initial data analysis can begin. First, the key attributes are identified by either correlation or some other methods to limit the scope of the initial blind analysis. When key features are identified the eliminated attributes would be integrated back in to compare findings.

For artist and song recommendation clustering is the most viable approach. Artists or songs in the same cluster will pose as related. Different clustering methods would be used to find the closest method related to the similar artists field the dataset provided. Songs will be based off of that same approach.

A possible application for this project is to create an interface for navigating the dataset with the option of recommendations based on a supplied song or artist query. The plan is to have geographical markers for each artist, hopefully with more distinguishable markers such as genre. This will provide a visual of music trends in respect to geographical domain.

A python script has been created that reads data from all hdf5 files. Data from these files is separated according to the schema adn put into separate lists. Another script takes these lists as input and inserts the values into corresponding database tables. Python lists are saved in the main memory and as the size of the data increases, the size of the list also increases. Hence, this approach is not scalable to very large datasets.In order to fix this issue, we have decided to read data from each hdf5 file and immediately push it into the database. This way, there is only one file worth data saved in the memory at any give time. This approach is more scalable for very large datasets.

5. LESSONS LEARNED

The time it takes to actually understand and interpret the data in the dataset was an under-estimation. Actually setting up the data into a preferred format was an obstacle not predicted before selecting this dataset. For instance, only one member was able to get the Python environment set up to extract the data from the dataset.

In the Big Data world, actual understanding of the data isn't as important as the recognition of relationships, dependencies, and attribute form. Since we have yet developed that 'sense',interpreting the data in its basic form had time set aside.

When using analysis tools such as R and Rattle it is necessary to interpret results at every step of the process as a sanity check. This will help ensure the usage of the correct

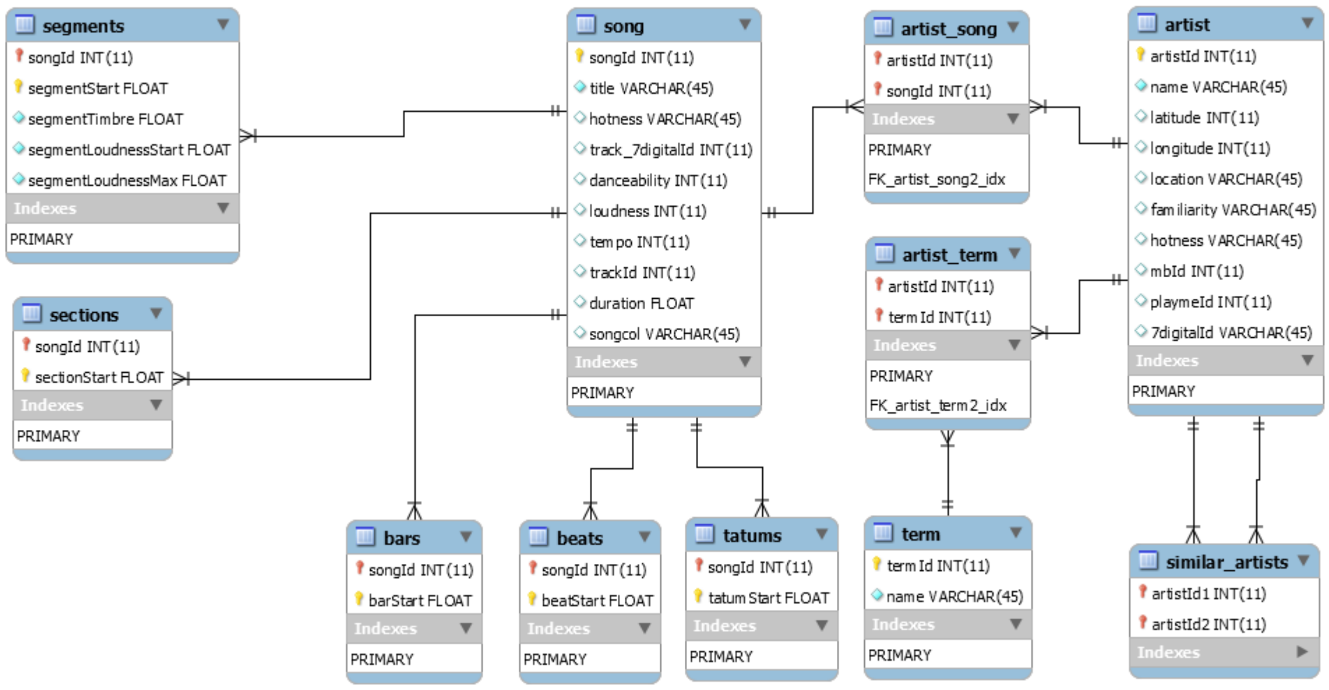

Figure 1: Schema diagram for the database (generated using MySQL)

data with finer tuning thus leading to more accurate results (refer to section 6.1.2).

Another obstacle we faced was to get the MySQL connector for python to work on our machines. Mac OS X has various versions of python installed and we had a hard time associating the MySQL connector to the version of python we are using. Finally, we were able to find a work around by installing a python module called PyMySQL.

6. CURRENT STATUS AND FUTURE WORK

6.1 Current Status of Project and Next Steps

6.1.1 Database Import & Management

At the moment the ER relationship model is up to date. Changes might be made after understanding the data more once the big table model is formed and the decided ER model starts into implementation.

Currently the dataset has been cleaned and converted to be added to the MySQL DBMS. The dataset was provided in hdf5 format. A Python interface[1] is used to retrieve data from the file format and another Python script has been created to use this interface and translate this data to a more easily readable 'big table' format.

So far we have created a script that iterates over all h5 files and generates a CSV with all the data (one big table). We have also created the script that reads data from the h5 files and pushes them into the database tables. Next steps are normalizing the data within the MySQL database. The relational management will be done within the database after import.

A MapReduce solution for reading data from files and pushing them to the database would be a more sophisticated and efficient approach. This approach can take advantage of the distributed transactions feature provided by MySQL. This is currently out of scope of this project and can be considered as future work.

6.1.2 Dataset Analysis

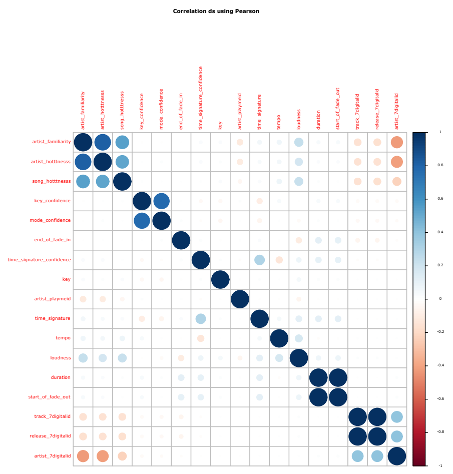

Figure 2: Significantly Correlation Data From Single Value Attributes

Another version of the Python script converts only the single valued attributes to a CSV file. This file was used to start initial analysis on the music database. It was recognized that only a few of the attributes hold highly correlated significance (see figure 2). Targeting these attributes should yield the most significant changes in model tuning. Following suit of our recommendation system, Kmeans clustering was performed on all of the single attributed data. An R script was created to perform Kmeans with a cluster value of k=10 on all numerical attributes of this file. The model gives evenly distributed results with the max size of a cluster as 501 rows and the min as 308. Due to the omition of NA values, only 3742 of the 10,000 rows were accounted for warranting a projected average of 374.2 rows per cluster. Next steps are further analysis on why a subset of data is used with the provided algorithm and initial association analysis on the data.

6.2 Deliverables

- Report
- SQL Scripts (for database operations)
- Python Scripts (for data import and exports)
- R scripts
- Rattle Implementations
- Rattle Models

7. REFERENCES

[1] T. Bertin-Mahieux. Mssongdb. `https://github.com/tbertinmahieux/MSongsDB.git`, 2010.

[2] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011. `http://labrosa.ee.columbia.edu/millionsong/`.

[3] T. H. Group. Hdfview. `http://www.hdfgroup.org/products/java/hdfview/`, 2014.