DATA420-25S2 (C)

Assignment 2

The Million Song Dataset (MSD)

Due on Friday, October 17 by 5:00 PM.

If you want to discuss the assignment material you can use the Discord server where the discussion will benefit all. If you have a question that requires an official answer you can use the forum on LEARN. If you have a more personal question you can email me or contact the class rep as needed.

A reminder that the Discord server is for discussion of concepts only, not for sharing code or answers to assignment questions.

Links

Report upload (pdf)
Supplementary material upload (zip, limited to 10 MB)

Discord server
Help forum for Assignment 2

Instructions

- Any assignments submitted after the deadline without obtaining an extension will receive a 20% penalty. You should aim to work on the assignment incrementally in your own time and in the labs each week.

- The forum is a great place to ask questions about the assignment! Any questions will be answered by a facilitator as soon as possible but students can also answer each other's questions so that you can all benefit from the answers.

- The data provided for the assignment is stored in Azure Blob Storage and outputs that you generate will be stored in Azure Blob Storage as well. Hadoop and Spark can both interact with Azure Blob Storage similar to how they interact with HDFS, but where the replication and distribution is handled by Azure instead. This makes it possible to read or write data in Azure over HTTPS where the path is prefixed by `wasbs://`. Follow the instructions in the notebook provided to load data from the data container and to save outputs that you generate to a separate user container where the path is prefixed by your username.

The Million Song Dataset (MSD)

In this assignment we will study a collection of datasets referred to as the Million Song Dataset (MSD), a project initiated by The Echo Nest and LabROSA. The Echo Nest was a research spin-off from the MIT Media Lab established with the goal of understanding the audio and textual content of recorded music, and was acquired by Spotify after 10 years for 50 million Euro.

- The Million Song Dataset (MSD)

The main dataset contains feature analysis and metadata for one million songs, with song ID, track ID, artist ID, and 51 other fields such as the year, title, artist tags, and various audio properties such as loudness, beat, tempo, and time signature. Note that there is an distinction between the terms song and track. ***Song*** is a general concept representing the musical work, and ***track*** is a specific instance or recording of a song. There may be multiple tracks for one song, differing in how they were recorded or how they sound. The tracks in the dataset have been matched to songs and artists.

The Million Song Dataset also contains other datasets contributed by organisations and the community,

- Taste Profile subset (user-song plays)
- All Music genre datasets (genre labels)
- All Music audio feature datasets (extracted from samples)

We will focus on the All Music and Taste Profile datasets, but you are free to explore the other datasets on your own as you work through the assignment material. There are many online resources and some publications exploring these datasets as well.

Taste Profile

The Taste Profile dataset contains real user-song play counts from undisclosed organisations. All songs have been matched to identifiers in the main million song dataset and can be joined with this dataset to retrieve additional song attributes. This is an implicit feedback dataset as users interact with songs by playing them but do not explicitly indicate a preference for the song.

The dataset has an issue with the matching between the Taste Profile tracks and the million song dataset tracks. Some tracks were matched to the wrong songs, as the user data needed to be matched to song metadata, not track metadata. Approximately 5,000 tracks are matched to the wrong songs and approximately 13,000 matches are not verified. This is described in their blog post in detail.

MSD Allmusic Genre Dataset (MAGD)

Many song annotations have been generated for the MSD by sources such as Last.fm, musiXmatch, and the Million Song Dataset Benchmarks by Schindler et al. These benchmarks contain song level genre and style annotations. We will focus on the MSD Allmusic Genre Dataset (MAGD) provided by the Music Information Retrieval group at the Vienna University of Technology.

This dataset is included on the million song dataset benchmarks downloads page and class frequencies are provided on the MSD Allmusic Genre Dataset (MAGD) details page as well. For more information about the genres themselves have a look at the All Music genres page.

Audio features

The Music Information Retrieval group at the Vienna University of Technology also downloaded audio samples for 994,960 songs in the dataset which were available from an online provider, most in the form of 30 or 60 second snippets. They used these snippets to extract a multitude of features to allow comparison between the songs and prediction of song attributes,

Rhythm Patterns
Statistical Spectrum Descriptors
Rhythm Histograms
Temporal Statistical Spectrum Descriptors
Temporal Rhythm Histograms
Modulation Frequency Variance

Marsyas
Timbral features

jMir
Spectral Centroid
Spectral Rolloff Point
Spectral Flux
Compactness
Spectral Variability
Root Mean Square
Zero Crossings
Fraction of Low Energy Windows
Low-level features derivatives
Method of Moments
Area of Moments
Linear Predictive Coding (LPC)
MFCC features

These features are described in detail on the million song dataset benchmarks downloads page and the audio feature extraction page, and the number of features is listed along with file names and sizes for the separate audio feature sets.

TASKS

The assignment is separated into a number of sections, each of which explore the data in increasing detail from processing to analysis to visualizations. You will need to plan your time carefully to complete the assignment on time..

These supplement the report requirements detailed on the cover page. In general, your write up should be accurate and concise and should demonstrate depth of understanding. The tasks are intentionally detailed to make it easier to work through the assignment step by step.

Processing

The datasets that you need have already been downloaded and are provided in cloud storage under `wasbs://campus-data@madsstorage002.blob.core.windows.net/msd/`. They have been placed into directories as described below.

The `main` directory contains song and track metadata from the main million song dataset but no audio analysis, song lyrics, genre tags, or user-song plays (see the page on getting the dataset as to why they are provided separately). Note that the `analysis.csv.gz` and `metadata.csv.gz` are ordered and can be joined based on row number to associate song and track.

The `genre` directory contains each of the genre datasets provided by the Music Information Retrieval group at the Vienna University of Technology. There is one tab separated file for each genre subset and you can use them interchangeably.

The `audio` directory contains audio features sets from the Music Information Retrieval group at the Vienna University of Technology. The `audio/attributes` directory contains attributes names from the header of the ARFF and the `audio/features` directory contains the audio features themselves.

The `tasteprofile` directory contains the user-song play counts from the Taste Profile dataset as well as logs identifying mismatches and matches that were manually accepted. You don't **need** to use these to do anything with these mismatches.

Q1 First you will explore the relevant datasets that you need at a high level.

Read through the links above and ***then*** use the `hdfs` command to explore the relevant datasets without loading any data into memory.

(a) Give an overview of the structure of the datasets, including their sizes, formats, data types, and how each dataset has been stored.

(b) Figure out how to load each of the different types of datasets, even if you infer schema and only load a small subset of the data for testing.

(c) Count the number of rows in each of the datasets. How do the counts compare to the total number of unique songs?

Q2 Complete the following data preprocessing.

(a) Load the audio feature attribute names and types from the `audio/attributes` directory and think about how they can be used to define schemas for the audio feature datasets themselves.

You could use the `hdfs` command to download the attributes files to your local directory and open them using Python or you could use `spark.read` to load them into distributed memory as a single column of text and collect them into memory on the master node to use locally.

(b) Note that each of the attribute and feature datasets share the same prefix and the attribute types are named consistently. Think about how you can create a `StructType` automatically by mapping attribute types to `pyspark.sql.types` objects.

(c) Do you think these audio feature attribute names are convenient to use as **column** names?

Think about the advantages and disadvantages of using these column names as you move on to develop models in the sections below.

(d) Develop a systematic way to rename columns in the audio feature datasets after you load them in Spark so that you **could** refer to each column in each dataset uniquely **if** you merged them together. Your column names should be short, intuitive, and easy to read.

Do not actually load and merge all of the audio feature datasets together as this would require a significant amount of distributed memory.

Audio similarity

In this section you will explore using audio based features to predict the genre of a track. This could be used to enable an online music streaming service to compare songs based entirely on the way their tracks sound, and to discover rare songs that are similar to popular songs even though they have no collaborative filtering relationship. This would enable users to have more precise control over variety, and to discover songs they would not have found any other way.

Q1 The audio feature datasets each have different levels of detail. If we were developing a real world model we would eventually consider all of them to select the best features for our particular model but for this assignment you will use the datasets specified below to save time.

(a) Load and merge the following audio feature datasets to use in the questions below.

```
msd-jmir-area-of-moments-all-v1.0
msd-jmir-lpc-all-v1.0
msd-jmir-spectral-all-all-v1.0
msd-marsyas-timbral-v1.0
```

The audio features are continuous values, obtained using methods such as digital signal processing and psycho-acoustic modeling. Generate descriptive statistics for each audio feature or column across the datasets after they have been merged.

Are any features distributed in the same way? Are any features strongly correlated?

(b) Load the MSD Allmusic Genre Dataset (MAGD).

Visualize the distribution of genres and comment on how this could impact the performance of a binary or multiclass classification model.

(c) Merge the genres dataset and your audio features dataset so that every track has a label.

Q2 First you will develop a binary classification model.

(a) You will use the following classification algorithms from the `spark.ml` library.

```
LogisticRegression
RandomForestClassifier
GBTClassifier
```

Explain why these algorithms are sensible candidates, taking into account considerations such as explainability, interpretability, predictive accuracy, training speed, hyperparameter tuning, dimensionality, and issues with scaling.

Based on the descriptive statistics from Q1 step (a), think about what processing you should apply to the audio features before using them to train each model.

(b) Convert the genre column into a new binary label that represents if the track is “Electronic” or some other genre.

What is the class balance of the binary label?

(c) Split the dataset into training and test sets using an 80 / 20 split. You should use stratified random sampling to ensure class balanced is preserved.

You may also want to use a resampling technique such as subsampling, oversampling, or observation weighting. Justify your choice of resampling method.

(d) Train each of the three classification algorithms specified.

(e) Use the test set to compute relevant performance metrics for each classification algorithm, including at least accuracy, precision, and recall.

(f) Discuss the relative performance of the classification algorithms based on the performance metrics that you computed in step (e).

How does the class balance of the binary label affect these performance metrics?

What is the best metric to compare the relative performance of the classification algorithms?

Q3 Next you will extend your work above to predict across all genres instead.

(a) You will use `LogisticRegression`. Does this classifier support multiclass classification?

Explain how it can be used to predict multiple classes and if you need to do any additional configuration or use any additional classes. Are there any additional assumptions that need to be satisfied?

(b) Convert the genre column into an integer label that encodes genre consistently.

How has your effective class balance changed going from binary to multiclass classification?

(c) Split your dataset into training and test sets, train the classification algorithm that you chose in step (a), and compute some relevant performance metrics that are suitable for multiclass classification. Make sure you take into account the class balance in your comments.

How has the performance of your algorithm been impacted by including multiple genres?

Q4 Now you will think about hyperparameter tuning.

(a) Look up the hyperparameters for each of the three classification algorithms.Think about how these hyperparameters affect the performance of each model and discuss if you would have chosen different hyperparameter values for Q2 part (d) or Q3 part (c).

(b) Explain how cross-validation works and why it is used for hyperparameter tuning.

(c) Explain how you ***would*** tune the hyperparameters of a classification algorithm in general, including how you would define the hyperparameter grid. **You do not need to write code or run it in Spark.**

How much do you expect hyperparameter tuning **could** improve your performance metrics?

Song recommendations

In this section you will use the Taste Profile dataset to develop a song recommendation system based on collaborative filtering.

Q1 First it will be helpful to know more about the properties of the dataset before you begin training the collaborative filtering model.

(a) Based on the size, format, and the number of rows, do you think you should repartition and cache the dataset to improve your efficiency when training collaborative filtering algorithms?

(b) Load the dataset and answer the questions below.

How many unique songs are there in the dataset? How many unique users?

How many unique songs has the most active user played?

What is this as a percentage of the total number of unique songs?

(c) Generate any other descriptive statistics that you think might be relevant to understand how the data is distributed.

(d) Visualize the distribution of song popularity and the distribution of user activity by collecting the counts of unique user plays per song and the counts of unique song plays per user respectively, and describe the shape of the distributions.

Q2 Next you will train the collaborative filtering model.

(a) Collaborative filtering determines similar users and songs based on their combined play history. Songs which have been played only a few times and users who have only listened to a few songs will not contribute much information and are unlikely to improve the model.

Create a smaller dataset of user-song plays by removing any songs which have been played fewer than N = 20 times and users who have played fewer than M = 20 songs in total.

How many unique users and songs remain in the dataset? How many have been excluded? Do you think the thresholds above are sensible?

(b) Convert the user and song identifiers into integer identifier in a consistent way.

(c) Split the dataset into training and test sets using an 80 / 20 split. Note that due to the nature of the collaborative filtering model, you must ensure that every user in the test set has some user-song plays in the training set as well. Explain why this is required and how you have done this while keeping the selection as random as possible.

(d) Use the `spark.ml` library to train an implicit matrix factorization model using Alternating Least Squares (ALS).

(e) Select a few of the users from the test set by hand, generate some recommendations using the model, and compare these recommendations to the songs the user has actually played.

Comment on the effectiveness of the collaborative filtering model.

(f) Use the test set of user-song plays and recommendations from the collaborative filtering model to compute the following metrics for K = 10.

- Precision @ K
- NDCG
- Mean Average Precision (MAP)

Look up these metrics and explain why they are useful in evaluating the collaborate filtering model. Explore the limitations of these metrics in evaluating a collaborative filtering model or recommendation system in general.

(g) How would you test the performance of a recommendation system that you have developed in a the real world where you have real users that interact with your system? You should explain what strategy you would use and what performance metrics you could evaluate.