DATA420-A2

david.ewing.nz

September 2025

Assignment Method Overview

1. **Data Exploration**

   (a) Overview of dataset structure, sizes, formats, data types, storage.
   Use HDFS or Spark commands to list files, check file sizes, and inspect file headers. Summarise formats (CSV, TSV, ARFF), data types (numeric, string), and storage locations.

   (b) Loading different dataset types.
   Use Spark's read methods to load small samples, infer schema, and validate by displaying the first few rows.

   (c) Counting rows and comparing to unique songs.
   Use Spark's count() and distinct().count() to compare row counts and unique song counts.

2. **Data Preprocessing**

   (a) Loading audio feature attribute names and types.
   Read attribute files as text, parse to extract names/types, and use to define Spark schemas.

   (b) Mapping attribute types to Spark types.
   Create a mapping (e.g., "NUMERIC" → FloatType), build StructType schema programmatically.

   (c) Evaluating attribute names as column names.
   Review names for clarity, length, and uniqueness; discuss pros/cons for modelling and merging.

   (d) Systematic renaming for merging.
   Propose a naming convention (e.g., prefix with dataset name), use Spark renaming functions.

3. **Audio Similarity**
   Classification

   (a) Merging audio feature datasets and generating statistics.
   Load datasets, join on track/song ID, compute descriptive statistics and correlations.

(b) Loading and analysing genre dataset.
Load genre data, count frequencies, visualise, and discuss class imbalance.

(c) Merging genres and features.
Join on track/song ID to create a labelled dataset for classification.

4. **Binary Classification**

(a) Algorithm selection and justification.
Discuss why logistic regression, random forest, and GBT are suitable; propose preprocessing.

(b) Creating a binary label for "Electronic".
Map genre column to binary, calculate and report class balance.

(c) Splitting data and handling class balance.
Use stratified sampling for 80/20 split; justify resampling.

(d) Training classifiers.
Fit each model using Spark ML, document training process.

(e) Evaluating performance.
Use test set to compute accuracy, precision, recall; present results in a table.

(f) Comparing algorithms and metrics.
Discuss which model performs best and why; explain metric choices.

5. **Multiclass Classification**

(a) Logistic regression for multiclass.
Explain one-vs-rest or multinomial logistic regression, note extra configuration if needed.

(b) Encoding genres as integer labels.
Use StringIndexer or similar, recalculate class balance.

(c) Training and evaluating multiclass model.
Train on multiclass labels, compute multiclass metrics, discuss impact of class imbalance.

6. **Hyperparameter Tuning**

(a) Hyperparameters for each algorithm.
List key hyperparameters, discuss expected effects and alternatives.

(b) Cross-validation explanation.
Describe k-fold cross-validation and its role in tuning.

(c) Tuning process and expected impact.
Outline grid/random search, estimate potential improvement in metrics.

7. **Song Recommendation (Collaborative Filtering)**

(a) Dataset partitioning and caching.
Assess dataset size, recommend repartitioning/caching for efficiency.

(b) Loading and analysing user-song data.
Count unique users and songs, find most active user, calculate percentages.

(c) Additional descriptive statistics.
Propose further stats (e.g., average plays per user/song).

(d) Visualising distributions.
Plot song popularity and user activity distributions, describe observed patterns.