Automated Genre Classification on the Free Music Archive

Erik Duus
MATH637 2021

Background

The preceding project introduced the FMA dataset and examined extracted audio features to determine if they conveyed information about the genre structure of the dataset.

Building on that effort, it seems natural to determine whether machine learning techniques can be applied

\to build an automatic genre classifier using the available audio features in the FMA dataset. Several different approaches are investigated, and some enhancements are applied to the dataset in the hopes of improving performance.

Classic Machine Learning Classifiers

The genre exploration focused on the FMA 'small' dataset, which comprises 8,000 tracks with a single root genre, evenly balanced across 8 roots. The ensuing investigations use the 'medium' dataset to provide more exemplars and a more realistic corpus. It contains approximately 25,000 tracks with a single root genre but unbalanced across roots. The root genre breakdown is as follows:

Root Genre	**# Tracks**	**Root Genre**	**# Tracks**	**Root Genre**	**# Tracks**
Rock	7103	Pop	1186	Soul-RnB	154
Electronic	6314	International	1018	Spoken	118
Experimental	2251	Classical	619	Blues	74
Hip-Hop	2201	Old-Time / Historic	510	Easy Listening	21
Folk	1519	Jazz	384		
Instrumental	1350	Country	178		

The bottom 3 genres are pruned from the dataset due to the low number of exemplars. The imbalance between genres is noteworthy, as is the support of each genre. For example, Folk is more prevalent than Pop or Country, which is not reflective of these genres' respective popularity.

SVM, Logistic Regression and KNN Classifiers

4 classic machine learning classifiers from the scikit-learn library are built against all features of the medium dataset with default settings. Training and test datasets are built using an 80/20 stratified split. For simplicity, cross-fold validation is not used. The resulting accuracy at first glance seems quite good:

	SVM-Linear	**SVM-RBF**	**Logistic**	**KNN**
Accuracy	0.68	0.66	0.64	0.59
F1-macro	0.52	0.44	0.33	0.46
F1-weighted	0.66	0.62	0.61	0.57

However, the macro-weighted F1 scores show that the genre imbalance probably skews the results. Examining the macro-weighted F1 scores for the 3 best and 3 worst classes:

	SVM-Linear	SVM-RBF	Logistic	KNN	support
Historic	0.95	0.97	0.91	0.95	102
Rock	0.81	0.78	0.77	0.74	1421
Classical	0.79	0.77	0.72	0.68	124
Country	0.15	0.00	0.00	0.22	35
Pop	0.12	0.00	0.06	0.15	237
Soul-RnB	0.06	0.00	0.00	0.12	31

The better performance of the well-supported Rock and Electronic genres overweights the overall accuracy and masks the poor performance on Country, Pop, and Soul.

Class-weighting the Classifiers

SVM and Logistic Regression provide the ability to weight classes according to their representation, therefore paying more attention to smaller genre classes. Overall accuracy decreases slightly, but macro-weighted F1-score improves modestly, indicating better improvement in the under-represented classes:

	SVM-Linear	SVM-RBF	Logistic
Accuracy	0.61	0.58	0.55
F1-macro	0.54	0.50	0.48
F1-weighted	0.63	0.60	0.58

The macro-weighted F1 scores show a slight decrease for the 3 best-performing classes, but a modest increase for the worst performers :

	SVM-Linear	SVM-RBF	Logistic	support
Historic	0.94	0.94	0.92	102
Rock	0.78	0.75	0.75	1421
Classical	0.74	0.73	0.70	124
Country	0.26	0.21	0.20	35
Pop	0.24	0.21	0.22	237
Soul-RnB	0.26	0.18	0.13	31

Hyper-parameter Tuning the Classifiers

A round of elementary hyper-parameter tuning is performed on the 4 class-weight-balanced classifiers using a RandomizedGridSearch. SVM-Linear and KNN do not improve, Logistic Regression improves modestly, and SVM-RBF improves substantially:

	SVM-Linear	SVM-RBF	Logistic	KNN
Accuracy	0.61	0.69	0.59	0.59
F1-macro	0.53	0.63	0.51	0.47
F1-weighted	0.63	0.69	0.61	0.57

Examination of the macro-weighted F1 scores shows improvement in the 3 worst genres with about the same performance in the 3 best genres.

	SVM-Linear	SVM-RBF	Logistic	KNN
Historic	0.94	0.99	0.94	0.93
Rock	0.78	0.82	0.78	0.73
Classical	0.72	0.83	0.75	0.68
Country	0.27	0.55	0.23	0.23
Pop	0.23	0.29	0.26	0.10
Soul-RnB	0.24	0.40	0.21	0.21

Results

SVM-RBF shows promise as a genre classifier. Support vector machines scale well to higher dimensions, and the RBF handles non-linear separating surfaces. Neither KNN nor LogisticRegression performed well. KNN may suffer from both the curse of dimensionality and from the class imbalance; a majority class can invade the neighbor space of a minority class. LogisticRegression is a linear model and prone to overfitting in high dimensions.

Ensembles - RandomForest and XGBoost

Given the difficulty of creating a separating surface between some pairs of genres and the class imbalance of the dataset, it seems natural to turn to ensemble classifiers. They can produce complex, nonlinear separating boundaries. They can also focus on difficult or minority classes (XGBoost, for example). Classifiers are built and tested using scikit-learn's RandomForest and XGBoost.

As with the previous classifiers, a RandomForest is fitted against the dataset with default parameters. To address the dataset imbalance and attempt to improve performance, a class-weighted RandomForest is also fitted, and then some simple hyperparameter tuning is attempted. Class-weighting and tuning have little impact, and in general, the RandomForest has mediocre performance compared to SVM.

	RF	RF-balanced	RF-tuned
Accuracy	0.64	0.63	0.63

F1-macro	0.45	0.46	0.45
F1-weighted	0.60	0.59	0.59

Another classifier is constructed in the same fashion using XGBoost:

	XBG	**XGB-balanced**	**XGB-tuned**
Accuracy	0.69	0.69	0.71
F1-macro	0.55	0.55	0.58
F1-weighted	0.67	0.67	0.69

Tuned XGBoost shows performance close to SVM-RBF. Examining the best and worst genres:

	XGB	**XGB-tuned**
Historic	0.96	0.97
Rock	0.81	0.82
Classical	0.83	0.85
Country	0..20	0.27
Pop	0.18	0.23
Soul-RnB	0.12	0.12

Tuned XGBoost shows modest improvements across the best and worst-performing genres.

Results

RandomForest significantly underperformed XGBoost. This may simply be due to their construction and the imbalanced training data; RandomForests grows decision trees randomly, with no attention paid to minority classes. XGBoost by contrast focuses on misclassifications with each round of training, so even with an imbalanced training set, hard-to-classify classes are given extra focus. The performance of both could perhaps be improved by using data augmentation techniques to balance the training data.

Neural Networks via MLPClassifier

Given the complexity of the classification problem, a neural network might prove to be successful. Since the audio features are summary statistics, a simple fully-connected network is perhaps the appropriate choice over more complex architectures like CNNs.

A genre classifier is fitted with scikit-learn's MLPClassifier using default settings against the dataset. As in the preceding analysis, a second classifier is fitted that attempts to address the class imbalance. MLPClassifier does not support class_weighting, as do SVC and LogisticRegression. To provide a more balanced training dataset, simple oversamping and undersampling are performed.

For each genre with support < 500 tracks, the training set is randomly oversampled to bring the genre support to 500. Additionally, for each genre with support > 3500 tracks, the training set is randomly undersampled to bring the genre support down to 3500. The training set genre breakdown is as follows:

Root Genre	# Tracks	Root Genre	# Tracks	Root Genre	# Tracks
Electronic	3,500	Instrumental	1,080	Jazz	500
Rock	3,500	Pop	949	Old-Time / Historic	500
Experimental	1,801	International	814	Soul-RnB	500
Hip-Hop	1,761	Classical	500		
Folk	1,215	Country	500		

The test set is left unchanged, and the MLPClassifier again trained with default settings.

Finally, a simple round of hyper-parameter tuning is performed to fit the third classifier. The summary results for the 3 classifiers are:

	MLP	MLP balanced	MLP tuned
Accuracy	0.69	0.65	0.68
F1-macro	0.54	0.56	0.59
F1-weighted	0.67	0.65	0.68

Modest performance increases are noted for both balancing and tuning. The final classifier performs roughly as well as SVC-RBF and XGBoost. Examining the performance of the top and bottom genres also shows similar results:

	MLP	MLP balanced	MLP tuned
Historic	0.94	0.96	0.98
Rock	0.81	0.80	0.82
Classical	0.78	0.78	0.77
Country	0.23	0.29	0.46
Pop	0.17	0.20	0.27
Soul-RnB	0.17	0.27	0.31

Results

MLPClassifier performs quite well after data augmentation and tuning, rivaling SVM-RBF and XGBoost. This is expected since neural networks can fit arbitrarily complex functions. The unbalanced training data again causes certain genres to perform poorly. Additionally, the Pop genre continues to be hard to classify, even though it is moderately well-represented in the training data.

Multi-Genre Classification using MLPClassifier

The preceding analysis was restricted to tracks with genres that share a single root genre, thereby eliminating approximately 75% of the dataset. While this choice simplified the classification problem, it may be suboptimal for several reasons:

- The single root genre subset is not representative of the full corpus.
- Important exemplars may be omitted.

- Genre combinations may themselves provide a structure that can be used by a classifier.

Therefore it may be more appropriate to reframe genre classification as a multi-label problem. Music genres might be viewed as an 'influence' or a 'flavor' as opposed to a single identity. Intuitively this seems natural; genre combinations like Country-Rock and Electronic-Pop are real and popular.

Given the size of the FMA genre hierarchy and the inconsistency in the number of genres tagged per track, some crude processing is performed to streamline the multi-genre problem. The genre list for each track is rolled up into a root genre list. In addition, the number of contributions to each root genre is noted, and the 2 most-contributed root genres are used as a multi-genre pair for the classification analysis.

MLPClassifier from scikit-learn is used to build the multi-label classifier. It naturally supports multi-label classification since it generates probabilities for each class, so a multi-label output is produced by selecting those classes above a probability threshold.

Tracks with <= 3 root genres from the full dataset are selected, providing ~87,000 tracks for analysis. The genre breakdown is still extremely imbalanced, but quite different from the single root genre dataset.

genre	# tracks	genre	# tracks	genre	# tracks
Experimental	28,001	Folk	8,272	Country	1,191
Electronic	26,084	Hip-Hop	6,810	Soul-RnB	926
Rock	25,554	International	3,623	Old-Time / Historic	792
Instrumental	9,079	Classical	2,695	Blues	689
Pop	9,076	Jazz	2,359		

To prepare the data for training, the root genre list is binarized, which is similar to one-hot encoding but allowing multiple classes to be represented per instance. An MLPClassifier with default settings is trained on an 80/20 split of the dataset, with no attempt at stratification:

accuracy	hamming loss	hamming loss - zeros
0.3208	0.0757	0.1025

Accuracy measures whether the predicted labels match the true labels exactly; this is a stringent measure that gives no credit for a partial match. Hamming Loss is often used to score multi-label classification, but here it is not a good fit. It measures the 'distance' from the predicted labels to the true labels. Since genre classification is dealing with 14 classes, with only 1 or 2 class predictions, the predicted and true labels are sparse. As seen in the table above, Hamming Loss on a prediction of all zeros scores similarly to the trained classifier's predictions.

The F1 scores from scikit-learn classification report seem more appropriate. The scores are computed for each label, and several different averages are produced:

	precision	recall	f1-score
micro avg	0.72	0.42	0.53
macro avg	0.63	0.28	0.35

weighted avg	0.70	0.42	0.50
samples avg	0.55	0.47	0.49

Precision scores are much higher than recall. This could be due to the 50% probability threshold used by MLPClassifier, leading the classifier to 'underpredict'. While MLPClassifier does not support changing the threshold for training, the predictions of the test set can be modified by using a lower threshold (in this case 30%):

	precision	**recall**	**f1-score**
micro avg	0.57	0.61	0.59
macro avg	0.53	0.41	0.43
weighted avg	0.56	0.61	0.57
samples avg	0.61	0.65	0.60

Lowering the probability threshold increases the macro-weighted F1-score at the expense of some precision. Examining the genre-specific F1 scores reveals the better-than-usual performance of the Pop genre. Notably, the Pop genre has much better support in the multi-label dataset.

	Multi-label	**Support**
Historic	0.84	161
Rock	0.69	5113
Classical	0.55	537
Country	0.07	231
Pop	0.30	1819
Soul-RnB	0.00	199

Balancing Training Data by Oversampling

Oversampling and undersampling are typically applied to unbalanced training data. For a multilabel dataset, however, these techniques are not currently well-standardized. To simply test whether oversampling produces beneficial results, a crude oversampling is performed where all tracks are duplicated that are not tagged with Experimental, Electronic, or Rock. Modest improvement is noted:

	precision	**recall**	**f1-score**
micro avg	0.73	0.43	0.54
macro avg	0.64	0.31	0.37
weighted avg	0.70	0.43	0.51
samples avg	0.56	0.48	0.50

Reclassifying the predictions with a 35% threshold, we again see modest improvement.

	precision	**recall**	**f1-score**

micro avg	0.56	0.64	0.60
macro avg	0.48	0.49	0.47
weighted avg	0.56	0.64	0.59
samples avg	0.60	0.68	0.61

The associated genre-specific F1 scores show modest improvement:

	Multi-label	**Support**
Historic	0.84	161
Rock	0.69	5113
Classical	0.58	537
Country	0.24	231
Pop	0.34	1819
Soul-RnB	0.09	199

Clearly, data augmentation techniques to balance the training data seem to have a beneficial impact on the classifier and warrant further investigation.

Hyper-Parameter Tuning the Classifier

Using a larger network of 400 hidden nodes and a larger regularization parameter, a configuration which showed some success in the previous analysis, modest improvement is again noted:

	precision	**recall**	**f1-score**
micro avg	0.69	0.49	0.57
macro avg	0.6	0.37	0.44
weighted avg	0.67	0.49	0.56
samples avg	0.59	0.54	0.54

Reclassifying predictions with a 35% threshold, we again see modest improvement, but now performance is no better than the smaller network:

	precision	**recall**	**f1-score**
micro avg	0.59	0.60	0.60
macro avg	0.51	0.45	0.47
weighted avg	0.59	0.60	0.59
samples avg	0.61	0.64	0.60

Given that the training data is still quite unbalanced, it would be premature to draw conclusions about the performance of the larger network vs. the smaller. Further research on balancing the training data would probably be the most important next step, followed by optimizing the network sizing.

Summary

A subset of the FMA dataset was used to train 4 classic ML classifiers for genre identification. They all indicated some ability to discern genres, with SVM performing the best. Ensemble approaches were also tried, with XGBoost performing similarly to SVM. In all cases, some genres were resistant to classification, most particularly the Pop genre.

A neural network classifier was also fitted using MLPClassifier, and random over and undersampling was used to better balance the training data, eventually achieving performance comparable to the SVM-RBF and XGBoost classifiers.

Finally, genre classification was reframed as a multi-label problem. A larger subset of FMA was used to fit another MLPClassifier, followed by some attempts at optimization via oversampling and tuning. The approach shows some promise and requires a particular focus on the augmentation of unbalanced multi-label training data.

Future areas of investigation could include:

- Expanding the FMA dataset to increase the representation of genres.
- Adopting a consistent and well-accepted genre hierarchy.
- Ensuring the accuracy of FMA genre labeling.
- Using the full extracted spectral features to retain the temporal structure.
- Investigating other audio features such as lyric content and rhythm features.