Music Genre Recognition from Audio Tracks and Metadata

Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel
Bilkent University
Departmant of EEE & CS
{berkay.erkan,can.kocagil,mehmet.caliskan,eren.ceyani,hammad.musakhel}@ug.bilkent.edu.tr

Abstract

Automatic recognition of audio tracks from music waves has been a challenging task in Music Information Retrieval (MIR). The mathematical representation of natural music, composed of rhythm, tones, intervals, patterns, and harmonies, is a complex subject for human specialists as there is a geometric interpretation in the humming of the strings and inherent subjective nature. The discrimination of genres from purely audio tracks is stochastic by nature and deeply controversial as the genres share common statistical knowledge. For instance, the musical composition emerges from the intersection of multi-cultural genres, known as fusion genres, e.g., country rock is a fusion of country music and rock music. Contextually, the difficulty level of autonomous discrimination of genres is changing over time. Music trends are changing, making the music genre classification task temporally dynamic and introducing a higher degree of conceptual complexity. For the scope of the project's progress, we designed a computational system of end-to-end learning mechanisms based on classical machine and ensemble learning algorithms for automatic recognition of music genres from Creative Commons licensed audio files, with hand-crafted track-level features.

Keywords- Audio Processing, Music Information Retrieval, Music Genre Recognition, Multi-class Time Series Classification

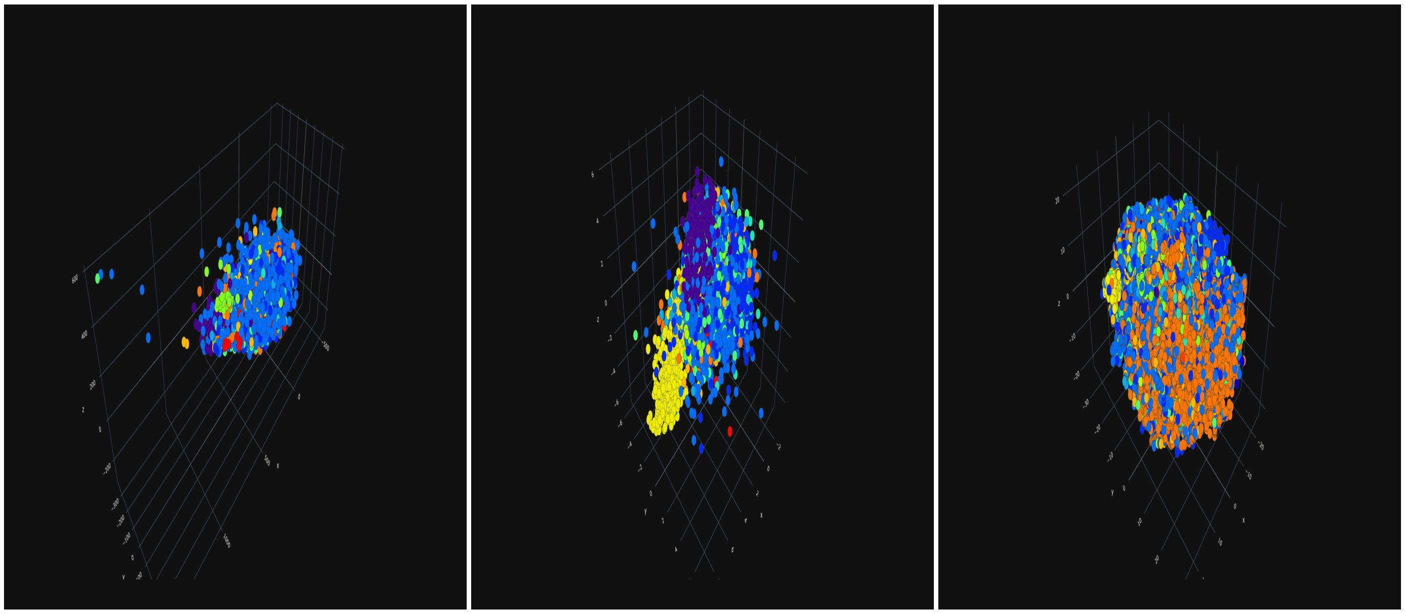


Figure 1: Left to right: Three dimensional Principal Component Analysis, Linear Discriminant Analysis, t-distributed Stochastic Neighbor Embedding (t-SNE) on metadata

1. Introduction

As the dynamics of common genres are temporally varied, representable and scalable datasets are required to perform cognitive tasks in Music Information Retrieval (MIR). The Free Music Archive (FMA) [1] is a dataset for everyday MIR analysis tasks and is supervised for browsing, searching, and organizing extensive music collections. The FMA dataset is a MIR benchmark dataset that consists of 106,574 tracks from 16,341 artists and 14,854 albums, arranged in a hierarchical taxonomy of 161 genres [1]. Its large composition is structured by 30 seconds of high-quality audio, pre-computed features, together with track- and user-level metadata, tags, and free-form text such as biographies, for scaling other music domain processing tasks [1]. FMA is a metadata-rich, future-proof, and permissively licensed dataset that offers quality audio, pre-computed music-level features, reproducibility, and long-term sustainability. Considering the computational comfort zone, the FMA dataset provides four versions, divided according to the sizes: small (30 seconds long, 8,000 tracks, eight balanced genres), medium (30 seconds long, 25,000 tracks, 16 unbalanced genres), large (30 seconds long, 106,574 tracks, 161 unbalanced genres) and full (106,574 untrimmed tracks, 161 unbalanced genres). In light of our computational resources being restricted to open-source computing, we will utilize the small or medium versions of the datasets. However, in the case of any computing device availability, we will try to process a full FMA dataset of 879 GiB of raw information.

We implemented trend-aware classifier-based learning algorithms, specialized in capturing the complex interplay of cultures to recognize genres autonomously. Our classes will be music genres supervised and annotated by the artists, from contemporary to jazz. By acknowledging the socio-cultural context, we designed cognitive machine learning algorithms to capture the temporal dynamics of audio, from conventional ML algorithms like SVM to more advanced meta-learning algorithms. Various ML-based techniques are utilized to accomplish the project's scope; to start with, naive and straightforward algorithms such as Decision Trees and SVM are introduced to create our baseline model to benchmark. Then more advanced ensemble classification algorithms such as Random Forests and custom ones are proposed to compare our multi-class results. Initially, we will compute a small-sized dataset that is class-balanced; as such, our primary goal will be to predict the single top genre on the balanced small subset. Then, in the case of large or full-size datasets, our main objective will be to maximize multiple top genres' accuracy. Our task is a multi-class classification of signals and metadata that requires supervised performance evaluation metrics from the computational perspective. In small or medium dataset sizes, single-top class accuracy is the primary metric to compare. Hence, our primary milestone was preparing the data pipeline and creating conventional ML-based classifiers to construct a baseline.

Musical genre recognition classifies the musical collections based on the audio-level, user-level, and track-level features. There are expected challenges to overcome, such as poor labeling and non-triviality extraction of features. In the former one, musical genres are loosely defined, as people often argue over the genre of a song. In our case, the genres are supervised by the artists themselves, so it will be smoother to perform data-centric computational tasks. In the latter one, automatic extraction of semantic features is not differentiable and conceptualized, so there is no standard way to extract features in music information retrieval. The Mel Spectrogram-based representation algorithms are proposed to overcome these challenges by converting audio signals to visuals and representing how the frequency spectrum varies over time [4]. The latter challenge will be tried to be overcome with spectrum-based representations before feeding the ML model. From a more musical perspective, the notations of composers and audios created by their artists consist of musical representations: counting, rhythm, scales, intervals, patterns, symbols, harmonies, time signatures, overtones, tone, and pitch [6], which are highly unstructured and complex to interpret mathematically. Hence, ad-hope statistical and structural pattern recognition frameworks will be fit to capture music's conceptual and contextual sides for the extraction of genres.

Our main study for the scope of the project progresses as follows.

- Metadata-level descriptive data analysis is applied.
- Then, manifold learning and dimensionality reduction methods are performed on the metadata-level music data.
- We built an end-to-end discovery machine learning pipelines to classify the music genres based on the metadata.
- Model explainability operations are applied to capture which features have greater importance.

2. Metadata-level Data Analysis

To gain insight into the FMA dataset, we performed a brief metadata-level data analysis. We analyzed how different audio features vary by genres. Those audio features include acousticness, danceability, speechiness and tempo, which are already given in the dataset. We also analyzed listening counts of each genre. The following figures are self-explained and best understood from the figures.

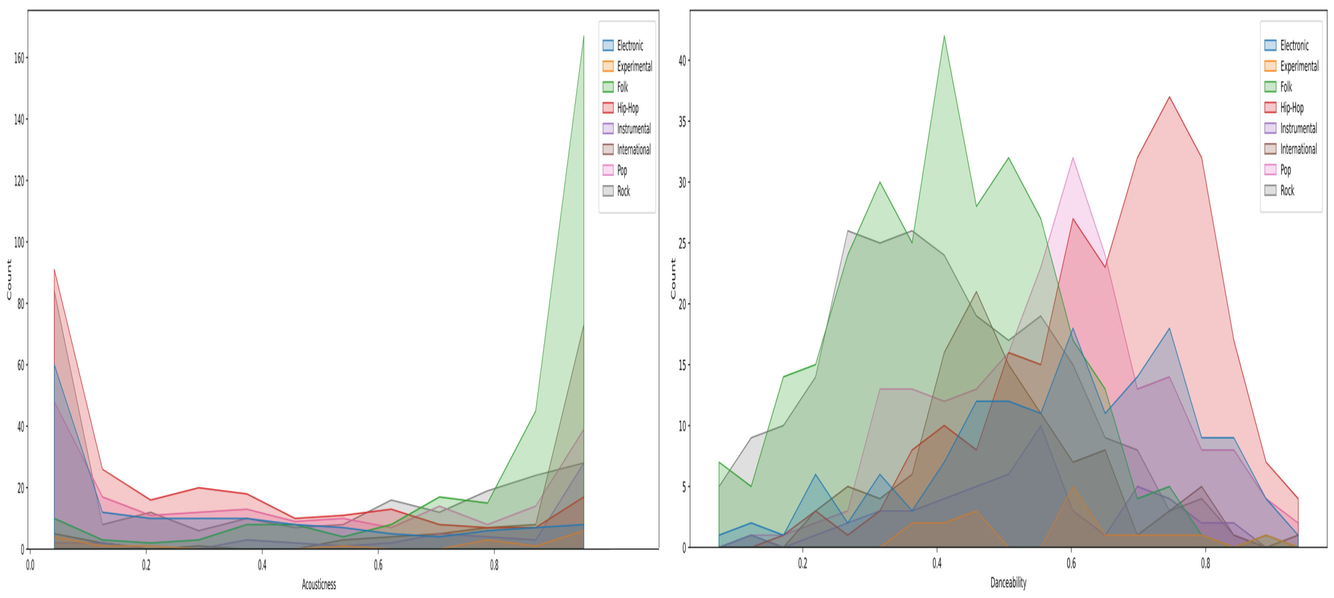


Figure 2: Left to right: Analysis of how acousticness changes for different genres, Analysis of how danceability changes for different genres

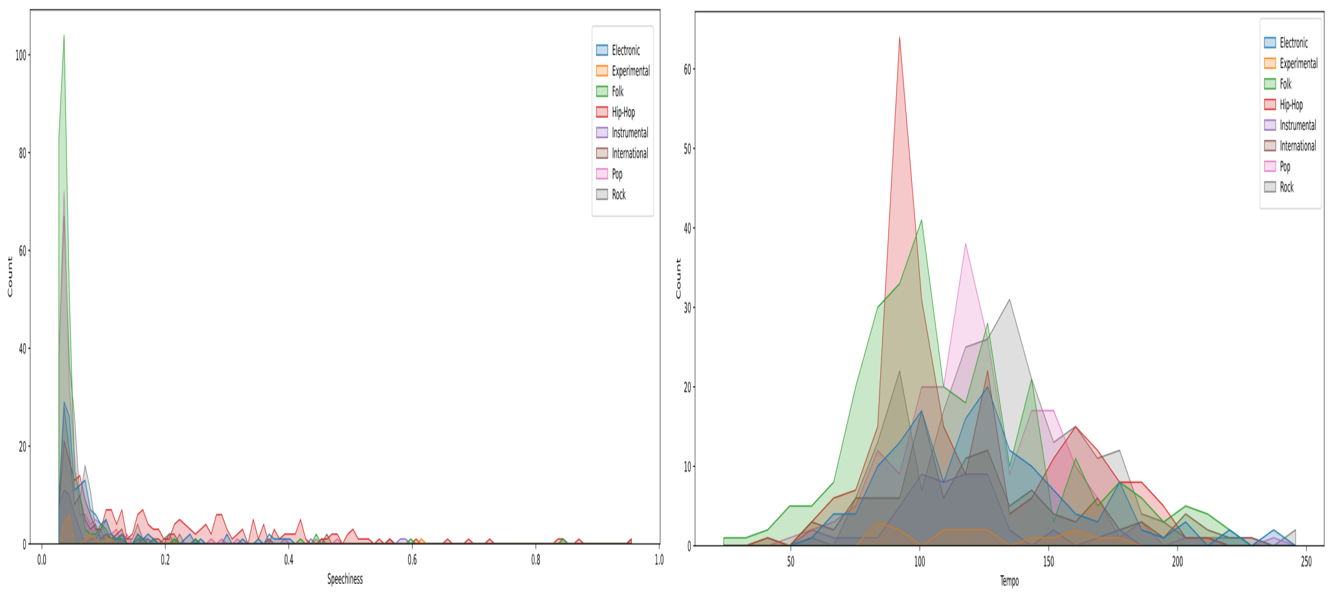


Figure 3: Left to right: Analysis of how speechiness changes for different genres, Analysis of how tempo changes for different genres

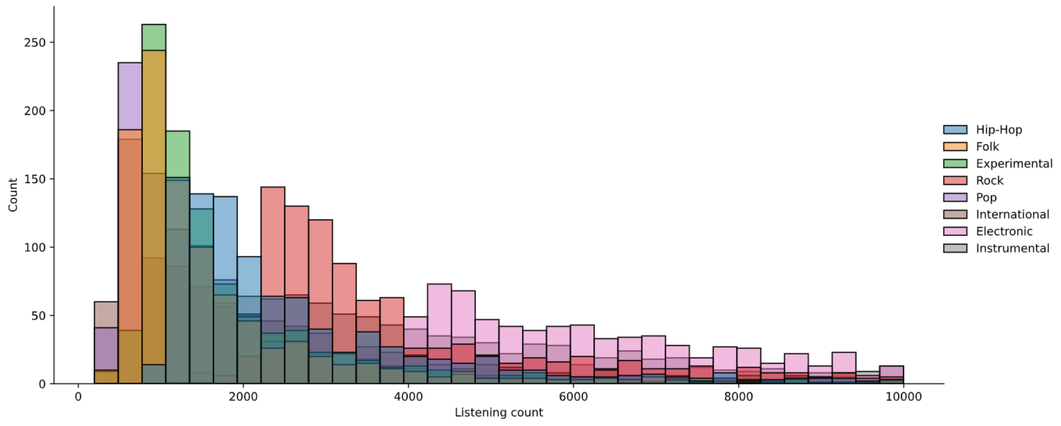


Figure 4: Analysis of how listening count changes for different genres

3. Dimension Reduction to Manifold Learning

We performed PCA and LDA as dimension reduction algorithms. Besides, t-SNE is performed to generate lower-dimensional manifolds of the metadata space. Further, we visualized all manifolds in 3-D dimensional space in figure 1.

3.1. Dimension Reduction: PCA

PCA is a linear unsupervised dimension reduction algorithm, and it computes principal vectors to change the basis of the representation [5]. PCA is a used algorithm in a broad range of topics from image compression to decorrelation of texts. Here, we performed PCA on metadata space and visualized it in figure 1 at the left corner.

3.2. Dimension Reduction: LDA

LDA is a supervised dimensionality reduction algorithm, and it is a generalization of Fisher's linear discriminant, which aims to find linear subspace that characterizes the original data space. Since it is supervised, it is a powerful paradigm in representation learning. Here, we performed LDA on metadata space and visualized in figure 1 at medium. From the figures, we can see that LDA outperforms other methods by uniquely separating geodesic distances in the manifolds.

3.3. Manifold Learning: t-SNE

T-SNE is iterative statistical approach for producing non-linear embedding of the original data space by preserving small pairwise distances or localized similarities. It minimizes the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. Here, we performed t-SNE on metadata space, and visualized in figure 3 at right corner.

4. Machine and Meta Learning

We applied more than ten genre recognition algorithms; we listed our methods and their brief descriptions.

4.1. SVC with Gaussian and Linear Kernel

Support Vector with the linear kernel is an efficient algorithm for linearly separable data spaces by fitting a hyper-plane that categorizes the metadata in our context [2]. We also implemented SVM with radial basis kernel and accumulated to mean of 60% accuracy with a standard deviation of 0.02 by 5-fold cross-validation.

4.2. SGD Classifier

SGD classifier is a linear classifier that uses stochastic gradient descent with Hinge loss to separate data spaces. The SGD classifier is implemented with l_1 regularization [2].

4.3. MLP

MLP is a simple neural network architecture that consists of a bunch of linear layers with ReLU non-linearity and is optimized by the Stochastic Gradient Descent rule [2, 3]. Multi-layer Perceptron Classifier accumulated a mean of 56% accuracy with a standard deviation of 0.02 by 5-fold cross-validation.

4.4. Logistic Regression

Logistic regression is a classical machine learning algorithm; it applies linear transformation on the data followed by sigmoidal activation to generate real-valued probabilities [2].

Given the input feature $\{X\}_{i=1}^{n}$, weight vector w, and the bias term c, the positive class probability can be computed as follows.

$$p(X_i) = \frac{1}{1 + e^{-(c+X_i^T w)}} \, \mathit{for} \; i \in \{\mathit{1,..., n}\} \tag{1}$$

As an optimization problem, binary class ℓ_2 penalized logistic regression minimizes the following cost function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log\big(\exp\big(-y_i(X_i^T w + c)\big) + 1\big). \tag{2}$$

Ridge classifier is an L2 penalized version of logistic regression that helps robust decoding with lower degrees of freedom [2].

4.5. Random Forest Classifier

Random Forest classifier is a meta estimator algorithm that fits parallel decision tree classifiers with bootstrapped samples of the original dataset to improve the predictive accuracy and model reliability [2].

4.6. Gradient Boosting Classifier

Gradient boosting classifier is boosting algorithm that ensembles weak learnings, generally decision trees. It constructs an additive model in a forward stage-wise fashion that enables the model to optimize the parameters on any differentiable loss functions. We utilized 100 weak learners to construct the end classifier [2].

4.7. Quadratic Discriminant Classifier

The quadratic Discriminant classifier is a class conditional density algorithm with quadratic decision boundaries to fit separate Gaussian to each class. It is a powerful method when we have priory knowledge that individual classes exhibit distinct covariance [2].

4.8. AdaBoost Classifier

AdaBoost classifier is a type of ensemble learning algorithm that fits weak classifier on the dataset then fits additional copies of the classifier with adjusted parameters based on the incorrectly classified objects [2]. The basic concept behind Adaboost is to set the weights of classifiers and train the data sample in each iteration such that it ensures the accurate predictions of unusual observations [2].

4.9. Extra Trees Classifier

Extra Trees classifier is another ensemble learning method based on randomized decision trees on the sub-sampled version of the datasets to improve predictive quality [2]. Extra Trees Classifier is an ensemble learning algorithm similar to Random Forest Classifier except for a few key differences. When Random Forest Classifier splits a node, it first finds the optimum split among all splits and chooses that optimum split as the cut point. Extra Trees Classifier chooses this cut point randomly; hence it runs faster than Random Forest Classifier, adding more randomization.

4.10. K-Neighbors Classifier

K-Nearest Neighbor is a simple distance-based supervised ML algorithm. K-Nearest Neighbor works by finding the distances between a query and all the examples in the data, selecting the specified number examples, say K, closest to the query, then votes for the most frequent label in the case of classification or averages the labels in the case of regression [2]. As our task is classification, we take the most frequent label in K nearest neighbors.

We utilized three cartesian space distance metrics as a distance metric: Euclidean, Manhattan, and Cosine. Euclidean distance is the most common one, as it computed the second-order distance between two points.

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_i - y_i)^2 + \cdots + (x_n - y_n)^2}. \tag{3}$$

In the generic form, it is Minkowski distance with $p = 2$. This distance metric is valid when real-valued vector spaces and the following conditions are satisfied.

- Non-negativity: $d(x, y) \geq 0$
- Identity: $d(x, y) = 0$ if and only if $x == y$
- Symmetry: $d(x, y) = d(y, x)$
- Triangle Inequality: $d(x, y) + d(y, z) \geq d(x, z)$

Then, the Minkowski distance, in general, can be computed as follows.

$$d(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}. \tag{4}$$

If we set $p = 1$, then the $d(x, y)$ function become Manhattan distance, that is first-order distance metric. It computes the absolute value of the points, whereas Euclidean distance punishes the large distance points in a quadratic manner. Finally, we utilized the cosine distance, which can be computed by $1 - S_C(A, B)$.

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}}, \tag{5}$$

4.11. Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable [2]. Bayes' theorem states the following relationship, given class variable y and dependent feature vector x_1 through x_n [2]

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)} \tag{6}$$

Using the naive conditional independence assumption that

$$P(x_i | y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i | y) \tag{7}$$

for all, this relationship is simplified to

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)} \tag{8}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule [2]:

$$\begin{gathered} P(y \mid x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y) \\ \Downarrow \\ \hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y) \end{gathered} \tag{9}$$

4.11.1. Bernoulli Naive Bayes Classifier

Bernoulli Naive Bayes classifier applies Bayesian rule to a dataset with the prior assumption that the data comes from multivariate Bernoulli distribution [2]. Bernoulli Naive Bayes algorithm is a variant of Naive Bayes algorithm which assumes attributes are independent and do not affect each other and gives all features equal weight [2]. A most important distinction of Bernoulli Naive Bayes is that it uses binary value features like true/false or 1/0. This algorithm assumes the Bernoulli distribution's prior distribution and uses Bayes' Rule to maximize the posterior distribution.

$$p(x) = P[X = x] = \begin{cases} q = 1 - p & x = 0 \\ p & x = 1 \end{cases} \tag{10}$$

The decision rule for Bernoulli naive Bayes is based on

$$P(x_i \mid y) = P(i \mid y) x_i + (1 - P(i \mid y))(1 - x_i) \tag{11}$$

4.11.2. Gaussian Naive Bayes Classifier

Gaussian Naive Bayes classifier applies Bayesian rule to a dataset with the prior assumption that the data comes from multivariate Gaussian distribution [2].

4.12. Nearest Centroid Classifier

In the Nearest Centroid classifier, each class is represented by with class centroid. New samples are classified based on the distance to the class centroid, so it is very similar to the supervised version of K-means [2].

4.13. Bagging Classifier

Bagging classifier is an ensemble learning method that fits base classifiers to random subsets of the original dataset. Then, the predictions are aggregated either by voting or averaging to produce end class-wise predictions [2].

5. Results

In this section, we provide corresponding results. All tables and figures are self-explained.

	Accuracy	Precision	Recall	F1 Score	Fitted Time (s)
MLPClassifier	0.58	0.58	0.58	0.58	506.96
SVC	0.58	0.58	0.58	0.58	318.92
ExtraTreesClassifier	0.56	0.56	0.56	0.56	32.91
RandomForestClassifier	0.56	0.56	0.56	0.56	142.81
LinearSVC	0.55	0.55	0.55	0.55	253.10
LogisticRegression	0.54	0.54	0.54	0.54	9.08
KNeighborsClassifier	0.52	0.52	0.52	0.52	0.08
BaggingClassifier	0.51	0.51	0.51	0.51	192.71
RidgeClassifier	0.51	0.51	0.51	0.51	0.60
SGDClassifier	0.49	0.49	0.49	0.49	9.37
DecisionTreeClassifier	0.39	0.39	0.39	0.39	23.61
AdaBoostClassifier	0.39	0.39	0.39	0.39	103.01
NearestCentroid	0.38	0.38	0.38	0.38	0.13
Perceptron	0.37	0.37	0.37	0.37	5.93
GaussianNB	0.36	0.36	0.36	0.36	0.17

Table 1: Machine Learning Models and Performance Table

	Accuracy	Precision	Recall	F-1 Score	Fitted Time (s)
VotingClassifier([('MLP', MLPClassifier()), ('RFC', RandomForestClassifier(n_estimators=50, random_state=1)), ('SVC', SVC())])	0.58	0.58	0.58	0.58	368.62
VotingClassifier([('MLP', MLPClassifier()), ('SVC', SVC())])	0.58	0.58	0.58	0.58	370.14
VotingClassifier([('KNN', KNeighborsClassifier()), ('RFC', RandomForestClassifier(n_estimators=50, random_state=1)), ('GaussianNB', GaussianNB())])	0.53	0.53	0.53	0.53	310.07

Table 2: Meta Learning Models and Performance Table

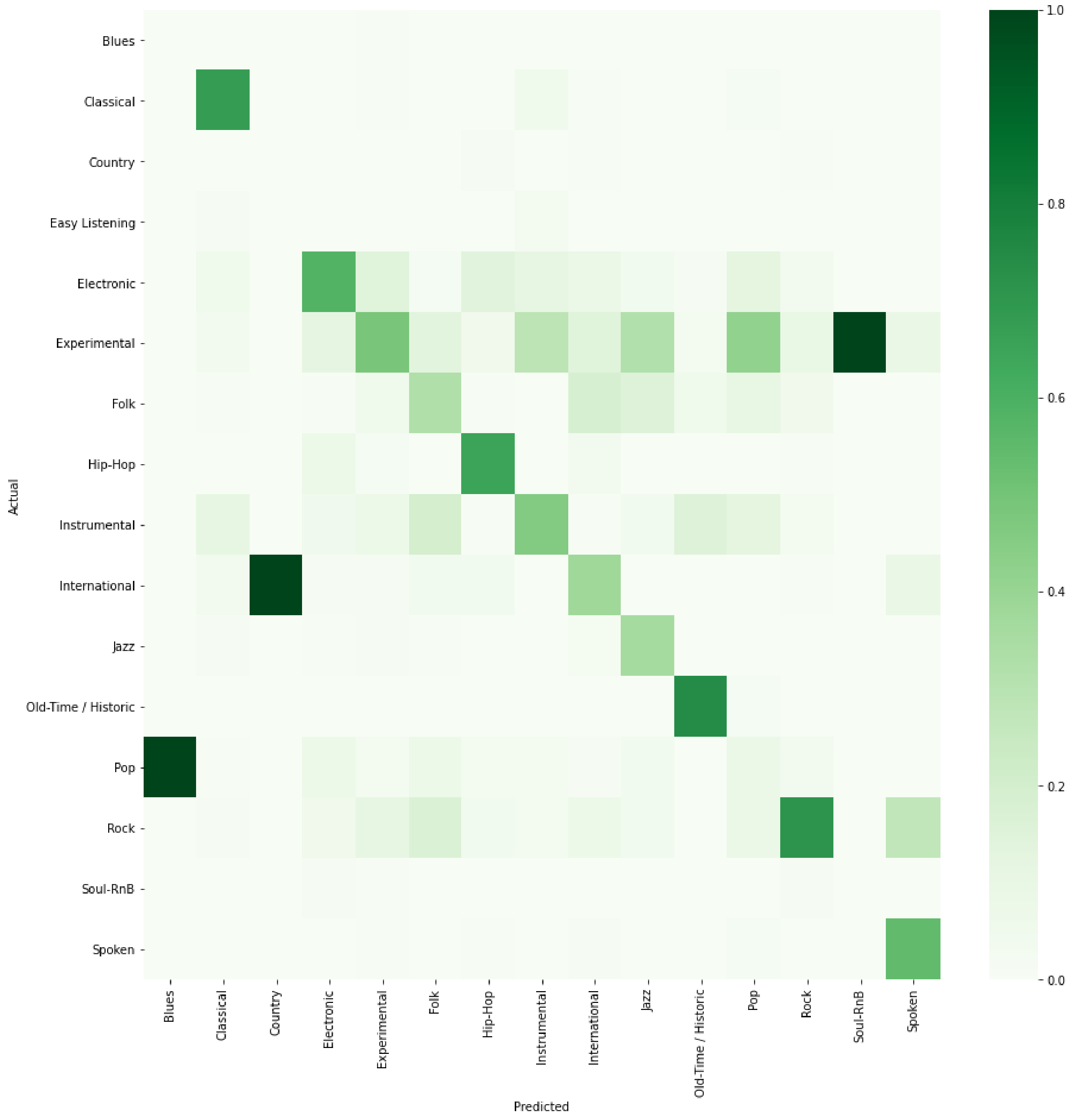


Figure 5: The heat-map of MLP Confusion Matrix

Predicted	Blues	Classical	Country	Electronic	Experimental	Folk	Hip-Hop	Instrumental	International	Jazz	Old-Time / Historic	Pop	Rock	Soul-RnB	Spoken
Actual															
Blues	0	0	0	0	9	0	0	0	0	0	0	0	4	0	0
Classical	0	72	0	0	10	1	0	2	1	0	0	1	0	0	0
Country	0	0	0	1	0	1	5	0	1	0	0	0	10	0	0
Easy Listening	0	2	0	0	3	0	0	1	0	0	0	0	0	0	0
Electronic	0	6	0	484	214	6	47	4	11	1	1	6	59	0	0
Experimental	0	4	0	98	709	37	17	10	18	8	2	20	159	2	1
Folk	0	1	0	4	84	88	2	0	23	4	4	5	84	0	0
Hip-Hop	0	0	0	65	30	1	216	0	4	0	0	0	7	0	0
Instrumental	0	12	0	41	116	53	3	16	1	1	11	6	49	0	0
International	0	4	1	14	25	11	14	0	47	0	0	0	11	0	1
Jazz	0	2	0	5	20	3	0	0	3	9	0	0	5	0	0
Old-Time / Historic	0	0	0	0	2	0	0	0	0	0	52	1	0	0	0
Pop	1	1	0	64	42	22	10	1	2	1	0	4	56	0	0
Rock	0	2	0	44	170	46	15	1	10	1	0	4	1168	0	3
Soul-RnB	0	0	0	10	11	0	1	0	0	0	0	0	21	0	0
Spoken	0	0	0	0	16	1	2	0	2	0	0	1	3	0	6

Table 3: MLP Confusion Matrix

6. Model Explainability

The Random Forest model is explained by the mean decrease impurity or Gini importance. The mean and standard deviation of impurity decrease accumulation inside each tree were used to determine Gini significance. The decrease in impurity is measured for features in each split.

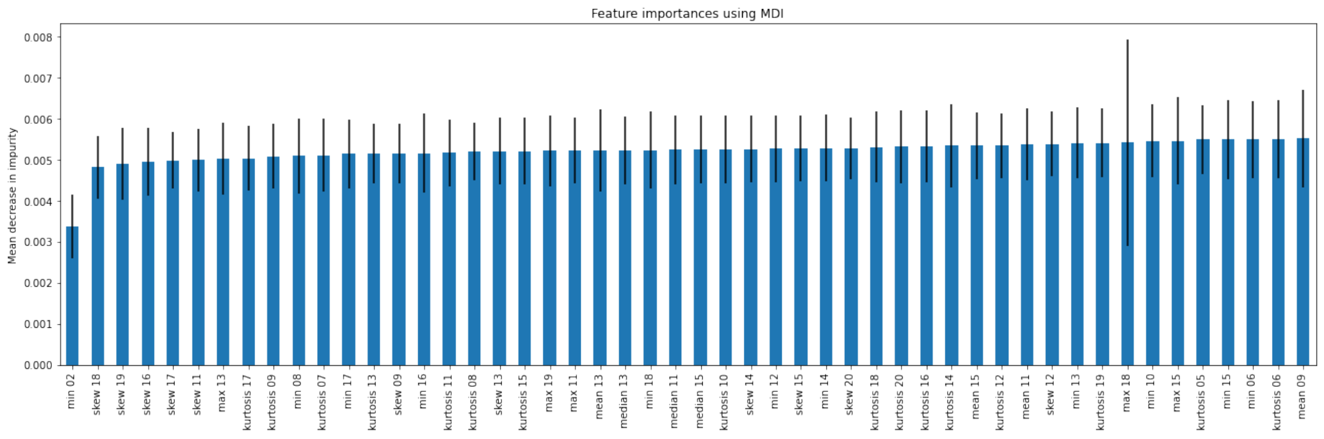


Figure 6: The error bars of Random Forest Mean Decrease Impurity feature importance

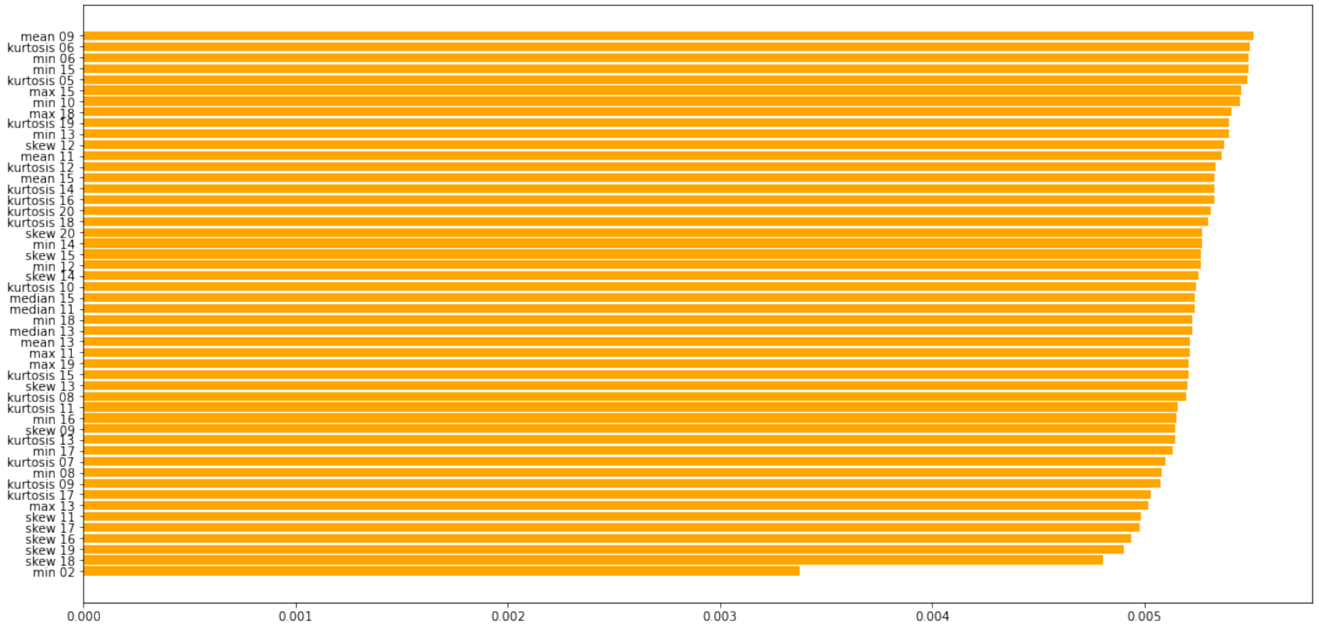
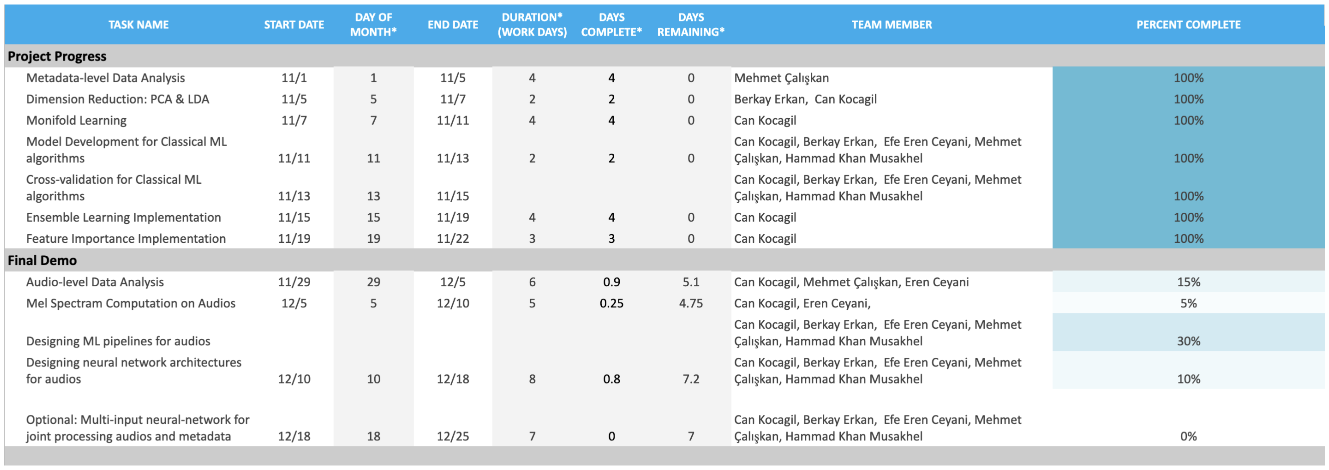


Figure 7: Sorted Random Forest Mean Decrease Impurity feature importance

7. Work Packages

As the project progressed, we completed the metadata-level data analysis, dimension reduction & visualization, machine and meta-learning model development, and feature importance implementation. The work packages, timelines, corresponding team members are depicted in the following figures.



TASK NAME	START DATE	DAY OF MONTH*	END DATE	DURATION* (WORK DAYS)	DAYS COMPLETE*	DAYS REMAINING*	TEAM MEMBER	PERCENT COMPLETE
Project Progress								
Metadata-level Data Analysis	11/1	1	11/5	4	4	0	Mehmet Çalışkan	100%
Dimension Reduction: PCA & LDA	11/5	5	11/7	2	2	0	Berkay Erkan, Can Kocagil	100%
Monifold Learning	11/7	7	11/11	4	4	0	Can Kocagil	100%
Model Development for Classical ML algorithms	11/11	11	11/13	2	2	0	Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel	100%
Cross-validation for Classical ML algorithms	11/13	13	11/15				Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel	100%
Ensemble Learning Implementation	11/15	15	11/19	4	4	0	Can Kocagil	100%
Feature Importance Implementation	11/19	19	11/22	3	3	0	Can Kocagil	100%
Final Demo								
Audio-level Data Analysis	11/29	29	12/5	6	0.9	5.1	Can Kocagil, Mehmet Çalışkan, Eren Ceyani	15%
Mel Spectram Computation on Audios	12/5	5	12/10	5	0.25	4.75	Can Kocagil, Eren Ceyani,	5%
Designing ML pipelines for audios							Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel	30%
Designing neural network architectures for audios	12/10	10	12/18	8	0.8	7.2	Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel	10%
Optional: Multi-input neural-network for joint processing audios and metadata	12/18	18	12/25	7	0	7	Can Kocagil, Berkay Erkan, Efe Eren Ceyani, Mehmet Çalışkan, Hammad Khan Musakhel	0%

Figure 8: Work Packages and Timeline

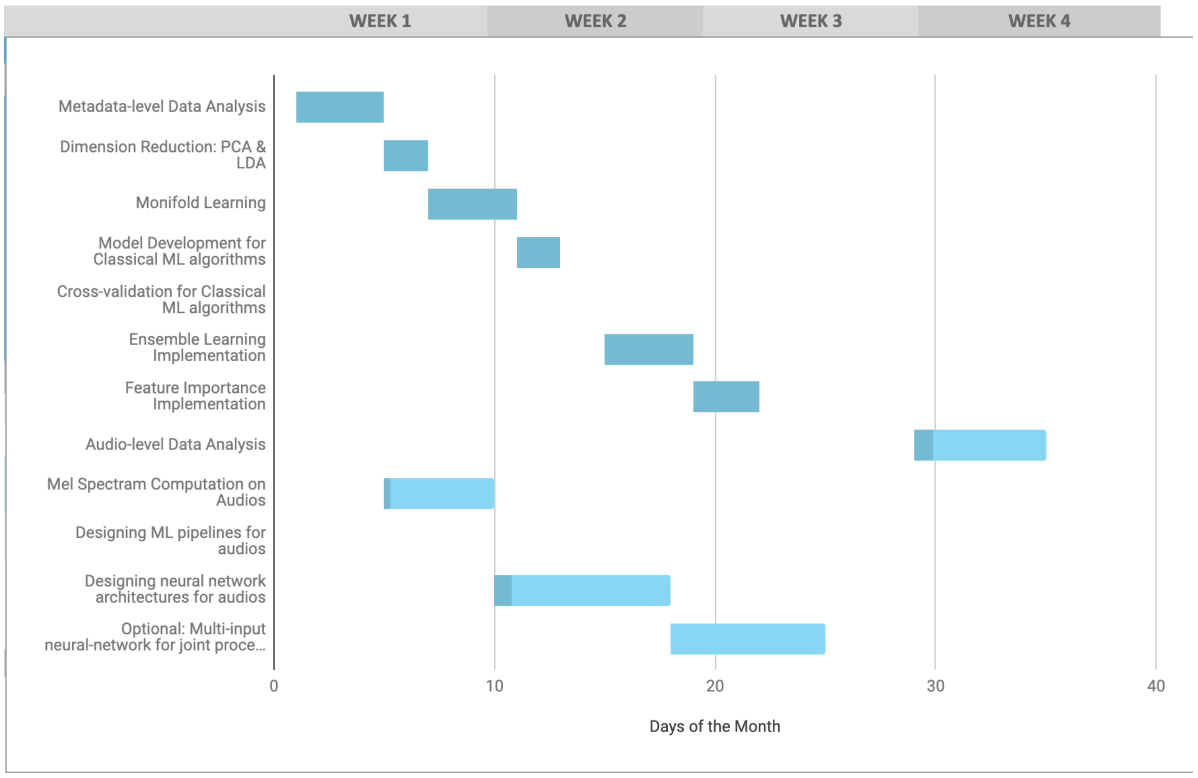


Figure 9: Gantt chart

We will be designing machine and deep learning-oriented recognition systems for computing audio-level signals. As of the start, audio-level data analysis will be made to discover and explore the dataset. Then, audio files will be processed by frequency-domain and Mel spectrum processes to extract semantic features. Then, conventional machine learning-based algorithms will be developed to fit audio features. As a final step, we will be designing learning systems by digesting the state-of-the-art neural network architectures for processing audio files to surpass our baseline accuracies.

8. Discussion & Conclusion

Musical genre recognition classifies musical collections into categories based on audio, user, and track features. Poor labeling and the non-triviality of feature extraction are seen as major roadblocks. In the first, musical genres are loosely defined, with people frequently arguing the genre of a song. In our scenario, executing data-centric computational tasks will be easier because the genres are supervised by the artists. As automatic extraction of semantic features is not differentiable and conceptualized by translating audio signals to visual signals and representing how the spectrum of frequencies evolves, there is no standard way to extract features in music information retrieval in the latter. The final hurdle will be attempted to be overcome utilizing spectrum-based representations before feeding the ML model.

Trend-aware classifier-based learning algorithms were designed with the intention of capturing the dynamic interrelationships of cultures in order to detect genres autonomously. Our classes will cover a variety of musical genres, from modern to jazz, and will be led and annotated by artists. We built machine learning methods to capture stochastic dynamics of music metadata and will build a neural networks for temporal dynamics of audio, from traditional ML algorithms like SVM to more advanced meta learning algorithms, while taking into account the socio-cultural environment. A multitude of machine learning-based techniques are employed to obtain the project's scope. To begin, we line with the basic and naive algorithms like Decision Trees and SVM to construct a baseline model to compare, and then we propose more complicated ensemble classification techniques like Random Forests and custom ones to compare our findings. The MLP and SVM classifiers were the winners, with a approximated accuracy of 60% in 16 classes.

References

[1] K. Benzi, M. Defferrard, P. Vandergheynst, and X. Bresson. FMA: A dataset for music analysis. *CoRR*, abs/1612.01840, 2016.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[4] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *CoRR*, abs/1712.05884, 2017.

[5] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[6] D. Wright. *Mathematics and music*, volume 28. American Mathematical Soc., 2009.