Appendix

I

Boxplots were used to identify the spread of numerical data. Histograms of each feature were created to visualize the frequency of values for the data. By doing this, we determined the skewedness of each feature. A heatmap was used to determine correlations between features. For features that were more correlated, we created scatterplots and used linear and quadratic regressions to find preliminary relationships. We also created histograms of categorical data to see the frequency of each value for the categorical data. Using domain knowledge, we grouped the various genres into 10 overarching genres. Bar graphs were used to visualize mean values for each numerical feature for the overarching genres. A random seed was used to split the data into 60% for training, 20% for validation, and 20% for testing.

II

We realized that many similar genres were classified into different groups, so we grouped the given genres into 10 overarching genres. These grouping are shown below. We removed the "Unnamed: 0" column, any rows with NaN values, and any songs with a duration of less than 30 seconds. We also removed songs with a tempo or time signature of 0, as these values do not make sense for songs.

```
{
    'pop': ['cantopop', 'j-pop', 'j-idol', 'k-pop', 'mandopop', 'pop', 'indie-pop',
'power-pop', 'pop-film', 'synth-pop'],
    'rock': ['alt-rock', 'alternative', 'hard-rock', 'indie', 'punk', 'j-rock', 'punk-
rock', 'psych-rock', 'rock', 'rock-n-roll', 'grunge', 'emo', 'rockabilly', 'guitar'],
    'metal': ['black-metal', 'death-metal', 'heavy-metal', 'metal', 'metalcore',
'grindcore'],
    'electronic': ['edm', 'electro', 'electronic', 'house', 'garage', 'j-dance',
'hardcore', 'hardstyle', 'industrial', 'techno', 'trance', 'dubstep', 'idm', 'minimal-
techno', 'progressive-house', 'chicago-house', 'deep-house', 'detroit-techno',
'disco', 'drum-and-bass', 'dub', 'club', 'dance', 'dancehall'],
    'hip-hop': ['hip-hop', 'rap', 'r-n-b', 'breakbeat', 'trip-hop'],
    'jazz': ['jazz', 'blues', 'soul', 'funk', 'ska', 'gospel'],
    'classical': ['classical', 'opera', 'piano'],
    'world': ['afrobeat', 'brazil', 'sertanejo', 'british', 'latin', 'latino',
'samba', 'salsa', 'reggae', 'reggaeton', 'tango', 'world-music', 'indian', 'iranian',
'turkish', 'malay', 'mpb', 'pagode', 'forro', 'french', 'german', 'spanish',
'swedish'],
    'folk': ['folk', 'bluegrass', 'country', 'singer-songwriter', 'songwriter',
'honky-tonk'],
    'misc': ['acoustic', 'ambient', 'anime', 'children', 'chill', 'comedy', 'disney',
'happy', 'party', 'study', 'sleep', 'show-tunes', 'new-age', 'kids', 'goth', 'groove',
'romance', 'sad']
}
```

III

Popularity, duration, danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, and tempo were used in both linear and quadratic regression. We used a sequential feature selector, forward selection, to determine the most influential features. We used R^2 as the scoring metric for cross-validation to identify optimal features. Because we used this metric, we found that regularization was not necessary. Overall, we determined that the importance of certain features varied by genre group. For

example, energy was more prominent for rock and electronic genres, acousticness and liveness were more prominent for folk and classical genres, and danceability was more prominent for pop and hip-hop genres.

IV

For logistic regression, the track-genre column was encoded using a label encoder to map genres to numerical labels. Non-numeric columns were converted via one-hot encoding, and data was standardized. We calculated accuracy, recall, and F1 score and plotted ROC curves and found the AUC to determine the model's ability to distinguish between classes. The default solver lbfgs used L2 regularization, so we did use regularization. However, removing the regularization did not affect the metrics.

We found that there were different key features for each genre. Energy contributed largely to rock, metal, and electronic music. Danceability contributed largely to pop and hip-hop identification. Acousticness contributed largely to classical and folk music. Loudness contributed largely to metal music.

V

We used KNN, a decision tree, and a random forest to try to classify genres of our data. For all of these, we one-hot-encoded all categorical data. While this isn't necessary for the tree models, it is for KNN, since distance between two points for categorical data does not make sense if the data is numeric.

Out of these three, the random forests had the best accuracy on the test set at 0.56, and was the best method by far for classifying the data. As expected, KNN was inefficient in analyzing our data due to the dataset's high dimensionality, and the presence of a lot of categorical data where distance makes much less sense. Decision trees make sense as a good model for the data, since there is a lot of overlap and the decision tree can make minute distinctions at each split. And random forests, as an ensemble version of decision trees, make sense as being more accurate.

VI

We applied one-hot encoding and scaled and mean-centered the data before doing PCA. The scree-plot results were unclear as to how many clusters should be used, but the top 4 principal components accounted for about 30% of the variability. Plotting the principal components showed that there was not clear separation of clusters due to heavy overlap.

After applying logistic regression to the PC-transformed dataset, we still produced relatively low accuracy rate of about 30%; the silhouette scores were also low.

We tried applying a Gaussian Mixture model on the PC-transformed dataset, which resulted in visually similar plots, but very low silhouette score. Even reducing the dimensionality of the GMM resulted in lower silhouette scores. We concluded that PCA transforming the data would not really help with determining genre. Since we did not have many features to begin with, we also believed that feature reduction was not a priority.

We also applied K-means clustering, agglomerative (hierarchical) clustering, and gaussian mixture models to the non-PCA-transformed data. All of these had terrible rand index with genre and <0.25 silhouette scores. Thus, we concluded that clustering would not help us with our problem.

VII

We used the dataset without string features, and one-hot-encoded the categorical features.

We went through a number of models, but ReLU, softmax, and dropout all significantly decreased the accuracy, so we ended up with a simpler model. Our model uses 2 hidden layers of size 64 using sigmoid activation, and log softmax output activation.

We trained using the Adam optimizer using cross entropy loss, and reported accuracy metrics at the end. We also included weights to account for class imbalance. We learned an initial learning rate via pytorch's `lr_find` function, and used the StepLR learning rate scheduler which reduced learning rate by 90% every 10 epochs. Our final accuracy was about 20%.

However, looking at accuracy metrics, recall for some classes like classical and metal was really high, indicating that if the model predicts that a song is classical or metal, then it is likely that it is right.

Interestingly, the model never predicted pop or hip-hop. This might be due to class imbalance, though pop is the most populated class and hip-hop the least. It might also be that the class weights undercompensated for hip-hop and overcompensated for pop. However, this was very common for many runs using many different hyperparameters.

VIII

We primarily used library methods to tune hyperparameters. For random forests and decision trees, we used `GridSearchCV` to tune max depth, critereon (GINI vs entropy), number of estimators (for RF), and other splitting/leaf parameters for decision trees, which checked every combination of hyperparameters. This did not really affect results, however. For neural networks, we used `find_lr` to find our initial learning rate, and used a learning rate scheduler to tune learning rate over the course of the training. We also, over the course of selecting the model, tried different batch sizes, hidden layer sizes, number of hidden layers, activation functions, and loss functions, all of which we "manually tuned" by checking the accuracy results after training.