Predicting Spotify Genres

A. Main Document

i. The Spotify Data Set

For this project, we used a dataset derived from HuggingFace, which contains a tabular dataset of Spotify tracks in CSV format with 114,000 rows and 20 columns. It was collected using the Spotify Web API and Python. The unit of observation is one Spotify track (or song). The dataset includes the following features:

- **track_id**: The Spotify ID for the track
- **artists**: The artists' names who performed the track
- **album_name**: The album name in which the track appears
- **track_name**: Name of the track
- **popularity**: A value between 0 and 100, with 100 being the most popular
- **duration_ms**: The track length in milliseconds
- **explicit**: Whether or not the track has explicit lyrics
- **danceability**: Suitability of a track is for dancing. A value of 0.0 is least danceable and 1.0 is most danceable.
- **energy**: A measure from 0.0 to 1.0 that represents a perceptual measure of intensity and activity.
- **key**: The key the track is in.
- **loudness**: The overall loudness of a track in decibels (dB)
- **mode**: Modality (major or minor) of a track
- **speechiness**: The presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.
- **acousticness**: A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic
- **instrumentalness**: Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content
- **liveness**: Detects the presence of an audience in the recording.
- **valence**: A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track.
- **tempo**: The overall estimated tempo of a track in beats per minute (BPM).
- **time_signature**: An estimated time signature.
- **track_genre**: The genre in which the track belongs.

The target variable is **track_genre**, representing the track's genre.

ii. Overview of the Problem

The focus of this project is to classify the genre of a track based on its feature attributes, and ultimately to predict the genre of a new track given its feature attributes. The data set includes a total of 114 unique genres at relatively balanced counts. However, to reduce the complexity of the problem and enable accurate predictions, we reduce the data to tracks belonging to twenty different genres, selected from the original pool of 114. This set of twenty was selected because they are the twenty most populated classes after preprocessing the data, further detailed in Appendix ii.

iii. Key Methodology

The key methodology that proved most effective for addressing the genre prediction problem began with rigorous data preprocessing. This step involved removing null values, duplicate rows, and irrelevant features such as artist names and track IDs, which do not contribute meaningful information for genre prediction. Feature selection was then performed using LASSO logistic regression and random forests, both of which are robust methods for identifying the most important predictors in high-dimensional datasets. By selecting the top 10 features that had the greatest impact on genre classification, we reduced model complexity while retaining the most relevant information. Finally, we trained a random forest model, which has proven to be a strong classifier for multi-class classification tasks, yielding an accuracy of 68%. Random forests work by aggregating the predictions of multiple decision trees, each trained on a different subset of the data, which helps to reduce overfitting and improve model generalizability. This ensemble approach effectively captures complex relationships in the data and improves classification accuracy.

iv. Results

Model	Test Accuracy	Validation Accuracy
Logistic Regression	55.0%	53.1%
Random Forests	**65.0%**	**64.1%**
Decision Trees	55.2%	50.1%
4 Layer Neural Network	57.7%	57.4%
6 Layer Neural Network	61.0%	60.4%
10 Layer Neural Network	59.1%	51.9%

We find that Random Forests yield the highest test and validation accuracy among the six models evaluated. To ensure robust performance comparisons, we employed 5-fold cross-validation, which splits the dataset into five subsets, iteratively training on four and validating on the fifth. This approach helps mitigate overfitting and provides a reliable estimate of model performance on unseen data.

Random Forests excelled due to their ability to handle high-dimensional data, robustness to noise, and the inclusion of feature randomness, which reduces overfitting compared to simpler models like decision trees. Logistic regression assumes a linear relationship between the features and the target variable, which may not capture complex, nonlinear patterns in the data. While it is efficient and interpretable, its performance is limited in datasets with intricate interactions among features. In contrast, Random Forests excel in such scenarios by creating nonlinear decision boundaries through multiple decision trees, allowing them to better adapt to complex data structures. On the other hand, neural networks often require a significant amount of data, computational resources, and hyperparameter tuning to achieve their full potential. In our case, the dataset size may have been insufficient to leverage the advantages of neural networks fully, leading to overfitting or suboptimal generalization.

Additionally, the model provided insights into feature importance, helping to identify key drivers of prediction. However, despite its strengths, the Random Forest method has limitations. It requires careful tuning of hyperparameters (e.g., number of trees, maximum depth) to optimize performance and is computationally intensive compared to linear models. Furthermore, its ensemble nature makes it less interpretable, which may not align with applications requiring clear, explainable models.

Future improvements could include dimensionality reduction techniques or feature engineering, as this could further enhance performance by simplifying the feature space and reducing noise. By addressing these limitations and exploring complementary methods, we aim to refine our predictive capabilities further.

B. Appendix

1. Exploratory Data Analysis

The descriptive statistics of the pre-processed data set shows that most of the features ranges between [0,1] while the rest vary drastically: some ranges between [0,100] (e.g. `popularity`), some [0,11] (e.g. `key`), or even to negative values (e.g. `loudness`). Thus, it will be important to standardize and/or mean center these features accordingly for the remainder of the project.

Like some of the categorical predictor variables, many of the continuous predictor variables (duration_ms, danceabilty, energy, loudness, speechiness, acousticness, and instrumentalness) follow non-normal distributions with a handful of outliers as illustrated by their overlaid density and box plots. This will be addressed by scaling and selecting features appropriately depending on the assumptions made by different modeling techniques (e.g. distance-based clustering methods may be sensitive to outliers, linear regression assumes normality of residuals and homoscedasticity).

Suggested by the heatmap of each feature's mean value per genre, some of the predictor feature class imbalance may be associated with specific classes in the response variable `track_genre`. For instance, `explicit`-ness is especially present in genres like grindcore, black-metal, and hardstyle while genres like sertanejo, gospel, and disney lean toward being on a major scale. This indicates that applying imputation to accommodate for these imbalances may negatively affect the class balance in the response variable "track_genre," which is arguably more important than that of predictor variables.

The correlation matrix between predictor variables found multicollinearity. Specifically the absolute correlation coefficient was greater than 0.5 between valence and danceability (0.52), energy and loudness (-0.73), energy and acousticness (-0.73), and loudness and acousticness (-0.53). Their scatterplots further illustrate how their relationships are non-linear. It is important to note these highly-correlated features since modeling techniques (e.g. logistic regression) can make assumptions about the absence of multicollinearity.

We explored how each numerical feature's distribution varies according to the categorical features "key," "explicit," and "mode" via category-annotated density plots. The numerical features, except perhaps "energy" and "acoustic", follow relatively similar distributions between the 11 "key" classes. Likewise, the distributions were similar between the 2 "explicit" classes and the 2 "mode" classes. We also explored how each categorical feature's distribution varies across the top 20 genres via stacked bar graphs.

2. Pre-processing

The dataset was given by 114000 observations across 21 features. After familiarizing ourselves with the feature descriptions and the basic structure of the data, we dropped the column "Unnamed: 0" unique for each observation and removed a single observation with null values along with 894 complete duplicates. However, we found that "track ID" was not unique for each observation, indicating there were duplicates in terms of "track ID." We discovered the reason behind this was that each unique track, instead of observation, was associated with a track ID. After isolating each feature as a potential explanation for the track ID duplicates, we found that there were 38948 tracks were duplicates of each other by being listed under multiple genres, and likewise for 9238 tracks

under different albums, 293 tracks under different popularity ratings, 2 tracks under different track names, and 2 tracks under different tempo. We left the genre duplicates intact to preserve information for genre classification. The highest popularity rating was kept for identical tracks with multiple popularity ratings to best reflect their performance. Identical tracks with multiple track names, album names, and tempo were handled as well by keeping their first occurrences.

The “track_name”, “album_name”, and “artists” features were dropped following this process since they are not within the scope of this project. This leaves us with 106811 observations across 15 features, excluding the response variable “track_genre.” The ordinal features “popularity", "key", and "time_signature" were given by numerical values, thus, are already encoded into quasi-interval variables and no further action is needed. The nominal features "explicit" and "mode" were one-hot encoded since their categories lack a natural order. These categorical features all exhibit some degree of class imbalance. Notably, “explicit” is dominated by its “non-explicit” class, “time_signature” is dominated by its “4/4” class, and “popularity” is dominated by its “0” rating class. However, not only are large proportions of data missing on these features, their missingness are associated with specific genres. Thus, imputation may not be reliable in this case, especially when our response variable “track_genre” is already very balanced. Observations for the top 20 most common genres were kept for the project. After preprocessing the dataset, a version with the original values were kept for conducting EDA, a z-score normalized version and a min-max scaled version with 80/20 train-test split were prepared for modeling techniques with different data preferences

3. Regression Analysis

Since the main problem we address with this project involves classification and not regression, a linear regression analysis is not directly useful for our goals. However, in exploring the data set and settling on a topic for the primary problem to explore, we conducted a regression analysis using loudness as the response variable and energy as the predictor variable.

The initial model achieved an R-squared value of approximately 0.593 on the training data, meaning that 59.3% of the variability in loudness is explained by energy. Considering that this model uses only one predictor variable, the model is quite good, though the analysis is brief and can be greatly expanded upon.

We attempted to improve the initial model by employing ridge regression with an lambda value of 1. Ridge regression applies an L2 penalty on the least squares loss function and ultimately prevents overfitting by encouraging the coefficient parameters toward zero. The R-squared value of this ridge regression model was also approximately 0.593, indicating that no accuracy was lost. This is explained by comparing mean-squared errors (MSE) of the initial model fit on the training and testing sets. The quantities were similar (7.809 vs 8.350, respectively), indicating that there was little need for regularization against overfitting. Thus, the ridge regression model did not change much from the original model.

The prediction of loudness using other/multiple predictors is a topic that could be further explored, but is outside of the scope of this project.

4. Logistic Regression

Logistic regression, when compared to standard regression analysis, was much more applicable to the research problem we were looking to solve. Since logistic regression aims to predict a value from 0 to 1 for the probability for a class label, we were able to leverage it to predict the genre of each song in the dataset.

Logistic Regression gave us our first insights into the importance of each of the features for each genre and collectively after averaging the absolute values of each feature importance. Features like popularity and loudness were consistently important for each of the genres, with features like acousticness and danceability also playing a heavy role in the predictive ability of the model. Features such as tempo, mode, time signature, and key had very low importance to the performance of the model.

Our logistic regression model ended up having an accuracy of 0.55, which when compared to some of our other models was a bit low. Later cross validation gave us an accuracy of 0.53. However, the information regarding which features were most important in determining the genre of a song was valuable information that would show up more than once in our research.

5. Random Forest

Random forest uses a large number of decision trees to come to a decision about the most likely class label of a particular observation based on the value of a predictor, whether it is above/below a certain threshold or a true/false feature.

Our random forest model mirrored some earlier results seen in the logistic regression analysis, with features like popularity and acousticness once again being features with importance while time signature and explicit were once again at the bottom of the list.

However, in contrast with the logistic regression, our random forest model resulted in the highest accuracy of every model we fitted. We were able to reach an accuracy of 65% and a cross validation accuracy of 64.1%.

This model ended up being the best performance of each of the ones we fitted. Random forest's handling of outliers and collinearity most likely contributed to its success in comparison to other models. It also fits the context of our problem, with several numeric and boolean variables that we are aiming to predict a categorical variable from. Its process of using multiple decision trees is vital for preventing overfitting. However, random forest uses a significant amount of resources to fit the model, so runtimes ended up being pretty long in our application.

6. Clustering Analysis

Principal Component Analysis is a technique for dimensionality reduction when a data set is very high-dimensional (i.e., contains many features). The goal is to simplify the data set while retaining as much information or variability in the data as possible.

Clustering is an unsupervised machine learning technique that creates groups in the data. For this clustering analysis, we attempt to cluster based on time signature.

Clustering is susceptible to a phenomenon known as the curse of dimensionality, in which a data set is so high-dimensional and complex that clustering is difficult to perform and largely inaccurate, as the more complex a data set is, the less meaningful distance metrics become. Thus, reducing the dimension of the data set may aid in clustering effectiveness. Our approach involves performing k-means clustering on the

data, then performing principal component analysis to create a transformed (simpler) data set, performing k-means clustering again on the new PCA-transformed data, and finally comparing evaluation metrics for the two clustering schemes.

Results showed that the first two principal components capture about 34.16% of the variability in the data, and after that, each principal component made a small, consistent contribution. Unfortunately, keeping only the first two principal components would simply result in a data set that does not capture nearly enough information from the original data to yield generalizable insights from analysis. Moreover, if we want to retain most of the information in the original data, let's say 90%, then we would need to keep the first ten principal components, which is not a very successful dimensionality reduction down from fourteen original features.

Keeping the first nine principal components and clustering with the new PCA transformed data set, we obtained only a slightly higher silhouette score of 0.1296, compared to a score of 0.1098 before performing PCA. Silhouette score is a measure of how tightly and distinctly the data is clustered, where 1 is tightly clustered and 0 is loosely (and indistinctively) clustered. It is one measure of effectiveness for a clustering algorithm. The calculated silhouette scores for both clustering schemes suggests that the effects of PCA are minimal for this data set, so we did not employ it in the main analysis of our classification of track genres.

The same approach may be taken to cluster the data based on other categorical features such as key, though we did not explore these techniques further so we could focus on the key methodology.

7. Neural Network

Neural networks are a powerful tool for multi-class classification tasks, capable of modeling complex patterns in data through their layered architectures. This makes them suitable for predicting Spotify genres, given the non-sequential, tabular nature of the dataset. However, due to the lack of spatial, temporal, or time-sensitive data, architectures like CNNs and RNNs, commonly used for image or sequence processing, are not ideal for this task. Instead, we designed three feedforward neural network architectures with 4, 6, and 10 layers, incorporating dropout and batch normalization layers to mitigate overfitting and stabilize training. To further optimize model performance, we tuned key hyperparameters, including batch size, dropout rate, and learning rate, through systematic experimentation.

The choice of architecture depth was driven by a desire to balance model capacity and training efficiency. The 4-layer model served as a baseline, providing a simpler structure to establish performance benchmarks. The 6-layer model offered additional depth, hypothesized to capture more complex relationships in the data without introducing significant overfitting. The 10-layer model, the most complex of the three, was designed to evaluate the trade-off between capacity and generalization. To ensure robust training, we employed ReLU activation functions for non-linearity, softmax for the output layer, and categorical cross-entropy as the loss function, given the multi-class nature of the task.

The results demonstrated that model performance improved with increased depth up to a point, beyond which overfitting became apparent. The 4-layer model achieved an accuracy of 57.7%, establishing a reliable baseline with stable convergence and minimal overfitting. The 6-layer model outperformed the

others with an accuracy of 61.0%, benefiting from its enhanced capacity to learn complex patterns while maintaining generalization. The 10-layer model, while theoretically more powerful, took significantly longer to train despite only achieving an accuracy of 59.1%, which is lower than the 6-layer model. This suggests that the dataset's size and complexity were insufficient to fully leverage the additional depth.

Overall, the 6-layer model proved to be the most effective architecture, achieving the highest accuracy and demonstrating a good trade-off between capacity and generalization. These findings underscore the importance of tailoring architecture and hyperparameter selection to the characteristics of the dataset. Future work could explore advanced regularization techniques, such as early stopping, to further enhance the performance of deeper architectures. Additionally, expanding the dataset or augmenting it with derived features could better support the capacity of more complex models.

8. Hyperparameter Tuning

Hyperparameter tuning played a pivotal role in optimizing the models. For the neural networks, batch size impacts how often weight updates occur; smaller batch sizes often generalize better but require more computational resources. Dropout rates were adjusted to regulate overfitting by randomly deactivating neurons during training, encouraging robustness. Learning rate, a critical parameter controlling the step size of weight updates, was carefully tuned to balance convergence speed and stability.

Across all deep learning architectures, the optimal hyperparameters varied. The 4 layer model performed best with a learning rate of 0.001, batch size of 64, and dropout rate of 0.2. Both the 6 and 10 layer model performed best with a learning rate 0.001, batch size of 32, and dropout rate of 0.1. This makes sense, since more complex models require smaller learning rates and smaller batch sizes to achieve stable training and reduce overfitting.

For the random forest, decision tree, and logistic regression models, we utilized grid search to systematically explore combinations of hyperparameters and employed cross-validation to ensure robust evaluation. In the random forest, the number of trees (n_estimators), maximum tree depth (max_depth), and the minimum samples per split and leaf (min_samples_split and min_samples_leaf) were tuned. The optimal configuration balanced model complexity and generalization, achieving higher accuracy and lower variance. Similarly, decision trees were optimized for depth, and leaf node size, improving predictive power without overfitting. Logistic regression benefited from regularization parameter tuning (C) to control overfitting, ensuring a balance between model simplicity and accuracy. These fine-tuning efforts collectively enhanced model performance, enabling more reliable predictions and insights from the data.