Appendix

Link to Appendix Code

i. EXPLORATORY DATA ANALYSIS - Check-In 1 Notebook

Explain the exploratory data analysis that you conducted. What was done to visualize your data and split your data for training and testing?

Before beginning to consider the methodologies and models we would use to model the data, we first conducted exploratory data analysis (EDA). We began by visualizing the distribution of the popularity feature using a histogram, which revealed a large number of songs with a popularity score of zero. These songs were deemed irrelevant for the analysis and were removed from the dataset. Next, a correlation heatmap was generated for the numeric variables, excluding irrelevant columns such as time_signature, Unnamed: 0, key, and mode, to explore relationships between features. This correlation heatmap was used to analyze feature collinearity, as this coule potentially affect the results of certain models.

ii. DATA PRE-PROCESSING AND FEATURE ENGINEERING

What data pre-processing and feature engineering (or data augmentation) did you complete on your project?

For data pre-processing and feature engineering, we first checked for missing values in the data. Since we only found one track with a missing value, we removed the row from the dataset. Furthermore, based on the data, we noticed tracks were duplicated in the playlist if they were listed in more than one album, even if the track name, artists, and genre were the same. We decided to remove those duplicates to reduce redundancy and overrepresentation of certain tracks. Next, since the goal of our project is to predict popularity, we removed tracks that corresponded to popularity scores of 0 to leave only meaningful data. Finally, we created a new column feature, popularity_rating, by categorizing popularity into three bins representing low, medium, and high popularity, enabling further analysis of popularity ratings.

iii. LINEAR REGRESSION ANALYSIS - Check-In 2 Notebook

How was regression analysis applied in your project? What did you learn about your data set from this analysis and were you able to use this analysis for feature importance? Was regularization needed?

One of the first methodologies we tried to model the data with was linear regression. We began by extracting a sample of the data to run linear regression on with the strongest positively correlated variable with popularity, loudness, from our correlation heatmap in our exploratory data analysis. After seeing the results of these, we moved on to creating a linear regression model

using all of the numeric predictor variables in the data and popularity rating as our response variable.

The predictor variable we examined first was loudness because it had the highest magnitude positive correlation with popularity, with a correlation coefficient of 0.079. From running the model with loudness as the predictor variable, we observed that the correlation coefficient is relatively low, 0.253, indicating a very weak positive correlation. To better understand this relationship, we created a scatterplot and added the least squares regression line, where the regression line provides a visual representation of the least squares fit.

We used evaluation metrics such as root mean squared error (rMSE), mean absolute error, mean absolute deviation (MAD), correlation, and R^2 score to provide insights into the performance of the linear regression model. The evaluation measures suggest that the linear regression model with loudness as the sole predictor is not effective at explaining or predicting popularity. This is consistent with the low correlation coefficient (0.0673) and weak R^2 value (0.00454). Additional predictors or a more complex model might be needed to capture the variability in track popularity.

To further explore whether linear regression would be an effective model, we proceeded with the least squares linear regression as our model, but with more predictor variables to see if that would improve our model's performance. We trained the linear regression model, and evaluated the model's performance using a 5-fold cross-validation, which revealed consistent results across each fold. After this, we trained the model on the training dataset and used the model to make predictions on the validation test.

To evaluate the model, we calculated the MSE, MAE, and R^2 scores. Both in the training and validation sets, we found an MSE of 315 and R^2 score of ~0.80. We also performed ridge regularization to see if it would improve the model performance, where the best alpha value was chosen based on the lowest cross-validated MSE. However, even after performing the ridge regression, model performance did not improve as we saw almost identical MSE and R^2 metrics.

The results of these linear regression models suggest that the models are too simple to capture the relationship we are trying to gauge and that we will need to explore other methodologies to tackle our problem.

iv. LOGISTIC REGRESSION ANALYSIS - Check-In 3 Notebook

How was logistic regression analysis applied in your project? What did you learn about your data set from this analysis and were you able to use this analysis for feature importance? Was regularization needed?

For our next methodology, we tried utilizing a logistic regression model to classify the music tracks into popularity categories. To do this, we first created a new feature column, popularity_class, where tracks would be classified as not popular or popular, so we would have a binary classification problem. This was done by splitting based on the popularity feature value, where scores of 1-50 were labeled not popular, coded as 0, and scores of 51-100 were labeled

popular, coded as 1. We focused on numerical variables, excluding some features that were not related, such as explicit, track_id, key, and time_signature. We learned there was a slight imbalance in the number of tracks under each popularity bin, with almost double the number of tracks labeled as unpopular compared to the number of popular tracks.

We then ran a logistic regression model, using each individual numerical column as a predictor against the probability_class, split into training and validation sets, and calculated model metrics such as the confusion matrix, accuracy, error rate, precision, recall, and calculated the Area Under Curve (AUC) to evaluate model performance. The threshold was set to 0.5 due to the binary model. The performance metrics revealed 0.72 prediction accuracy, 0.28 prediction error, 1.0 True Negative Rate (TNR), and 0.0 True Positive Rate (TPR) with 0.0 precision, recall, and F1 score. These results reveal inadequate predictions, with highly skewed accuracy towards predicting true negative rates, which could be due to the inherent imbalance of track popularity count in the dataset using this binary model. The logistic regression AUC was calculated to be 0.54, just above the probability of flipping a coin. Our project also used 5-fold cross-validation to assess the model stability and performance metrics in each fold, which resulted in 0.5 AUCs for all folds and accuracies between the 0.6 to 0.8 ranges. From this cross-validation, we calculated the optimal threshold to be 0.26. This optimal threshold is determined by maximizing the difference between TPR and False Positive Rate (FPR), and the resulting confusion matrix is compared to the original matrix. Rerunning our model with this threshold resulted in 0.42 prediction accuracy, 0.58 prediction error, 0.81 TPR, 0.27 TNR, 0.3 precision, 0.81 recall, and 0.44 F1 score. This optimal threshold helps account for the imbalance in binary categorization, slightly improving our precision, recall, and F1 score metrics, and balancing out the TPR and TNR.

Even after applying the code to our optimal threshold, the performance metrics show unsatisfactory results, which suggest logistic regression with our binary classification model may not be helpful for analysis in our dataset nor for looking at feature importance, although the variables in our dataset are standardized and may be easier to interpret. Regularization was not used in our case, as the dataset was clean and did not show high multicollinearity or overfitting.

The results of these logistic regression models suggest that the binary classification models are too simple to capture the relationship we are trying to gauge and that again, we will need to explore other methodologies to tackle our problem.

v. PCA AND CLUSTERING - Check-In 5 Notebook

How were PCA and clustering applied on your data? What method worked best for your data and why was it good for the problem you were addressing?

In our analysis, Principal Component Analysis (PCA) and clustering were utilized to uncover patterns in the dataset and simplify the complexity of the data. Between the two methods, PCA worked better and produced clearer results when compared to the clustering methods, as K-means clustering yielded largely uninterpretable results and did not provide

actionable insights for the goal we were working towards. This was likely due to the fact that we included a large set of features to help with our predictor variable, resulting in difficulty of clear groupings to determine popularity of a song track, resulting in inconclusive clustering. In the end, this emphasized both the lack of clarity and impracticality of the clustering method for our context.

For the PCA method applied, the dataset was composed of continuous variables such as "danceability", "energy", "valence", "tempo", "acousticness", "instrumentalness", "liveness", "speechiness", and were standardized using a z-score normalization. PCA was then applied via Singular Value Decomposition (SVD), which generated principal components ranked by their explained variance. Thus we were able to capture the proportion and cumulative variability explained.

We then plotted the cumulative variability explained against the number of principal components, revealing that the number of principal components did not greatly affect the proportion of variability explained (with the variability explained increasing evenly across PCs). Our Scree plot did not show a significant elbow curve and only reached a cumulative variability explained at the 6-7 PC. We then looked at the feature importance of each component and saw that our first principal component (PC1) is contributed mostly by instrumentalness, valence, and tempo, whereas liveness, speechiness, and energy contribute the most to our second principal component (PC2). This contribution of distinct features to our first two principal components suggests that even though our PCA model explains a smaller, yet even, proportion of the variability across the components, it is still able to capture different patterns in the data.

The problem to be addressed was evaluating feature importance among the variables and dimension reduction. By transforming the original correlated features into independent principal components, we retain only the top components that explain most of the variance, simplifying the dataset for analysis. This reduction makes computation more efficient without significant information loss. Additionally, projecting the data into a lower-dimensional space allows us to visualize and identify meaningful structures, such as clusters of tracks based on their audio characteristics. These insights can inform further analyses, such as mood-based playlist creation or genre classification, enhancing our understanding of the dataset's underlying pattern
From our PCA, we found that in order to maintain 80-90% variance, we needed at least 7 principle components, meaning we reduced the feature dimension slightly. However, from our scree plot, we see a steady curve, where each principal component explains a similar proportion of the variance, rather than a steep curve or bend. From this, we concluded that PCA may not be the most helpful method for dimension reduction with this Spotify tracks dataset.

vi. NEURAL NETWORK - Check-In 6 Notebook

Explain how your project attempted to use a neural network on the data and the results of that attempt.

For this project, we wanted to use neural networks for extracting feature importance. To do this, we approached it by performing a feature permutation test. This test is a method for quantifying the importance of each feature in a predictive model by evaluating the drop in accuracy when the values of a specific feature are randomly shuffled (or permuted). By doing so, we break the relationship between that feature and the target variable, effectively removing its predictive power and extracting the importance of that feature for the model's prediction.

To do this, we first trained a neural network on all the features to predict the popularity of the song. The architecture that we used included two hidden layers, one with 64 neutrons and one with 32, with a ReLU activation function which fed into a 3 neuron output layer with a softmax activation to classify the song into one of the 3 popularity bins. The trained network then was run through the permutation test where each feature had its values scrambled and the performance impact evaluated. The neural network's feature importance rankings reveal a slightly different perspective compared to the decision tree and random forest models. The most influential features for the neural network are energy (0.0413), danceability (0.0396), and valence (0.0315). Interestingly, features like acousticness and duration_ms, which were dominant in the decision tree and random forest, rank lower in the neural network's analysis. This suggests that the neural network captures different patterns and interactions between features.

The random forest and decision tree models emphasized acousticness, duration_ms, and speechiness as top predictors, which aligns less with the neural network's rankings. One possible reason for these differences is the limited number of permutations performed for the neural network, which may have restricted its ability to fully explore feature importance. We had limited time and compute resources so we were unable to take large permutations of the data. Because of this, we realized that it is much more costly in time and computation to accurately analyze feature importance and thus was not heavily considered in our project. Additionally, when training the neural network first, we saw that the accuracy was low, showing that it was not doing a good job of predicting popularity across the classes. This points to how difficult it is to choose a good architecture for a neural network for this task as well as the fact that analyzing feature importance for a poorly performing model may not be as helpful. Overall, we concluded that neural networks was not a practical option for feature importance with our dataset and wasn't considered heavily.