Final Report: Capstone Project Two
Predicting track popularity on Spotify

Springboard Data Science Career Track
Leo Evancie

The Problem:

Spotify hosts some 70 million songs and caters to over 300 million listeners worldwide. Each track has an associated popularity score, derived from listener engagement. So, Spotify can determine on a post hoc basis whether a given song performs well on the platform. But what about new music?

Using the data science method, can we use song characteristics like duration, lyrical explicitness -- even abstract concepts like “danceability” -- to predict whether a song will be popular? Or, to ask the question in a different way, can we glean any patterns that might give artists an understanding of which song characteristics typically associate with popularity?

The Approach:

First, I used Spotipy, a Python library built to interact with Spotify’s API, to scrape ten thousand tracks, mostly sampling from the past ten years.

Thanks to Spotify’s stellar data integrity efforts, there were vanishingly few issues with missing data. The data-cleaning process mainly involved some string analysis (e.g., converting the string representation of a list of countries where the track is available into the integer count of said countries), feature-engineering (e.g., creating a Boolean column indicating whether the song features a guest artist, based on whether the title included the string “feat.”), and scaling.

Exploratory data analysis helped me determine which variables might be pertinent to the question of track popularity. For example: track number, duration, and explicitness each showed some association with popularity, while tempo and modality did not. Ultimately, I narrowed down a list of seven features (details shown in notebooks and model summary) to feed into various modeling frameworks.

While Spotify reports popularity as an integer from 0 to 100, I observed a bimodal distribution, with a large proportion of tracks showing a popularity score of zero or close to it. As such, I decided to treat this as a classification problem. By reencoding popularity scores less than 50 as 0, and greater than or equal to 50 as 1, I turned the popularity column into a binary indicator. Fortunately, my dataset included roughly half of each classification, so I did not have to worry about class imbalance. This method addressed the original problem of predicting whether or not a given track will be popular; we do not need to try to predict the exact popularity score out of 100.

Being that this is a learning exercise as much as it is a skills demonstration, I decided to approaching the modeling step by not only comparing the predictive performance of various types of models, but also exploring how hyperparameter tuning affected each type of model, as well as comparing times required to fit models to the training data and generate predictions from the testing inputs.

I tested four different types of classification models: Logistic regression, K-nearest neighbors, random forest, and gradient boosting. I employed some hyperparameter tuning for each.

The Findings:

With a combination of strong f1 test score, lack of overfitting to training data, and quick prediction times, I judged the gradient boosting classifier as the best model to predict popularity. The model used the following features from the original dataset: single, danceability, explicit, collaboration, time signature, duration, and track number. Further detail may be found in the modeling notebook and the model summary document.

The set of features may provide clues for artists looking to optimize their content for Spotify popularity. Popular tracks tend to share the following characteristics within the artist's control: Singles, rather than tracks from an album; explicit lyrics; guest-artist feature; standard 4/4 time signature; roughly 200 seconds long; low track number. (The keen observer may note that we have essentially reverse-engineered pop music.)

Further Research:

Spotify's API offers a plethora of data endpoints, allowing for the collection of a wider selection of track characteristics that may be at least as effective in predicting popularity. For example, the track endpoint does not include track genre, but perhaps one could ascertain genre information from the album endpoint, and then merge on track ID. New features or not, there is also the potential for further feature-engineering than what I explored here.

While I performed some parameter tuning, this is always an area for further potential exploration. The naive Bayes approach could explore the vast parameter spaces of the ensemble models more intelligently.