December 13, 2021

Up Next: A comparison of modeling frameworks for developing a song recommendation system

Beck Addison

STA 325

1 Introduction

The popularity of music streaming services in recent years has driven out previous players in the market of music distribution to become the highest grossing distribution medium of any previous form in history. As of 2019, the global market for music streaming was valued at $20.9 billion dollars and is expected to grow at a 17.8% compound annual growth rate over the 2020-2027 period (Salzman, 2021). However, this rapid growth combined with widespread consumption has driven major platforms such as Spotify and Apple Music (to name a few) to compete in what has effectively become a commodity market, where sellers are in a race to the most narrow profit margins in order to outsell their many competitors. The strategy for several of these services, such as the aforementioned Apple Music but also smaller players including YouTube Music (owned by Google) and Amazon Music Unlimited, is to use the parent organization's other revenue streams to subsidize free tier benefits that entice users to purchase a subscription, often at a loss or near-loss to the service itself.

Spotify, on the other hand, has made solid if small profit on a large operating income, and represents a staggering 27 percent of recorded music revenues globally (Prey, 2020). Given that they operate independently of any parent company, Spotify has found an edge in their extensive reliance on recommendation algorithms, best illustrated by their development of an autogenerated "New Music Friday" playlist and also their on-the-fly "radio" generation button, where a user can generate a unique list of stylistically similar songs to any given song by simply clicking a button. While other services have now caught on to this trend, Spotify continues to pioneer creative methods to utilize user data effectively.

In this project, three methods are compared for effective classification of an individual's song preferences using a combined analysis of Spotify's "Audio Features", a set of quantified indicators collected by Spotify for every song in their database. These models will attempt to effectively predict which songs the user likes based in a similar way to Spotify's own recommendation system. These methods include a Random Forest model, a K Neighbors Classifier, and a Logistic Regression. Each method will be compared against the others for performance, and the best model will be chosen based on the

2 Data

All of the recommendation models will be trained on the songs available in the Spotify Song Attributes dataset, available on Kaggle. This dataset contains a total of 2,017 songs with the following features associated with each observation (Unless otherwise noted, descriptions sourced from the Spotify Developer Documentation):

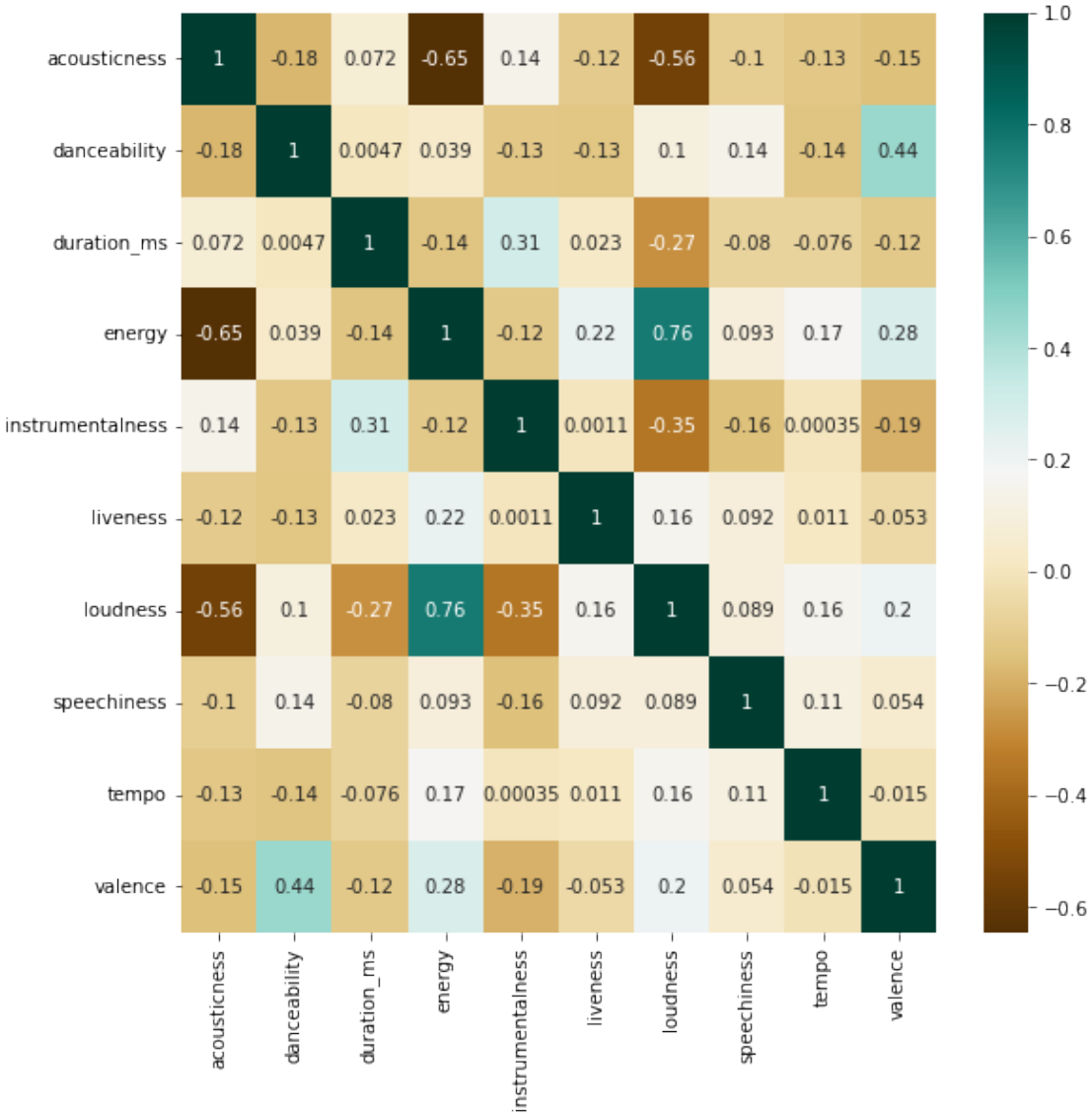
- **acousticness** *(a continuous float on `[0.0, 1.0]`)*: A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
- **danceability** *(a continuous float on `[0.0, 1.0]`)*: How suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
- **duration_ms** *(a continuous integer on [0, $+\infty$))*: The duration of the track in milliseconds.
- **energy** *(a continuous float on `[0.0, 1.0]`)*: A measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
- **instrumentalness** *(a continuous float on `[0.0, 1.0]`)*: 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. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
- **key (an ordinal integer on [-1, 11])**: The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C(sharp)/D(flat);, 2 = D, and so on. If no key was detected, the value is -1.
- **liveness** *(a continuous float on [0.0, 1.0])*: Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
- **loudness** *(a continuous float on [-60.0, 0.0])*: The overall loudness of a track in decibels (dB).
- **mode** *(a nominal integer in {0, 1})*: Indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
- **speechiness** *(a continuous float on [0.0, 1.0])*: Detects 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. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
- **tempo** *(a continuous float on [0, $+\infty$))*: The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
- **time_signature** *(an ordinal integer on [3, 7])*: An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of "3/4", to "7/4".
- **valence** *(a continuous float on [0.0, 1.0])*: The musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

Additional variables listed on this dataset not included in the Spotify Developer Documentation include:

- **song_title** *(a nominal string)*: the title of the song as listed on Spotify.
- **artist** *(a nominal string)*: The primary artist of the song as listed on Spotify.
- **target** (an ordinal integer in $\{0, 1\}$): A label placed on the data by the curator of the dataset. This value represents the response label for this project, and indicates whether he liked the song or not.

2.1 Exploratory Data Analysis

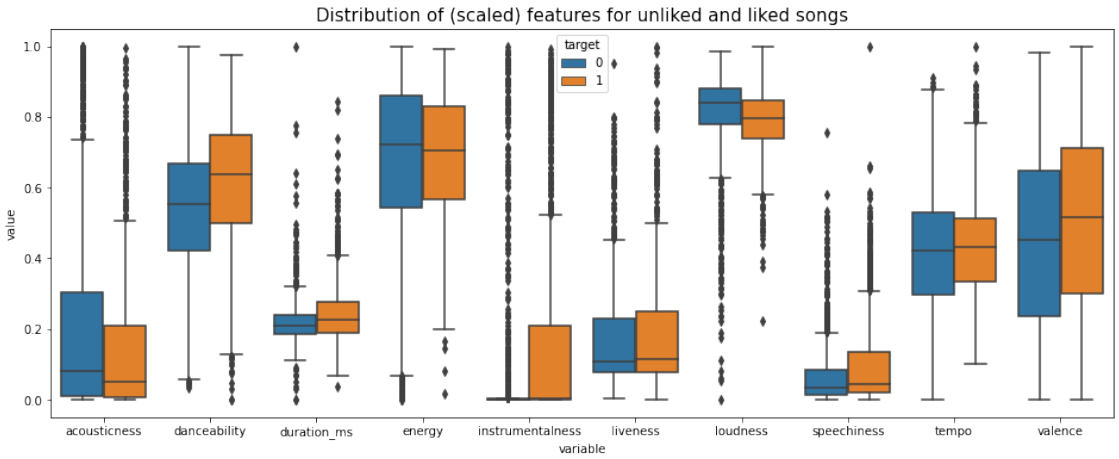
First and foremost, we can intuit that there is likely some amount of correlation between the song attributes. This is illustrated below:



A correlation plot of the song attributes as recorded by Spotify.

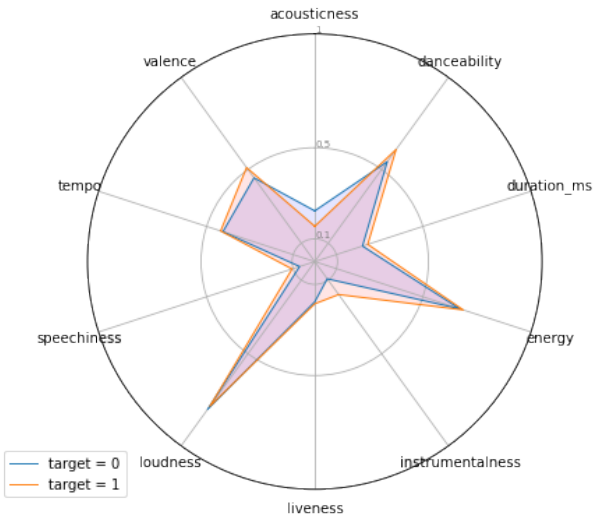
Some of the notable stand-outs here include energy and acousticness (a strong negative correlation) and energy and loudness (a strong positive correlation). This makes intuitive sense - an acoustic track is likely to be slower in tempo and energy and certainly not be very loud. In contrast, songs with high valence appear to have a similarly high energy and loudness, pointing to their upbeat, positive nature.

As mentioned previously, this user encoded a variable called `target` that is 1 to denote a liked song and 0 to denote a song that was not liked. We can see in the plot below how the liked songs compared to unliked songs when considering these song attributes:



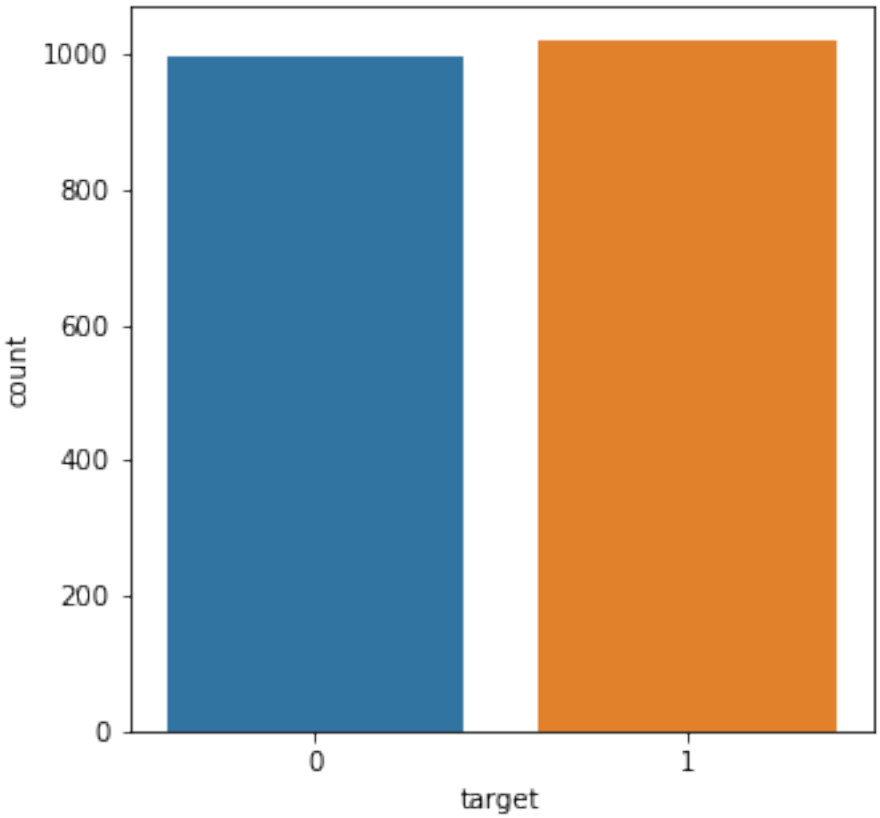
A set of boxplots showing the distribution of attribute values for songs the individual doesn't like (blue) and songs he does like (orange).

From above, it appears that this individual prefers songs that are typically not very acoustic but very danceable and with high valence, compared to unliked songs. Likewise, he appears to like songs with higher instrumentalness than in unliked songs. However, it should be noted that, for the most part, his preferences fall in line with the available music in the dataset. This could be comparable to someone who prefers very pop-centric, of-the-moment songs that are easy to dance to and are high energy. Another way of visualizing this is given below:



A radar plot showing the mean value for each of the song attributes of songs the individual doesn't like (blue) and songs he does like (orange).

Finally, we can see that the number of liked to unliked songs is roughly equal - the dataset is thankfully not imbalanced with respect to the response label:



A histogram of the target variable value counts. Since counts are similar, we can conclude the classes are balanced.

2.2 Encodings and Transformations

To prepare the data for processing, I passed it through the following encoders:

1. An ordinal encoder of the `mode`, `key`, and `time_signature` variables - this allows the modeling functions to see these variables as ordered categorical (ordinal) variables rather than as continuous variables.
2. Scaling of the numerical attributes - I scaled and centered the numerical song attributes using the `sklearn` package's `MinMaxScaler`. This was particularly important for variables such as `tempo` and `loudness` - otherwise their impact could be misunderstood by the model relative to other variables.
3. A one hot encoder of the `song_title` and `artist` variables, so that they could be properly examined by the various regression methods used.

In addition, the data was split into a test and training set, on a 30-70 test-train percentile split.

3 Methodology

The following regression methods were chosen to predict the outcome of `target`: A random forest classifier, A logistic regression model, and a k-nearest neighbors classifier. Grid search with 10-fold cross-validation was performed on each of the models' tuning parameters and scoring on Accuracy and ROC AUC (see the table below), and the best model of each of the model classes was chosen for comparison (see Results). These three algorithms were chosen based on their intuitiveness and simplicity, as well as their relevance to the context of the data itself.

To some degree, we have the underlying prior belief that there is some clustering that is occurring in the dataset - this person's tastes in music are likely to be consistent, and therefore he's more likely to prefer songs similar to other songs he's liked than he is to prefer songs he didn't like before. This is the underlying intuition of K-nearest neighbor models - that points around a given point are likely to be in the same class as that given point, compared to points further away. In addition, since the concept of "nearness" is very broad (Euclidean distance is traditional, but truly any type of measurement of closeness between two points works), it allows for a lot of flexibility and experimentation to perform a grid-search with. Finally, K-nearest neighbors is easy to understand - this is particularly important when conveying results to people who are not familiar with a more complicated modeling algorithm.

Like the KNN classifier, the random forest classifier is useful in large part because it is fairly easy to understand when the number of trees in the set are fairly low; while it is more complicated than simply picking the nearest points to a given point, an RF classifier is also very fast to compute and scales remarkably well to accomodate more data. In addition, RF classifiers like this are robust to outliers, and from the boxplot shown above, it appears that there may be a fair amount of outliers in both classes. One drawback is that it is not as interpretable at scale (i.e. as the number of trees grows); thus, while I expect this model to be more accurate than the KNN model, on the spectrum of interpretability established by these models, the RF is probably the least interpretable.

Finally, the logistic regression model was chosen to act as a deviation from the typical set of categorical classifiers. Since we are trying to provide the user with songs he will like, the idea behind this model, that we can give the odds he'll like it, is perfect to consider the problem from less binary perspective. The previous two models provide an answer that states "you will like the song" or "you won't like the song" based on the data, whereas the logistic model can answer the question of whether he will like the song with an answer that is more nuanced. From a technical perspective, the logistic regression model is also very easy to train and, if explained properly, has a very convenient tabular representation with coe icients that can help to explain the individual effects of the predictors on the models.

The Grid Search parameters used for the KNN and RF models were specified as follows:

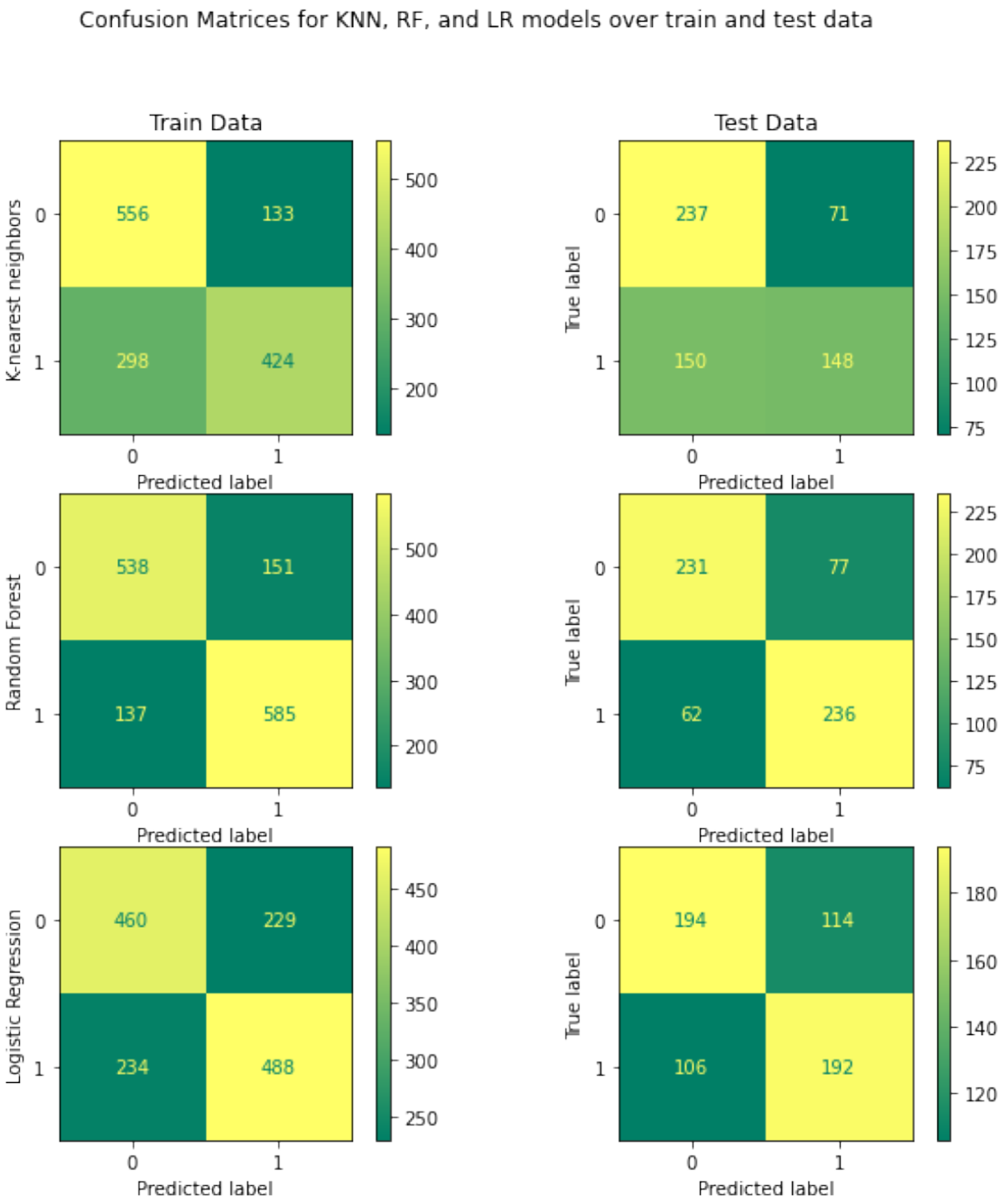
K-nearest neighbors Classifer:

- K: $[0, 1, \ldots, 15]$
- weights: 'distance', 'uniform'

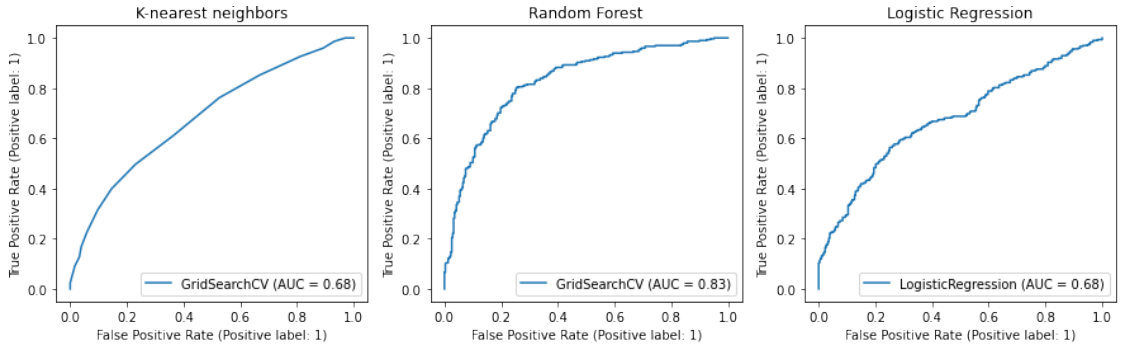
Random Forest Classifier:

- N (number of trees): $[64, 65, \ldots, 128]$. This range of trees seems arbitrary, but was educated by the explanation in (Baranauskas 2012) of an optimal tradeoff between ROC and runtime.
- Max Depth (the number of levels in the tree): $[1, 2, \ldots, 5]$
- Minimum Sample Split (how many samples required to split a branch/node into two children): $[2, 4, 6, \ldots, 20]$

4 Results



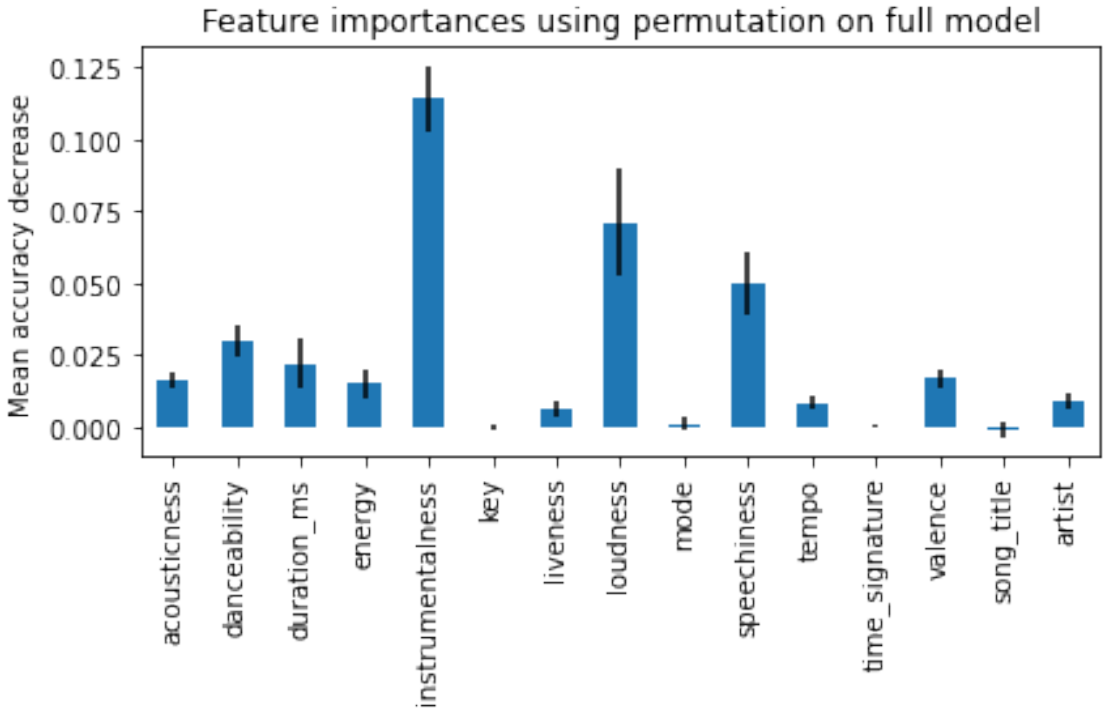
A set of confusion matrices detailing the number of true negatives (top left quadrant of each plot), false positives (top right quadrant), false negatives (bottom left quadrant), and true positives (bottom right quadrant). The first column shows the performance of the model(s) over the training data, while the second column shows performance over the test data.



A set of ROC curves, one for each of the models.

From the results above, we can see that our Random Forest model had the highest accuracy. In the first chart, we see the confusion matrices produced for both training data and testing data - notice that, for the K-nearest neighbor model, the rate of false negatives is much higher than the rate of false positives, meaning that the KNN model is conservative in its confirmation of what songs our subject likes; meanwhile, the logistic regression model has a fairly balanced false positive/false negative rate, with similar overall accurace. This is reinforced by the ROC plot above - with an AUC of 0.68, they can accurately predict this individual's song preferences about 68% of the time for both of the models - keep in mind, however, that the distribution of the types of wrong answers will be different between the two models, as explained previously. However, in the context of the problem, neither model is very impressive - both are only slightly better than flipping a coin to decide whether he likes the song or not. Regardless, both models fail to meet the performance of the Random Forest model, which correctly classifies the outcome 83% of the time. Let us take a deeper look at this model.

The first important thing to examine with this model is the feature importance of each of our variables of concern. This will help to elucidate what decisions, conscious or unconscious, this individual is making to choose whether he likes or doesn't like a song.



The feature importance table of the random forest model, taken using permutations on the full model to avoid high cardinality variables (categorical variables with many classes, i.e. the artist and song_title variables).

From this feature importance table (above), it appears that none of the variables are particularly powerful in increasing accuracy individually, but that among those features, the instrumentalness of the song is most impactful (even then, it was only responsible for a 0.100 to 0.125 mean reduction in accuracy). This comes at no surprise - in the boxplot displayed in the Data section, we saw that there was a notable difference in the instrumentalness of the songs he liked, versus the songs he didn't. Other important features included the speechiness and loudness of the song, while features like the key didn't make a noticeable difference in his decision making.

5 Conclusion

The results of the modeling procedure indicated that, for the construction of a recommender system using previously collected "liked" songs data, a random forest or otherwise tree-based process is probably best. There is a strong and valid argument that there exist multiple interaction terms, which forests are excellent at discovering, even if it isn't necessarily visible in the feature impor-tance analysis (Wright, Ziegler and Konig, 2016). This would naturally confuse a K-nearest neighbor model, which is looking for clear separations along the given parameter space (which didn't include interaction terms). In addition, one of the key drivers of song performance is the complex application of melodies, harmonies, and effective lyrics - concepts that, at least currently, are almost impractically di icult to boil down into a quantifiable value for a model to infer upon. In a study by Rentfrow, Goldberg and Levitin (2011), similar results were shown in predicting what songs subjects preferred based on the features of the song - only moderate correlations were established, but the researchers also noted that there were certain qualities of the music, notably social qualities, that weren't captured as part of the study. The same is true here, and thus we can only recognize that we have an imcomplete picture of a very complex psychology.

While this project demonstrated a moderately accurate random forest-based model for predicting song preference, this project serves as a more effective demonstration of the complexity of even simple concepts such as the decomposition of a song into its component parts. Further study into these concepts could work to reveal what drives human musical preference and even help us discover how we write songs to be preferable, opening up doors to the generation of likable music by machine learning or other algorithms from the field of artificial intelligence.

References

Prey, R. (2020). Locating Power in Platformization: Music Streaming Playlists and Curatorial Power. Social Media + Society. https://doi.org/10.1177/2056305120933291

Rentfrow, P. J., Goldberg, L. R., & Levitin, D. J. (2011). The structure of musical preferences: a five-factor model. Journal of personality and social psychology, 100(6), 1139–1157. https://doi.org/10.1037/a0022406

Salzman, A. (2021, October 27). Spotify has finally found a profitable tune. Barron's. Retrieved December 13, 2021, from https://www.barrons.com/articles/spotify-has-finally-found-a-profitable-tune-51635346820.

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

Wright, M.N., Ziegler, A. & König, I.R. Do little interactions get lost in dark random forests?. BMC Bioinformatics 17, 145 (2016). https://doi.org/10.1186/s12859-016-0995-8