Nick Seguljic

Final Report:
Spotify Popularity Prediction

The Problem

Spotify is a digital music service that gives access to millions of songs. It has an online API where any user can download musical metrics about any song it owns e.g. energy, valence, tempo. It is also known for putting out a weekly playlist for all users that is wildly successful in pinpointing what a user may like before they like it. Given this, I want to be able to use their online information and see how well I can predict the popularity of a song. My goal will be to predict attributes of a successful song from music with popularity greater than 80 and see if I can predict the popularity score of a song that does not have a popularity score assigned to it, using the musical metrics provided in the data.

Going forward, I will look to see if songs can be simplified down to numerical features and if these values can help to determine the popularity of a song. Being able to predict the popularity a song could be useful to anyone looking for new music or for advertising for music. For the purpose of this project, I will observe a given genre, break the songs into “popular” and “unpopular” music (a popular song will be defined as a song with a popularity feature >80) and analyze the data to see if any of the parameters stand out in terms of predictive power and appear to be unique and/or popular to a given genre.

Data Wrangling

After reading in the data using pandas.read_csv(), I observed the raw data head, and tail to look at the columns. The columns are the features that come from the Spotify API.

Next I analyzed the range of values of the features:

Feature	Data Type
genre	object
artist_name	object
track_name	object
track_id	object
popularity	int64
acousticness	float64
danceability	float64
duration_ms	int64
energy	float64
instrumentalness	float64
key	object
liveness	float64
loudness	float64
mode	object
speechiness	float64
tempo	float64
time_signature	object
valence	float64

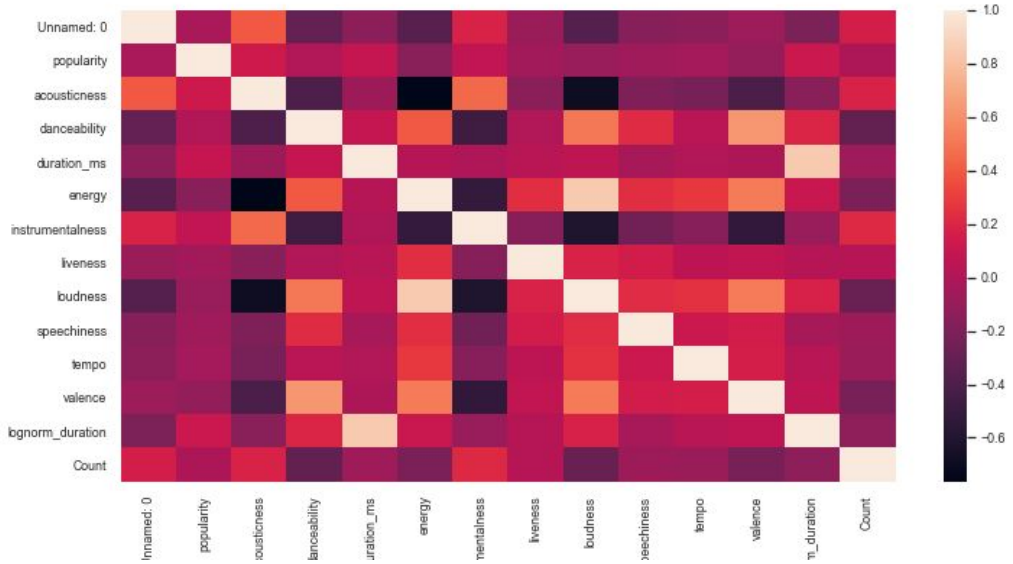
Next I added four features: the time bucket feature to determine if popularity is largely associated with a range of song times, a log transform of the time feature, a count of the number of songs that an artist has already made and put out on Spotify, and the feature popularity (a binary designation if a song has a popularity greater than 80 or not).

Lastly, Genres or styles that are not contemporary or commercially marketed in substantial numbers have been excluded from this list like “Children’s Music” and “Anime” as they are not represented as mainstream music most music platforms . In addition, both categories would probably have song features and signatures that are much too different from the conventional Rock and Pop.

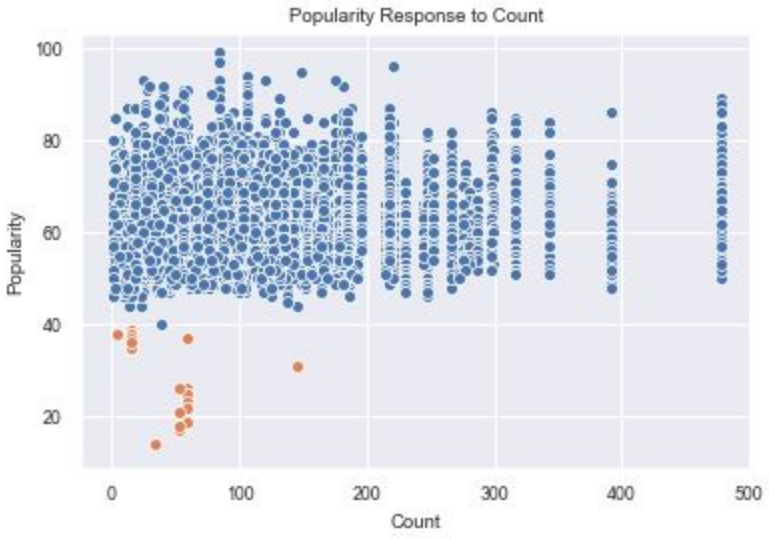
Exploratory Data Analysis

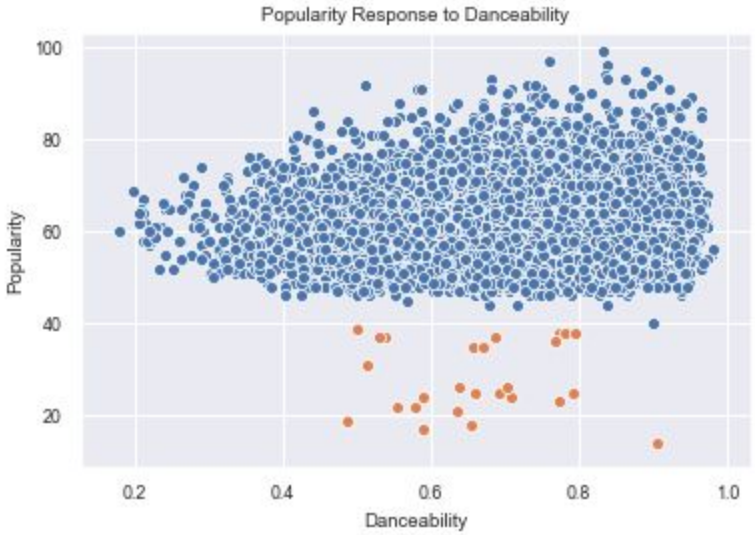
This notebook looked at how features trend with a song’s popularity score and sought to view how the average feature values vary between a popular song from that genre and an unpopular song from that genre. In addition, it observed how features vary from that genre to all music and looked to see if those differences were statistically significant.

The first item I looked at in this section for each genre was the correlations that each feature had with popularity; here it is for Rap:

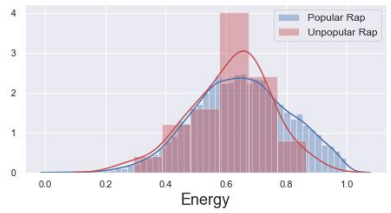


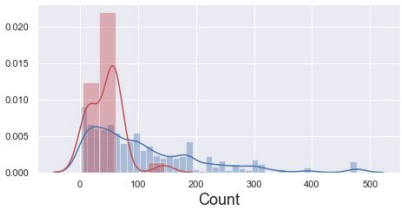
I then looked at each given genre's features with the highest correlation popularity. First creating scatter plots, here are examples from Rap music for Count and Danceability:

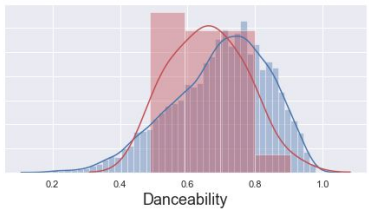


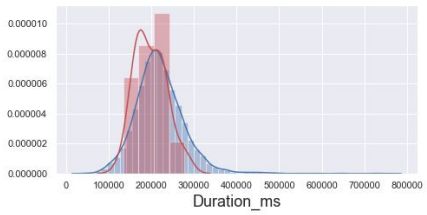


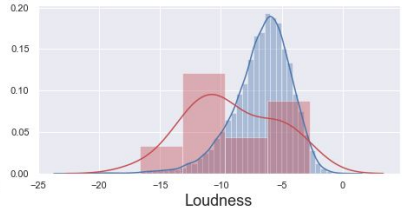
Here are the distribution plot examples from Rap, Pop, Dance, and Blues:

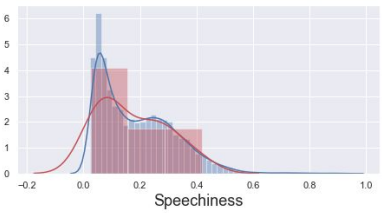

Popular Rap
Unpopular Rap
Energy

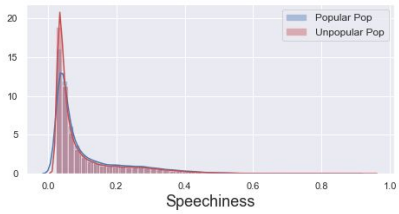

Count

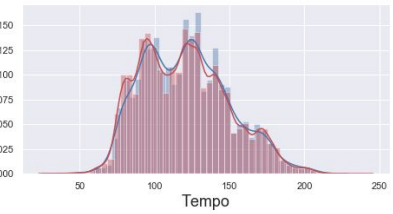

Danceability

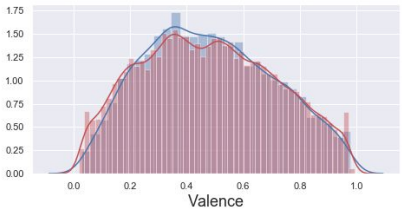

Duration_ms

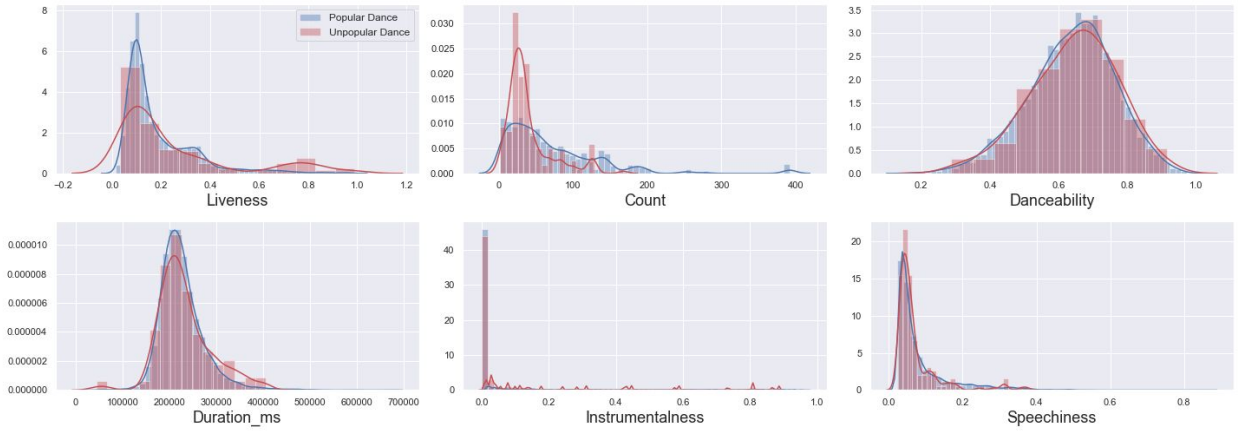

Loudness

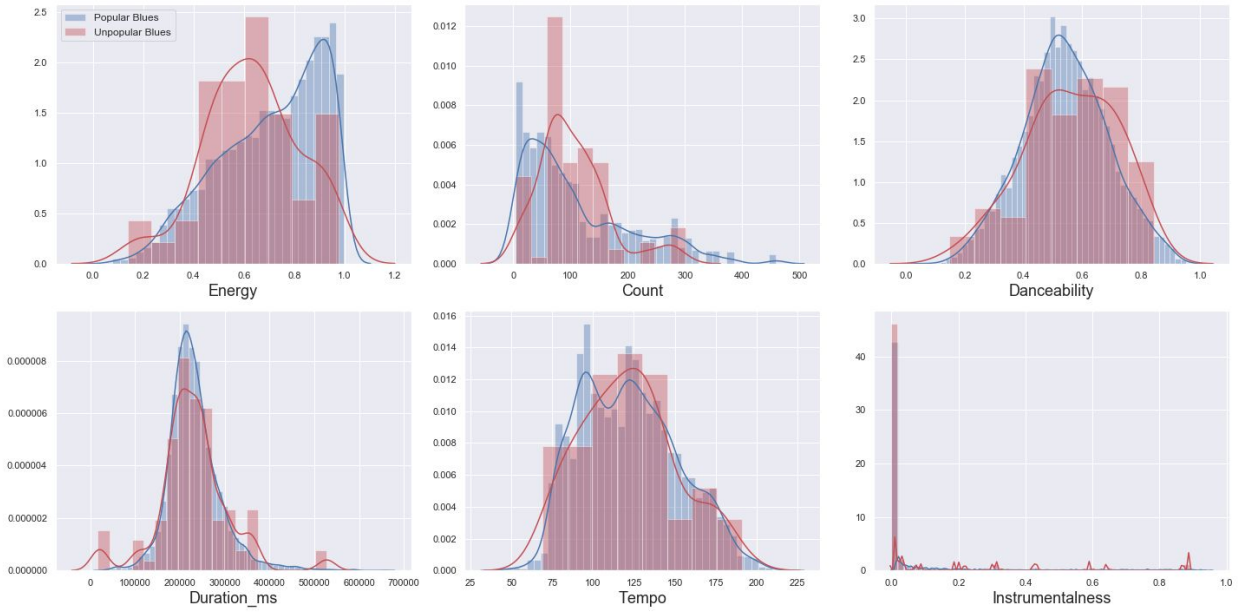

Speechiness


Popular Pop
Unpopular Pop
Speechiness


Tempo


Valence



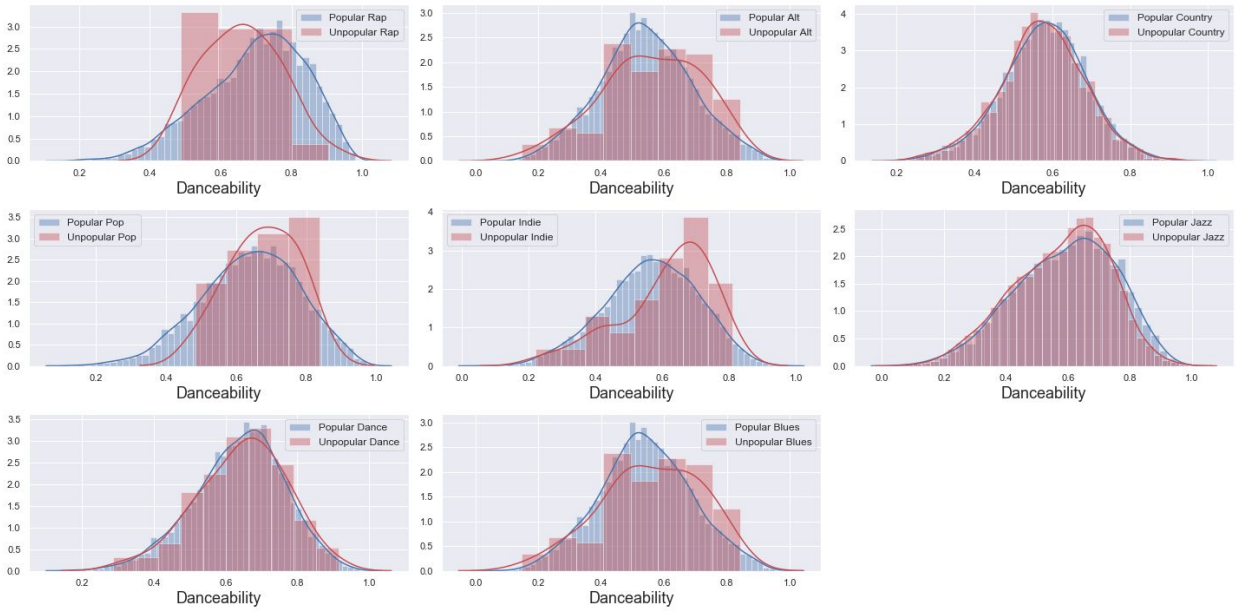


Like most genres, Rap had its largest distinction between popular and unpopular songs in the Danceability feature. The more popular a song is, the more danceable it tends to be. This trend was true for all genres except for Blues, which had a large feature distinction from count.

There do not appear to be too many differences between popular and unpopular Blues music. Popular blues has slightly more danceability on average, and a bimodal peak for tempo, that has roughly the same average for popular and unpopular music.

To conclude, I wanted to see if there were statistically significant differences in danceability in all genres.

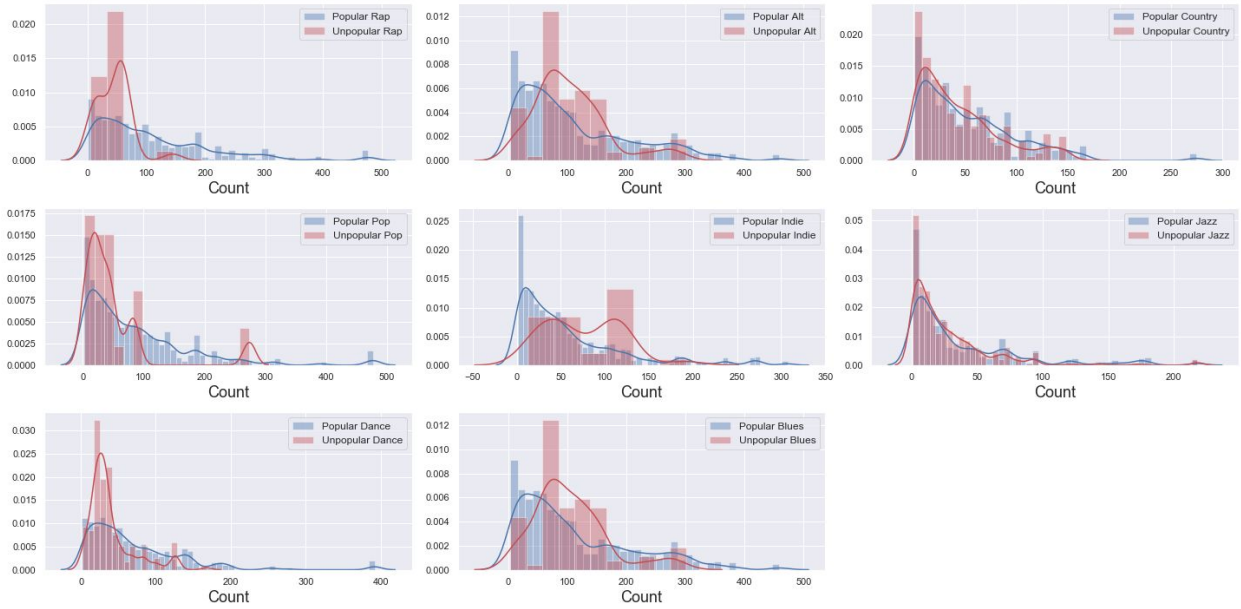
The null hypothesis (H_0) was that the danceability for each genre comes from the same dataset and that they should have the same mean and distribution of data. If the null hypothesis is rejected then it means they do not necessarily not come from the same dataset, proving this difference will be useful to distinguish popular music from a genre.



Genre	Statistic	p-value
Rap	-20.682	6.36E-93
Alternative	-74.657	0
Country	-74.657	0
Pop	-29.622	1.99E-184
Indie	-27.149	2.07E-156
Jazz	-100.249	0
Dance	-68.712	0
Blues	-41.185	0

We can reject all of the null hypotheses because they all have p-values less than .05, and say that the data from each genre popular versus unpopular do not not necessarily come from the same set.

One other feature I looked into to see if differences were statistically significant between popular and unpopular features was the count feature. It stood out for Blues and as mentioned before it was a better indicator of popularity than danceability. Here are the graphs for count and the t-test scores:



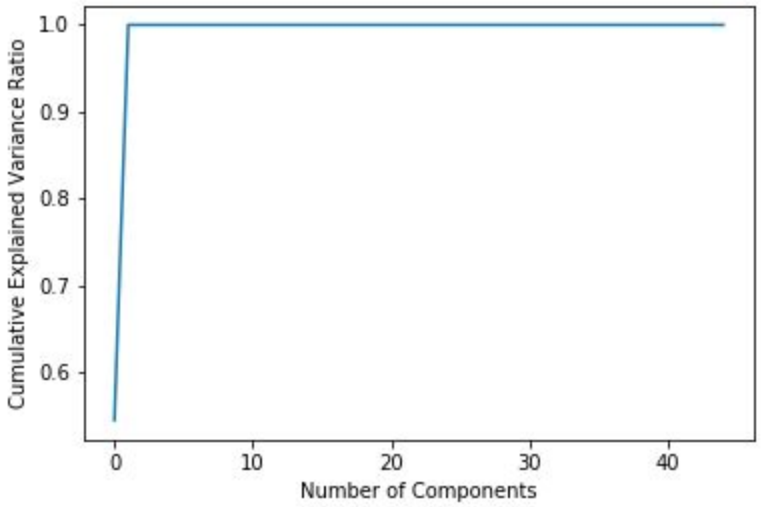
Genre	Statistic	p-value
Rap	-3.305	4.57E-01
Alternative	-0.745	0.457
Country	-9.007	2.56E-19
Pop	-2.133	3.29E-02
Indie	2.496	1.26E-02
Jazz	-12.164	8.66E-34
Dance	-5.717	1.12E-08
Blues	-0.745	0.457

From this we can reject the null hypothesis for Jazz, and unlike before we can accept the null hypothesis for Dance and Pop. This makes sense because there was a clean distinction for popular versus unpopular Jazz music, and the plots generated for Dance and Pop looked identical.

Model Selection

To begin I selected LinearRegression, RandomForestRegressor, KNeighborsClassifier, DecisionTreeClassifier, and AdaBoostClassifiers to see if I could predict the actual popularity of a song.

After using get_dummies to work with qualitative features, using train_test_split and scaling the data using StandardScaler. I also wanted to see if a Principal Component Analysis(PCA) could be used to reduce the number of features. Here is the plot of the number of components versus the cumulative explained variance. I did not transform the data with any of this information because the number of components it wanted to reduce to is 1 which I believed would be too few for a proper analysis, however later on I will go into what features mattered the most for the models to possibly shed light on why the PCA wanted to reduce down to one component:



I was able to look at the models and compare accuracy, MAE, MSE, and RMSE:

Model	Accuracy	MAE	MSE	RMSE
AdaBoostClassifier	0.000068	44.764981	2149.432	46.36197
KNeighborsClassifier	0.000271	36.673023	1462.29	38.2399
DecisionTreeClassifier	0.016491	16.878181	409.6733	20.24039
RandomForestClassifier	1	0	0	0
LinearRegression	NaN	16.245047	423.5996	20.58153

Linear Regression cannot have an accuracy associated with it which is why I looked at the three other variables. RandomForest probably had a perfect accuracy because of overfitting, KNeigbors had an accuracy of zero.

Going forward, I had the lowest error for every metric with the Linear Regression Model so I tuned the hyperparameters according to this:

I redid the model and got:

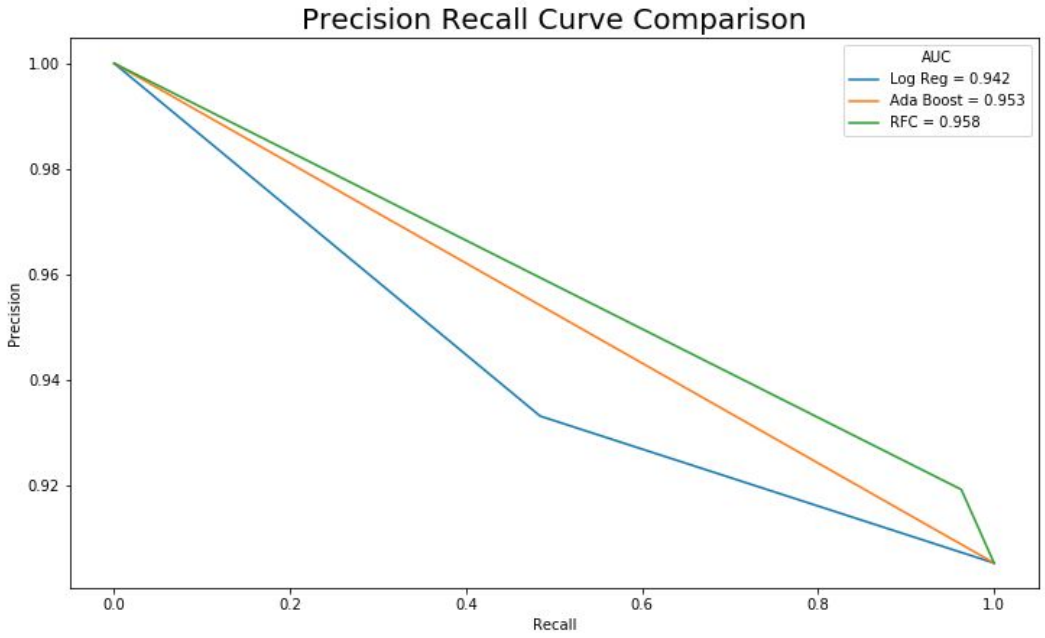
```
Mean Absolute Error: 6.237420071514329
Mean Squared Error: 71.77087073161633
Root Mean Squared Error: 8.471769043807576
Mean cross validation test score: 0.955077746550893
Mean cross validation train score: 0.9644744855320806
```

These numbers were not at all what I wanted so I wanted to see if this could be redone. Instead of predicting the popularity score, I wanted to see if just if a song is popular could be predicted.

I used the same models but swapped Logistic Regression for Linear Regression since I was looking for a binary output:

Model	Accuracy	MAE	MSE	RMSE
RandomForestClassifier	0.899355	1.635222	0.101052	0.317887
LogisticRegression	0.489243	123.9856	0.510757	0.714672
AdaBoostClassifier	0.247302	0.097251	0.097251	0.311852
KNeighborsClassifier	0.097251	230.2009	0.902749	0.950131
DecisionTreeClassifier	0.097251	230.2009	0.902749	0.950131

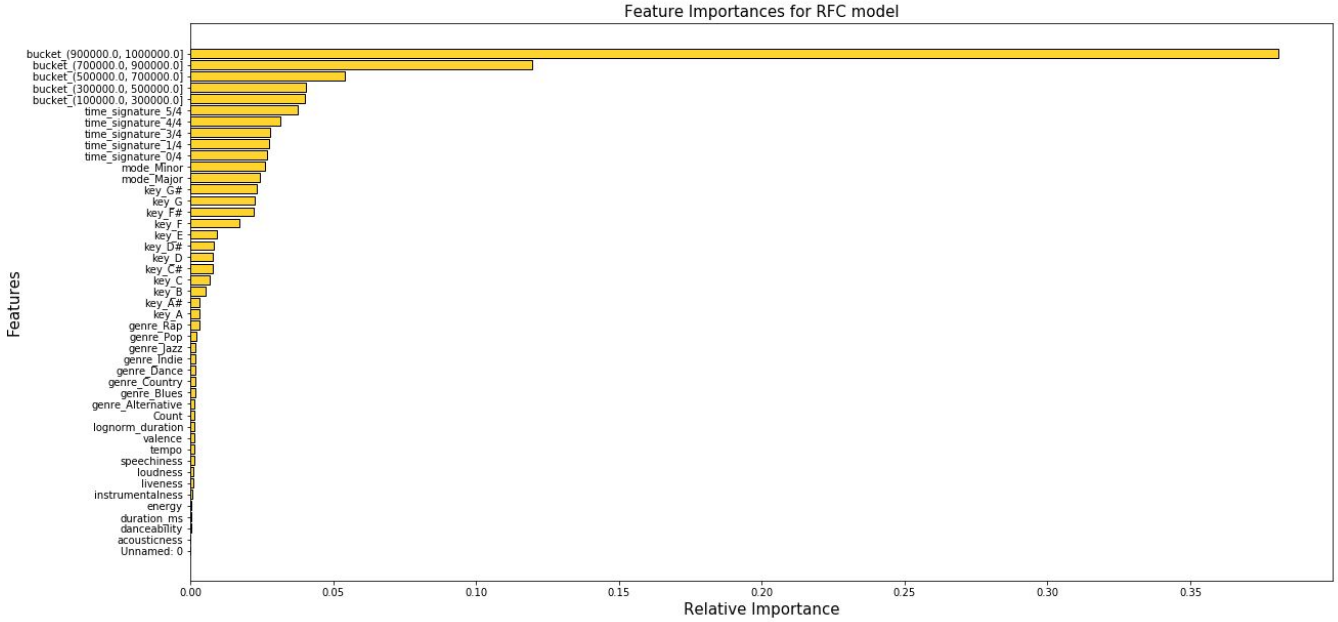
Before doing hyperparameter tuning I wanted to look at the precision recall curves for the top three performing models. Precision is the ratio of the number of true positives to the sum of the true positives and false positives. It describes how good a model is at predicting the positive class. Looking at the curve above reveals that I should move forward with hyperparameter tuning of the Random Forest Classifier, because it produces a lower number of true positives as well as false positives. The curve can be seen here:



Then just as before I tuned the hyperparameters of the best model, for this I chose Random Forest and got an accuracy of 0.903.

Major Findings

· For all genres except for Blues, the more danceable a song is the more likely it is to be popular.
· Trying to predict popularity alone was best done with the Linear Regression model which had an MAE 0f 6.23, and an RMSE of 8.47, much better than all other models.
· Trying to predict if a song will be popular was best predicted by Random Forest or not has a 90% chance of being correct and had an MAE of 1.64 and RMSE of 0.32.
· The features that contributed most to the popularity of a song for the Random Forest model are the time buckets and can be seen here:



· As mentioned from the PCA, the analysis wanted to reduce down to one feature. I think that the time buckets had such a strong influence on making a song popular or not, e.g. if a song is longer than 10 minutes the chances of it being listened to and popular were so low that the model thought it the best way to predict an unpopular song, leading to an analysis saying that it only needed that feature in order to reject the song being popular.
· Going forward knowing the importance of these features may prove useful for further model development. As you can see features like acousticness add almost no value, whereas features like the time buckets seem to contribute a lot toward popularity. This graph doesn't add up to what the PCA was conveying, that one feature explains most of the variance, so my choice to not move forward with it and reduce features was most likely right.

Recommendations

If a person or company wanted to get insight into if a song will be popular or not and they can properly define the number for popularity that they wish to exceed, a model can be developed with at least 90% accuracy using songs of all types of genres to help make this prediction.

As it stands now, I do not think there is a straightforward quantitative way for a record label to look into this, so using these models or similar models could prove useful to them.

Further Research

Future work to better increase the predictive power of these models could be to look into different features and/or feature reduction to see if these features or lack of features could better contribute to a model. It is possible that the designation of popularity at 80 is too arbitrary and changing to a different number could yield models that have higher predictive power. Finally, training and testing data on models that look exclusively at one genre could lead to less noise in the data and to better predictive power.
