Spotify Hit Predictor Dataset

By: Daniel Garza, Andrew Nguyen, Rahman Khandakar, Devin Smith

GitHub Repository:
https://github.com/AndrewN2001/3337_Project/tree/main

About the Data

The dataset we will be using is from kaggle and uses data from Spotify to determine if a track is a hit or not.

About the Data Continued

- Track - Track's name
- Artist - Artist's name
- Uri - Identifier
- Danceability - Value from 0 - 1 determining how suitable the track is for dancing
- Energy - 0 - 1 representing a perceptual measure of activity
- Key - Estimated overall key of the track
- Loudness - Average decibels of the entire track -60 to 0 db
- Mode - Modality of track
- Speechiness - Presence of spoken words. Between 0 and 1 denoting how much is just words.
- Acousticness - Confidence measure from 0.0 to 1.0 if a track is acoustic
- Intrumentalness - Prediction if track contains no vocals. 0 - 1
- Liveness - Possible presence of an audience. 0 - 1
- Valence - 0 - 1 describing positive vibes
- Tempo - Overall beats per minute
- Duration_ms - Duration of track in milliseconds
- Time_signature - How many beats per bar
- Chorus_hit - Estimate of when chorus starts in track
- Sections - The number of sections
- Target - Whether a track is a hit(1) or a flop(0) based on if the track was in the Billboard's Hot-100 in its decade.

Problem Statement

The problem is to predict whether a song is a hit or a flop based on the given dataset of songs from the 1960s-2010s. The ‘target’ column will serve as the label, where 1 will represent a hit and 0 represents a flop. A song is a hit if it has featured in the weekly list (Issued by Billboards) of Hot-100 tracks in that decade. The goal of this project is to build a data science model that can accurately predict the target label for new songs based on the provided features.

Significance

Finding out if a track can be a hit or miss can help record labels identify who among new upcoming artists to invest in. Models created on the data on many tracks can help predict if the tracks new artists are making have a high likelihood of producing a hit. If the model shows an artist has potential, then it is a safer investment on the record label, and the new artist can be found and supported.

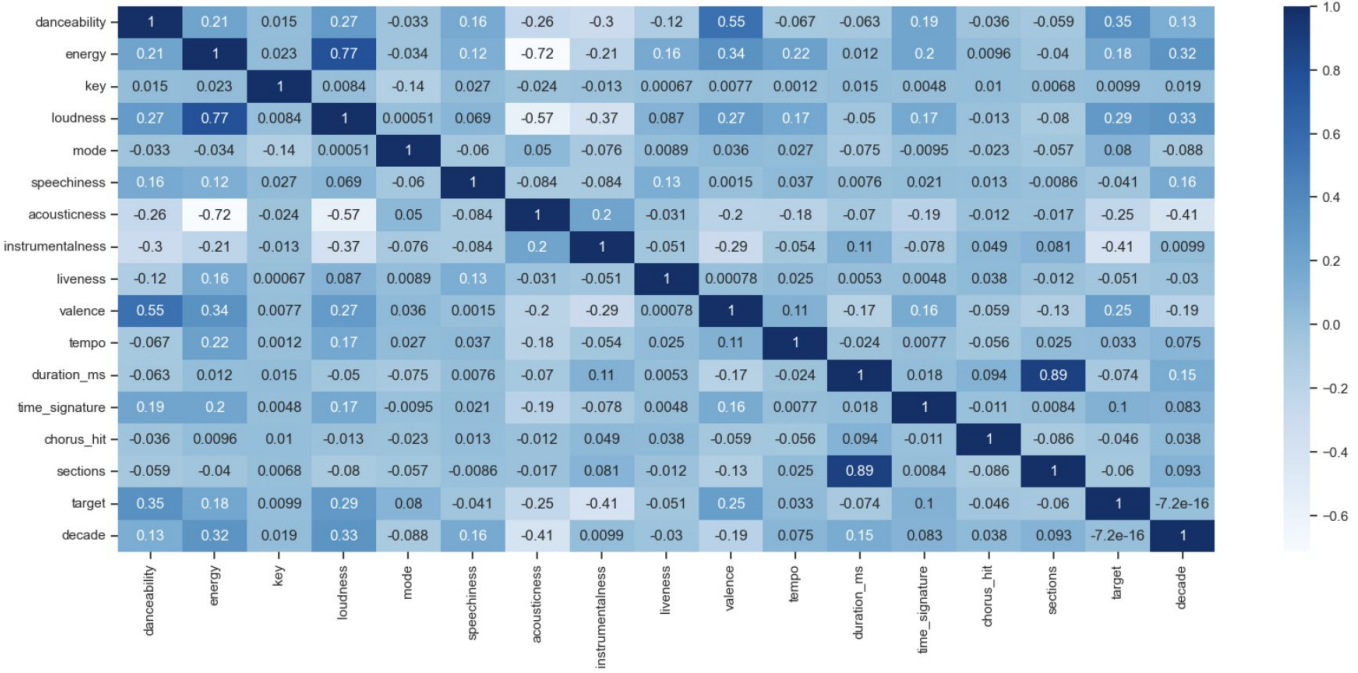
Challenges

- Feature Selection and Engineering: This dataset contains various features related to the songs, such as the audio characteristics, the release year, and artist information. With this large of a dataset, we will need to figure out which features are most important for predicting the target label to ensure maximum model performance.
- Model Selection and Evaluation: Choosing an appropriate selection of models for this classification task is crucial. Models, such as logistic regression, SVM, decision trees, and random forests, must be considered to see which model has the best performance. Evaluation using metrics such as accuracy, precision, recall, and F1-Score must be factored in our results.

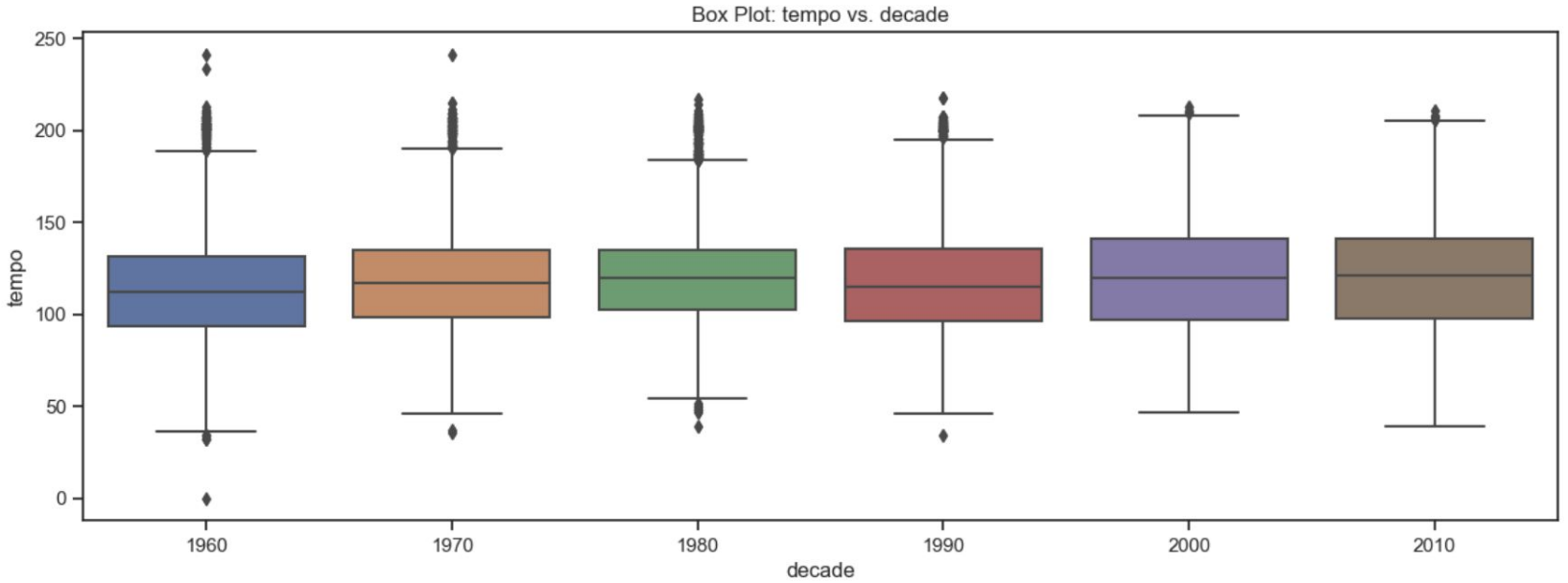
Approach

- Our approach to solving the problem involves the implementation of multiple machine learning models, including Logistic Regression, Support Vector Machines (SVM), Decision Trees, and Random Forests. These models were selected based on their ability to handle large datasets and being suited for classification problems.
- We begin by displaying important plots that pertain to the problem at hand. Afterwards we proceeded with preprocessing procedures, such as data cleaning, handling missing values if needed, and feature engineering to prep the dataset for modeling. Then, we split the dataset into training and testing subsets using train-test-split.

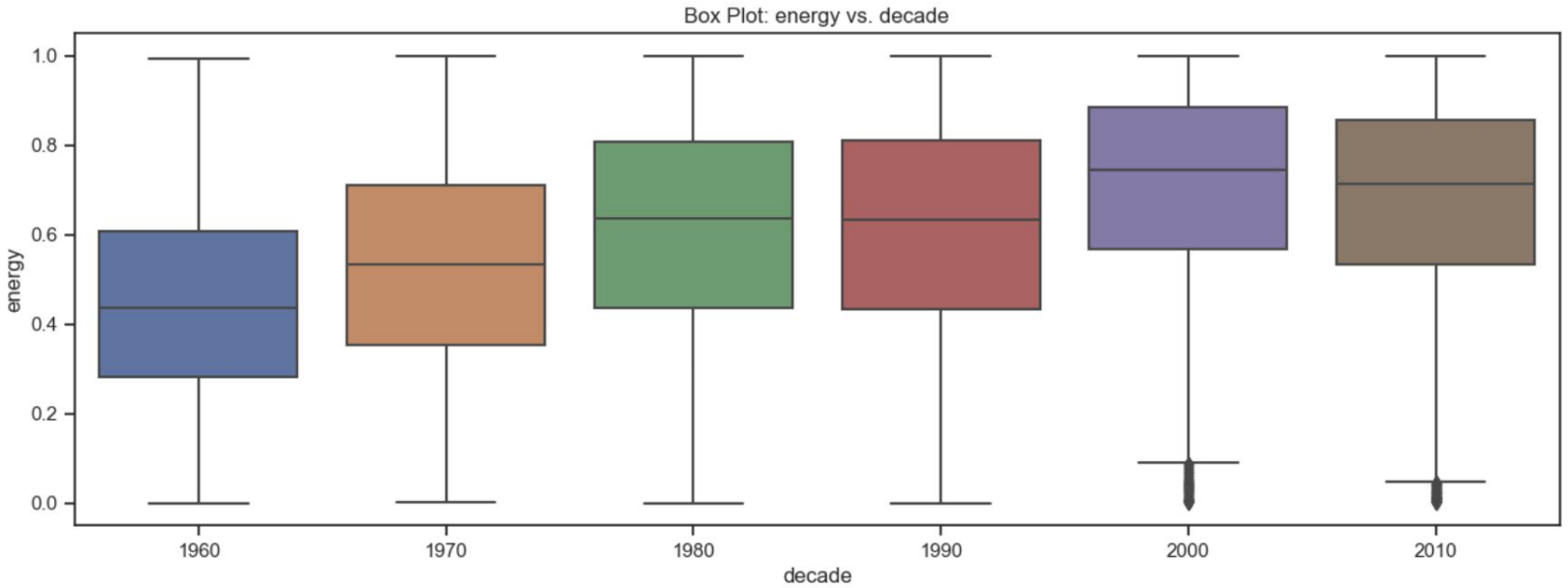
Data Exploration



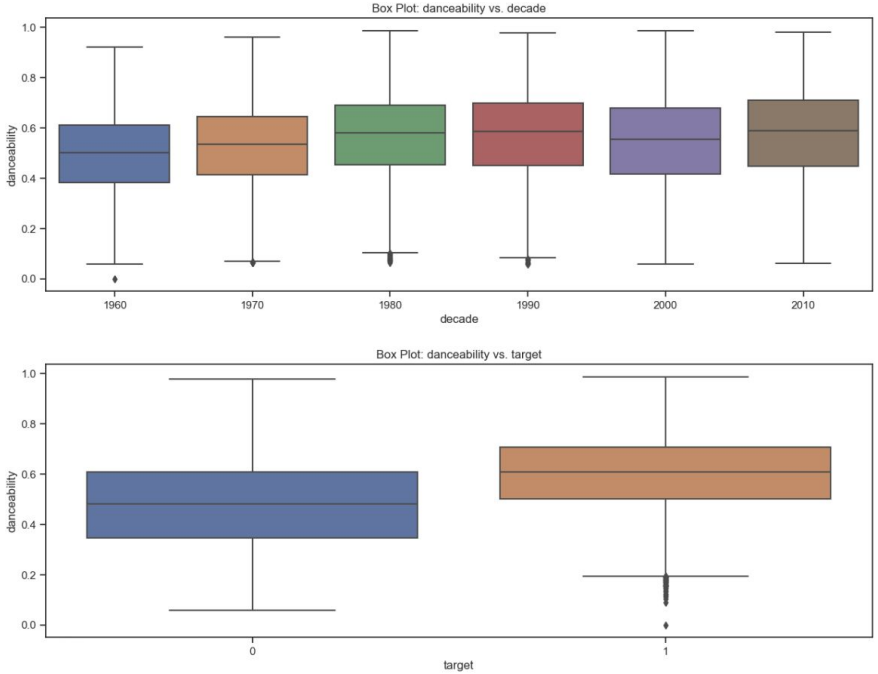
Data Exploration



Data Exploration



Data Exploration



Logistic Regression

Our problem is to find a binary classification of if a track is a hit or miss given many attributes of the task. A simple model to start is logistic regression for its efficiency handling large datasets and interpretability of the results.

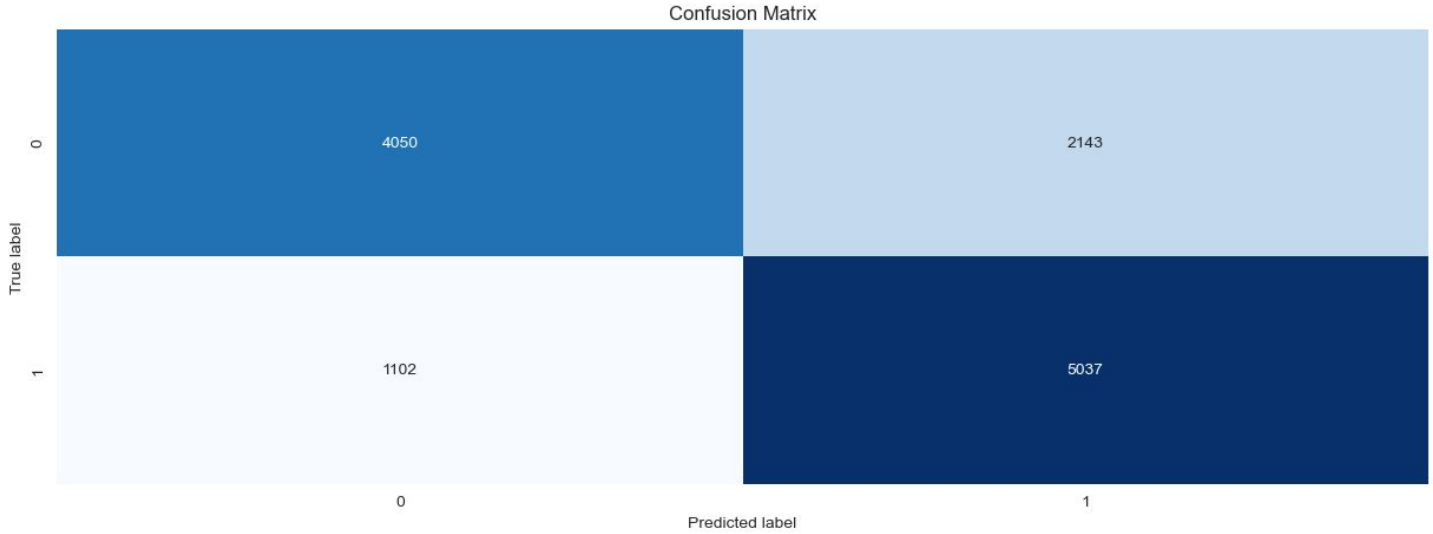
The biggest issue initially was that without any preprocessing, the model was barely better at predicting if a track was a hit than by guessing at random.

After scaling the data, the accuracy found after cross-validation was found to be 0.74.

This shows that some features were dominating in influence within the model because of how large in magnitude their numbers were. One place could have been the duration of the tracks that were represented in milliseconds.

	Attribute	Coefficient	Odds Ratio
0	danceability	0.701168	2.016107
1	energy	-0.507253	0.602148
2	key	0.041760	1.042645
3	loudness	0.744224	2.104808
4	mode	0.175983	1.192418
5	speechiness	-0.237275	0.788774
6	acousticness	-0.617724	0.539170
7	instrumentalness	-1.023384	0.359377
8	liveness	-0.041552	0.959299
9	valence	-0.068807	0.933507
10	tempo	0.098162	1.103141
11	duration_ms	0.021029	1.021252
12	time_signature	0.069471	1.071941
13	chorus_hit	-0.052519	0.948836
14	sections	-0.084236	0.919214
15	decade	-0.428611	0.651413

The danceability and loudness each multiply the probability that a track will be by about 2 when increasing by one unit while all else remains equal. Acousticness multiplies the probability that a track is a hit by about 0.5 which is the feature with the most demerit when trying to find if a track is a hit.



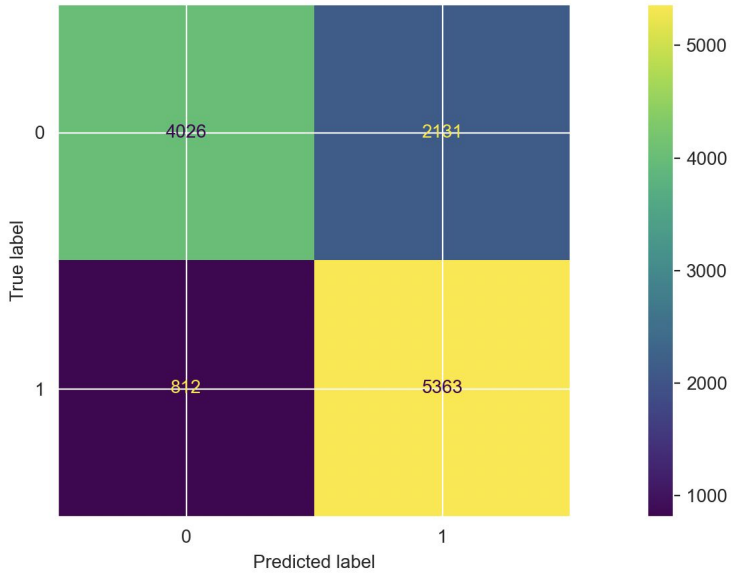
	precision	recall	f1-score	support
0	0.79	0.65	0.71	6193
1	0.70	0.82	0.76	6139
accuracy			0.74	12332
macro avg	0.74	0.74	0.74	12332
weighted avg	0.74	0.74	0.74	12332

Support Vector Machines (SVM)

- This model was chosen not only because of its exceptional ability and effectiveness to perform with datasets with a large number of features, but also because of the ability to distinguish important features to be used in training as well as handling imbalanced data in which we predicted that the dataset would have (in which case, it did not.)
- The main challenge with this model was that since this dataset had over 40000 entries, the model will be computationally taxing and time-consuming, as well as the increase in memory requirement. For this, I decided to use the PCA technique to reduce the number of features, but retaining the most important ones, simplifying the model and improving its performance and computational efficiency.
- After using PCA to reduce the dimensionality of the dataset and using GridSearchCV for hypertuning, the best result for this model would be 0.77.

Support Vector Machine (SVM) cont.

	precision	recall	f1-score	support
0	0.83	0.66	0.74	6158
1	0.72	0.87	0.79	6174
accuracy			0.77	12332
macro avg	0.78	0.77	0.76	12332
weighted avg	0.78	0.77	0.76	12332



Decision Tree

- We decided to test the Decision Tree model because they require no preprocessing of the data, they make no assumptions about the distribution of data and they handles collinearity efficiently.
- The best results for the decision tree were seen when using gini criterion, max depth of 7, no scaling and no principal component analysis.
- A disadvantage of this model is that as the tree gets larger it can be prone to overfitting and dependent on pruning techniques. It can also lose valuable information while handling continuous variables.
- When testing with this model the highest test score we could achieve was 0.76

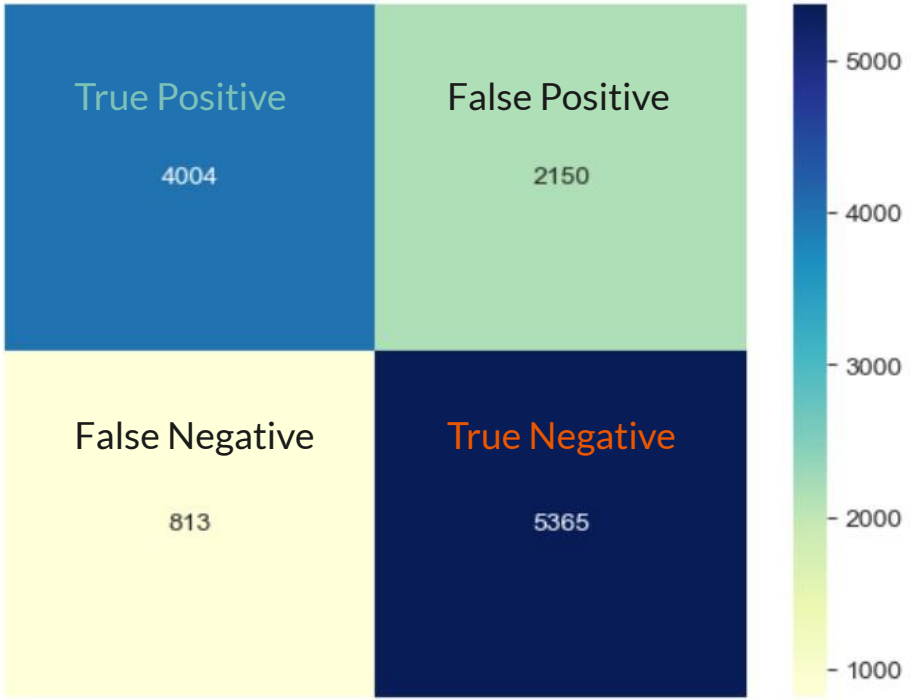
Decision Tree Stats

```
Model accuracy score with criterion gini index: 0.7597
```

```
[[4004 2150]
 [ 813 5365]]
              precision    recall  f1-score   support

           0       0.83      0.65      0.73      6154
           1       0.71      0.87      0.78      6178

    accuracy                           0.76     12332
   macro avg       0.77      0.76      0.76     12332
weighted avg       0.77      0.76      0.76     12332
```



Random Forest Algorithm

- With such a large dataset, it is important that we account for noise which is why we chose to use Random Forest as one of our models. There are many advantages when using random forest, particularly in a binary classification problem. A huge advantage of random forest is that it is robust to outliers and missing values. Other benefits are that it is easy to implement and is computationally efficient even for large datasets.
- Knowing that Random Forest is robust to outliers and has little overfitting compared to individual decision trees, we expected random forest to have a high accuracy score. Random forest received an accuracy score of 0.81, the highest of all our test data. This was not surprising based on the advantages mentioned before.

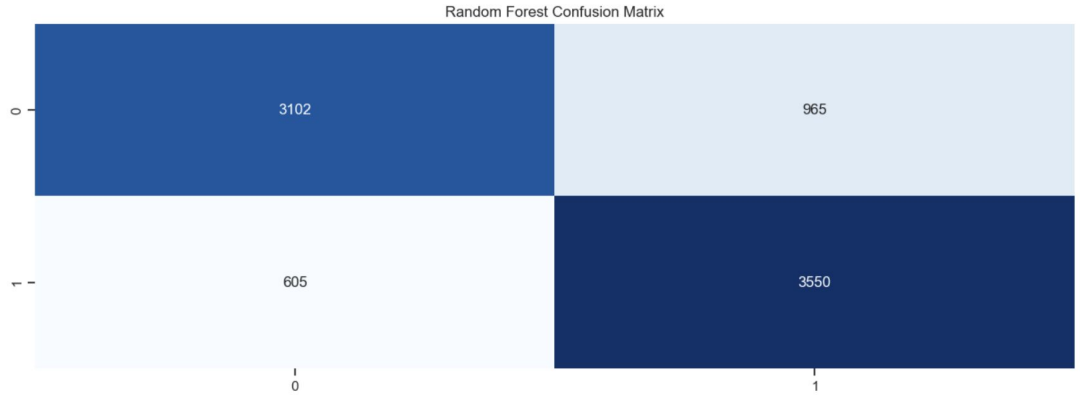
Random Forest Results

```
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.76      0.80      4067
           1       0.79      0.85      0.82      4155

    accuracy                           0.81      8222
   macro avg       0.81      0.81      0.81      8222
weighted avg       0.81      0.81      0.81      8222

Confusion Matrix:
[[3102  965]
 [ 605 3550]]
```



Conclusion

- SVM: 0.77
- Logistic Regression: 0.74
- Decision Tree: 0.76
- Random Forest: 0.81

When predicting a hit record, random forest would be the best model to use. With its' ability to handle high-dimensional data, effectively address imbalance classes, provide insights to feature importance, and deliver the highest performance, it would be the most suitable choice for this task. Record labels could use this model when deciding which new artists to invest in and be 81% confident of their results.