Predicting Billboard Hit Songs

By Group 3:
Thomas DeNezza
Syed Sabeeh Hassany
Jackson Winslow
Jason Wu

CSCI3387
Dr. Jose Bento Ayres Pereira
May 5th, 2022

Introduction: (Sabeeh)

“Them Changes” by Thundercat was a completely unknown song and unknown artist until November 13th 2017. Suddenly, overnight the song became a viral sensation with TikTok users using the song in their videos over 1 billion times as of today. The song was instantly propelled to the the Billboard top 50 songs and became recognizable as a “hit” song. Overnight, the song and the artist were propelled into mainstream Music media and even went on to win a Grammy Music Award after toiling in the music industry for over 5 years.

Nowadays, a song's popularity is often determined by its trendy use on social media platforms, predominantly TikTok and Instagram. Here, users can take songs and overlay them on videos or skits they make of themselves and share their videos to millions around the world. This is great because many underground musicians can have their “big break” in an unconventional way. However, this also often results in songs gaining popularity in very unpredictable and unmeasurable ways. Because of this, it's more important now than ever for artists to know how their songs will perform.

Problem: (Jackson)

In the music industry, it's not uncommon for artists to release songs that become overnight sensations, changing their careers in the process. If artists could know which qualities of their songs determine popularity they can optimize their performance and tune them for popularity. Logically we can figure that there must be a pattern behind what makes these songs rocket in popularity. The ability to identify this pattern and predict hits with accuracy prior to their release or exposure could prove useful for both business and personal reasons.

Solution: (Jackson)

To address this issue, we aim to develop a model that can predict whether or not a song will be a hit within the genres of Rap, Pop, Country, and Jazz. This binary prediction will be determined by models trained on the performance of songs associated with quantified values of their audio features. In order to define what constitutes a hit in training, we will use the USA Billboard Hot 100 list for the respective genre of each song. We aim to produce two separate analyses:

1. Evaluating our ability to predict whether a song was a hit or not individually across the four genres as well as all together (and which models perform best)
2. Evaluating which features are most influential in a song becoming a hit for each genre

Having access to these analyses can provide interested parties with valuable information as to which genres are easiest to predict a hit within as well as which characteristics of a song are most impactful to a song’s success.

Approach Overview: (Jackson)

We approached this project in a three step process:

1. Data generation
    a. Curating our own dataset by combining information from Spotify and Billboard API's
2. Model creation and augmentation
    a. Training base models on a binary classification, hypertuning these models to achieve best prediction accuracy
3. Analysis
    a. Analyzing our results in the context of each of our evaluations

Sourcing and Preparing the Dataset: (Thomas & Jackson)

While there were a variety of reputable and widely-used datasets available online, we opted to construct our own to ensure our models were of high quality to produce the most accurate results. For each genre, we begin this process by fetching 1,000 songs from the Spotify API from within a given year starting at 2022 and working back to 2012 (thus totaling in 10000 songs per genre). The data we saved from the API consisted of the song name, artist, as well as a variety of audio features that quantify different characteristics of the song. To make this data compatible with data from the Billboard API, each song was standardized so that no featured artist names were included in the song title and only the first listed artist on the song was saved.

We then fetched every Hot 100 list for the given genre and year from the Billboard API, applying the same song standardization process to make each of our sets of retrieved songs compatible. We then compared the songs fetched from Spotify to those fetched from Billboard, creating a new feature in our dataset named "hits" and assigned a "1" when the song was included in a Billboard list and "0" if not.

After repeating this process for each year we included in our dataset, we were left with a dataset consisting of 10,000 songs, each of which had the following features:

1. Name - Track name (not used as a feature for bias, used as sudo join for Billboard API and Spotify API).
2. Artist - Artist name (not used as a feature for bias, used as sudo join for Billboard API and Spotify API).
3. Genre: A feature specific to the mixed genre dataset that represents a specific genre. Genre is represented by an integer from 0-3 (0 → rap, 1 → country, 2 → pop, and 3 → jazz).
4. Danceability - A confidence measure of how danceable a track is based on tempo, rhythm, beat strength and beat regularity. A 0.0 represents low danceability and 1.0 represents high danceability.

5. Energy - A measure from 0.0 to 1.0 that captures intensity and activity of a song. Dynamics, loudness, timbre, onset rate, and general entropy contribute to this feature.
6. Key - The key the track is in, represented using standard pitch notation, ranges from -1 to 11.
7. Loudness (dB) - The average loudness of a track measured in decibels. Values are typically in the range of -60 to 0 dB.
8. Mode - The modality of a track (major or minor). Major is represented by 1 and minor is represented by 0.
9. Speechiness - A measure of spoken words in a track. Values range from 0.0 to 1.0, with values above 0.66 describing tracking that are probably made entirely of spoken words, values between 0.33 and 0.66 describe tracks that may include both spoken words and music, both sectioned and layered (this includes rap). Values below 0.33 most likely represent music without speech included.
10. Acousticness - A confidence measure on whether the track is acoustic or not. A 0.0 represents low confidence and a 1.0 represents high confidence.
11. Instrumentalness - A measure of whether or not there are vocals present in the track. A rating approaching 1.0 indicates low likelihood of vocals while ratings near 0.0 indicate that vocals are present in the track.
12. Liveness - A measure of whether a track seems to be performed live or not. A 1.0 predicts the track is performed live where a liveness rating of 0.0 indicates that the track is unlikely to be live.
13. Valence - A measure from 0.0 to 1.0 describing positivity conveyed by the track. A 1.0 describes a more positive track, while low valence describes a more negative track.
14. Tempo (BPM) - The overall estimated tempo of a track measured in beats per minute.
15. Duration (ms) - Duration of the track in milliseconds.
16. Hit - A binary value based on whether a given track has been on Billboard Hot 100. A 0 value represents false and a 1 represents true.

It is important to mention that the data is not evenly distributed with 50% of the songs being hits and 50% not being hits. Generally, we had the majority of the songs in our dataset not being hits, where roughly 16.4% of the songs were classified as hits. This is important to note because this metric will inflate our accuracy when we test our models. Thus, it becomes very important to carefully analyze the results of our models specifically, the F1 score and false positives which is a good indicator of our model quality. Below is the specific breakdown of hits to non-hits in percentage per genre:

1. Rap: 22.69% hits, 77.31% non-hits
2. Jazz: 3.21% hits, 96.79% non-hits
3. Pop: 16.17% hits, 83.83% non-hits
4. Country: 23.51% hits, 76.47% non-hits
5. Mixed: 16.39% hits, 83.61% non-hits

Picking Models: (Jackson)

After referencing past attempts at predicting whether or not a song would be a hit from a variety of other projects, we gathered a list of classical classifiers that we expected to perform well in this context. We continued to add more relevant classifiers as we went in our process. We focused on classifiers that are optimized for binary classification i.e. a hit or not a hit. This, mixed with a few models we wanted to experiment with that we covered in class this semester, made up our pool of models with a 70/30 train-test split. Our list consisted of:

1. Logistic Regression
2. K-Nearest Neighbors
3. Decision Tree
4. Random Forest
5. Support Vector Machine (Linear Kernel)
6. Support Vector Machine (RBF Kernel)
7. Gradient Boost
8. AdaBoost (uses Random Forest as base estimator)

We supplemented these eight models with a feedforward neural network that goes from 12 input features to the first hidden layer of 128 neurons, followed by two additional hidden layers of 256 neurons. All intermediate layers were simple ReLU layers. The output layer at the end is a sigmoid layer to optimize for binary classification. We used a simple binary cross-entropy loss function and used Adam optimization over gradient descent or SGD. This was because of Adam optimizers advantage of working well with sparse gradients and larger amounts of parameters. The binary accuracy and precision metrics were used specifically to work with avoiding false positives and false negatives.

An 80/20 train-test split was used specifically for the neural network and the model was trained over 50 epochs.

Tuning Classic Models: (Jackson and Sabeeh)

After running each of our base models through a simple grid search of hyperparameter tuning, we picked the top 3 performing models and further tuned them in order to achieve maximal accuracy. These models were run through a more thorough grid search that checked ~20 estimators, ~10 learning rates, and ~10 maximum depths where relevant in the models. We tuned a model for each genre's dataset individually, as well as for a dataset of all the genres mixed. This resulted in the following hyperparameters:

Rap:

- Random Forest
  - n_estimators=600
  - max_depth=None
- Gradient Boost
  - n_estimators=200
  - learning_rate=0.5
  - max_depth=10
- AdaBoost
  - base_estimator= RandomForestClassifier(n_estimators=200, max_depth=15)
  - n_estimators=100
  - learning_rate=0.5

Jazz:

- Random Forest
  - n_estimators=200
  - max_depth=15
- Gradient Boost
  - n_estimators=100
  - learning_rate=0.1
  - max_depth=3
- AdaBoost
  - base_estimator= RandomForestClassifier(n_estimators=200, max_depth=15)
  - n_estimators=100
  - learning_rate=0.5

Pop:

- Random Forest
  - n_estimators=50
  - max_depth=15
- Gradient Boost
  - n_estimators=100
  - learning_rate=0.1
  - max_depth=3
- AdaBoost
  - base_estimator= RandomForestClassifier(n_estimators=100, max_depth=7)
  - n_estimators=100
  - learning_rate=0.5

Country:

- Random Forest
  - n_estimators=100
  - max_depth=7
- Gradient Boost
  - n_estimators=100
  - learning_rate=0.1
  - max_depth=3
- AdaBoost
  - base_estimator= RandomForestClassifier(n_estimators=100, max_depth=7)
  - n_estimators=100
  - learning_rate=0.5

Mixed Genres:

- Random Forest
  - n_estimators=100
  - max_depth=15
- Gradient Boost
  - n_estimators=100
  - learning_rate=0.1
  - max_depth=3
- AdaBoost
  - base_estimator= RandomForestClassifier(n_estimators=100, max_depth=7)
  - n_estimators=100
  - learning_rate=0.5

Classic Model Analysis by Genre: (Thomas & Jackson)

Rap:

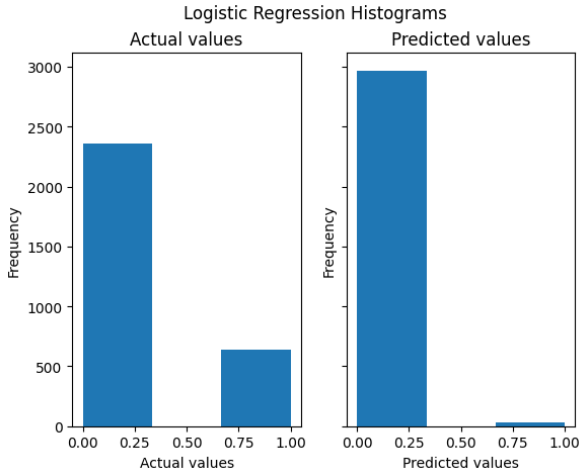
The models that stood out based on prediction accuracy were:

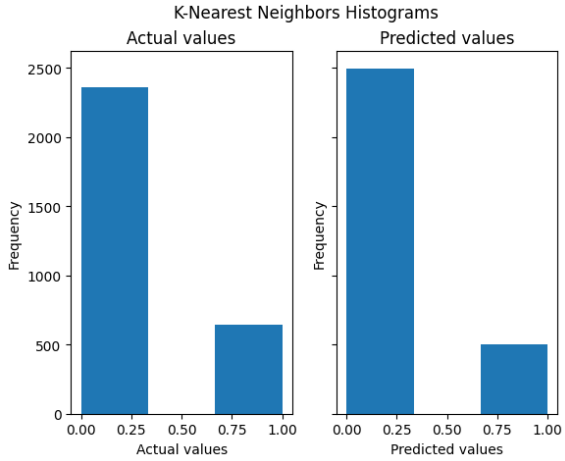
1. 91.43% - AdaBoost
2. 90.83% - Random Forest
3. 90.50% - Gradient Boost
4. 86.43% - Decision Tree

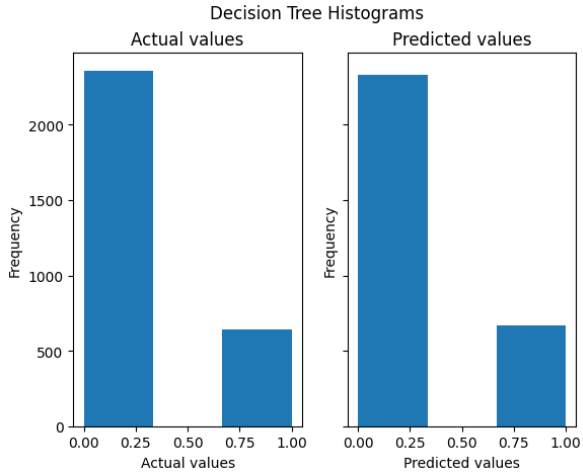
When ranking these models based on F1 score, we see the following:

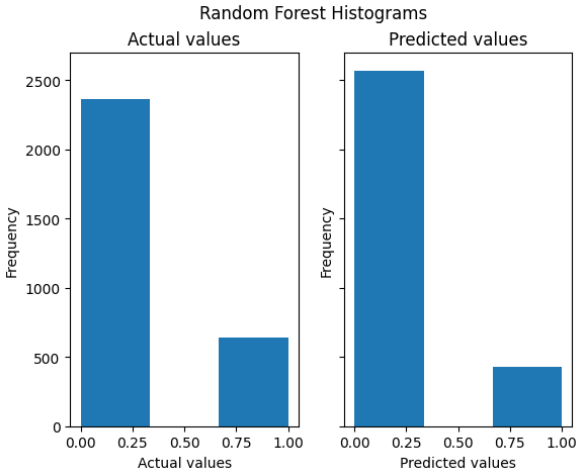
1. 75.87% - AdaBoost
2. 74.35% - Gradient Boost
3. 74.32% - Random Forest
4. 68.91% - Decision Tree

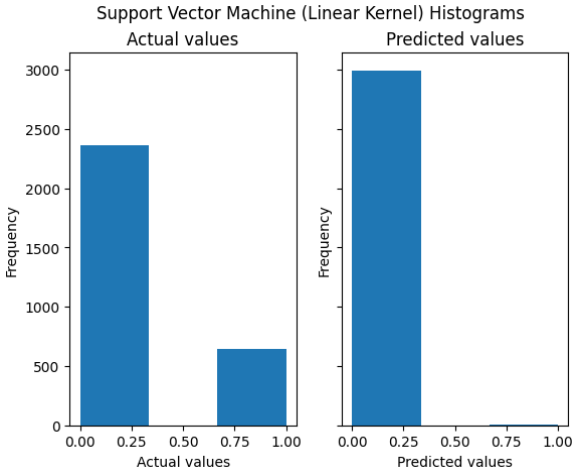
AdaBoost, Gradient Boost, and Random Forest all come out as clear leaders in terms of overall model performance. The F1 score shows that these three models are relatively of much higher quality in terms of correctly classifying songs as hits or not hits, however they are only decent in terms of overall quality for future use.

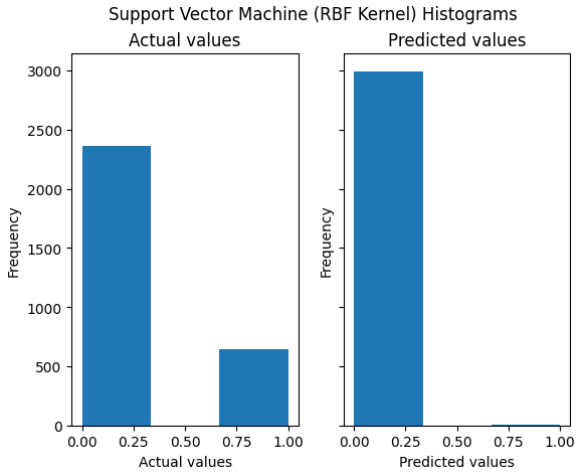


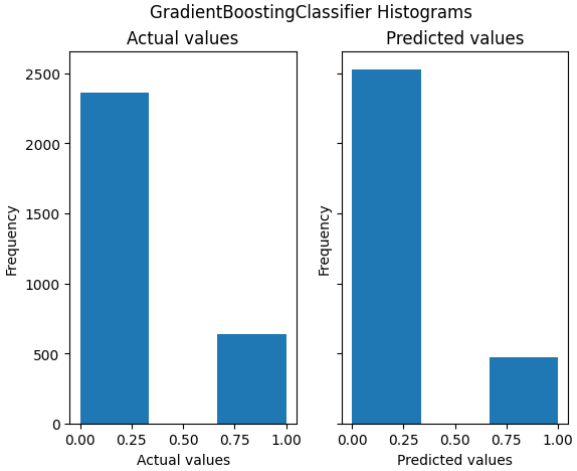


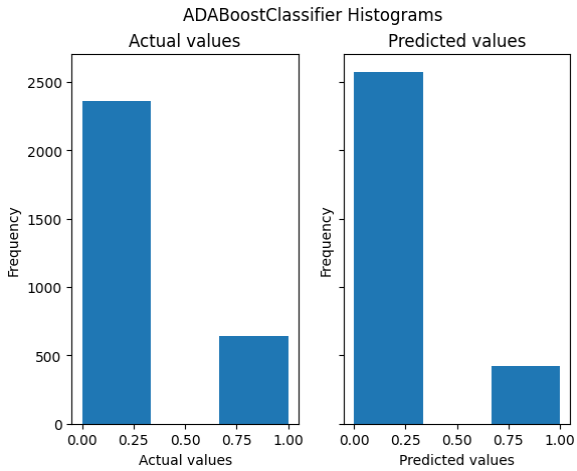





Support Vector Machine (Linear Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


Support Vector Machine (RBF Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


GradientBoostingClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values

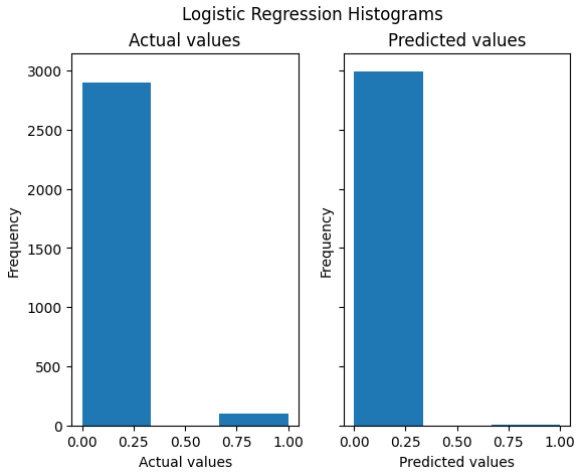

ADABoostClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values

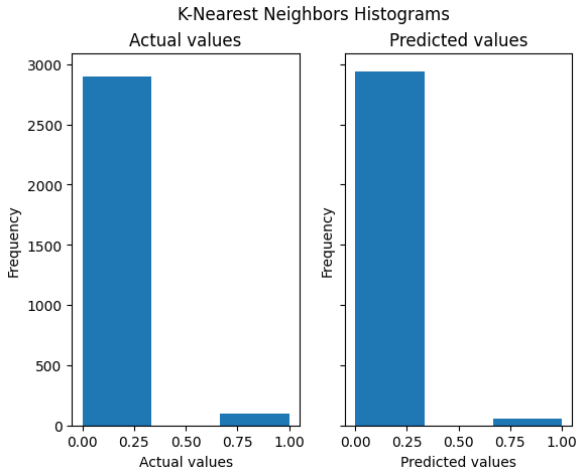
Jazz:

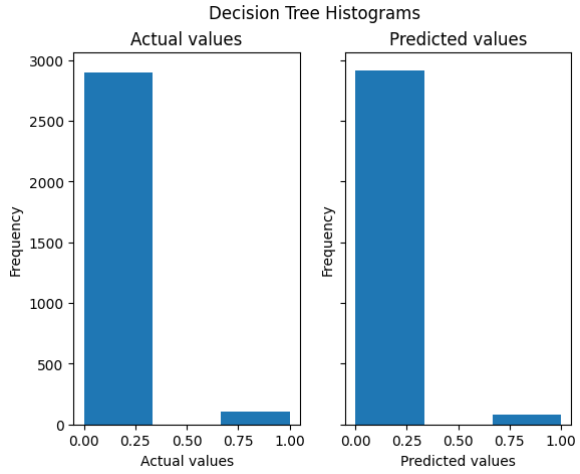
All eight of the models that we trained had a prediction accuracy within the range of 96.20% to 97.53%. However, only the following models stood out in terms of F1 score:

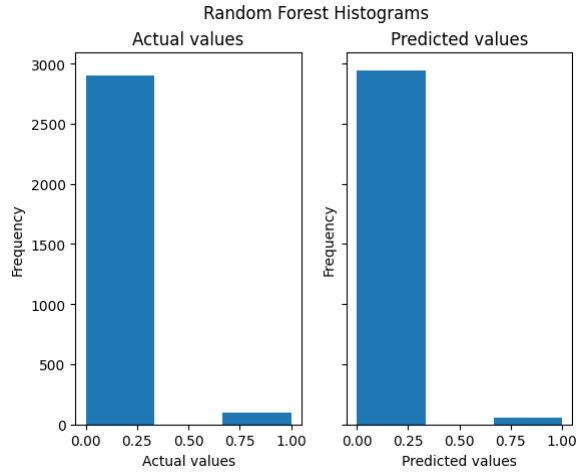
1. 52.56% - Random Forest
2. 51.89% - Decision Tree
3. 50.98% - AdaBoost

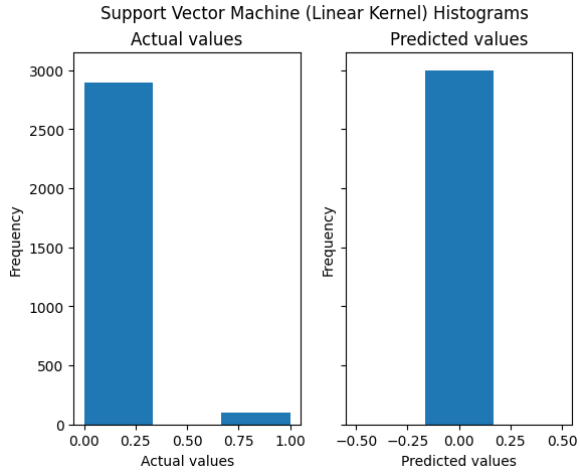
While these models were of relatively higher quality, their overall quality is still terrible given a nearly 50% F1 score for each of them. In reflection we attribute this to the highly skewed distribution of positive versus negative labels in our dataset.

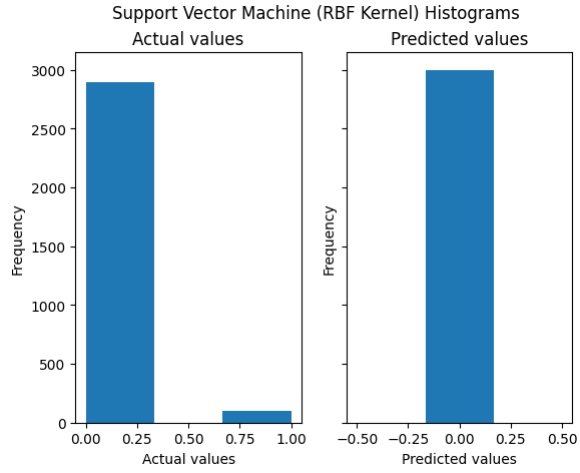


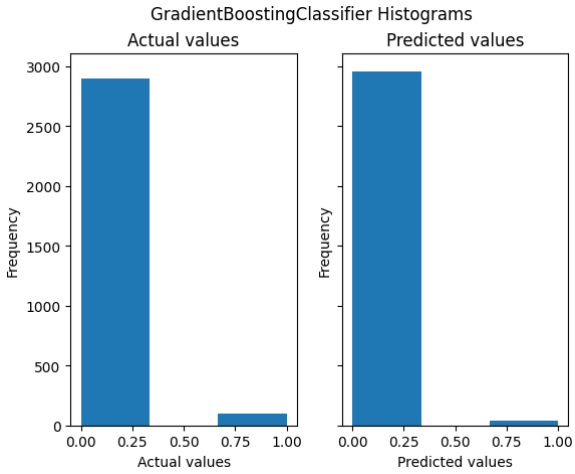


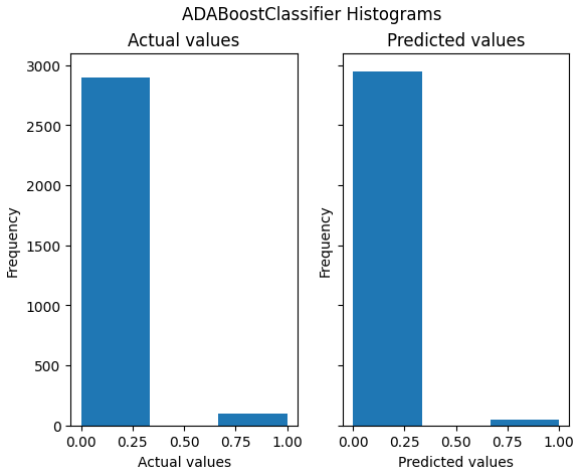









GradientBoostingClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values
0
500
1000
1500
2000
2500
3000
0.00
0.25
0.50
0.75
1.00


ADABoostClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values
0
500
1000
1500
2000
2500
3000
0.00
0.25
0.50
0.75
1.00

Country:

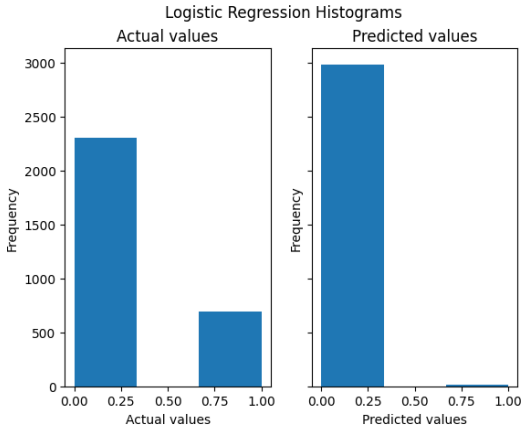
The models that performed the best based on prediction accuracy were:

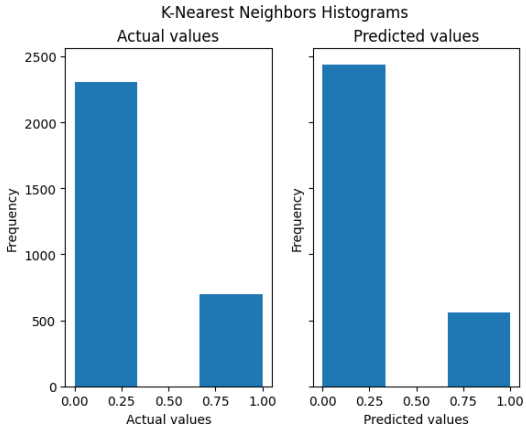
1. 90.17% - AdaBoost
2. 85.53% - Decision Tree

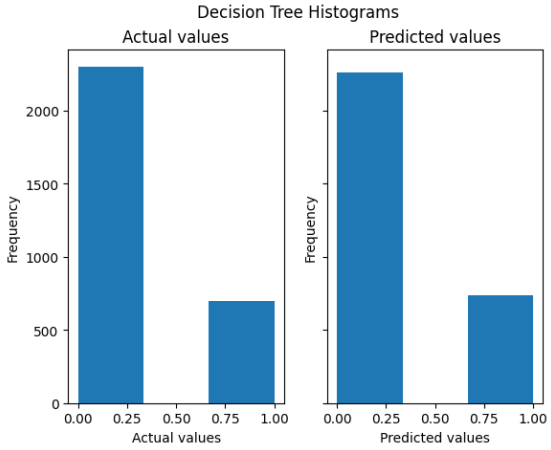
These two models also had by far the highest F1 scores, ranked as:

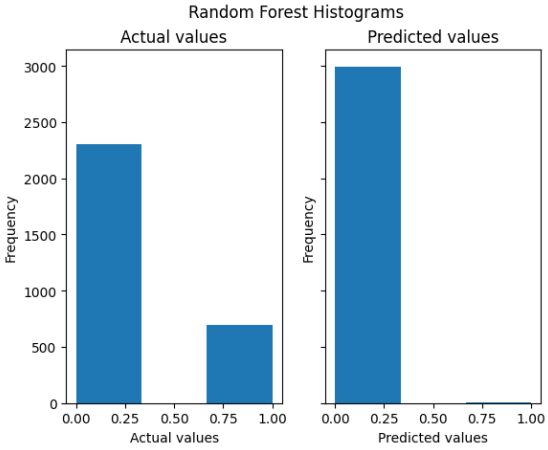
1. 76.19% - AdaBoost
2. 69.69% - Decision Tree

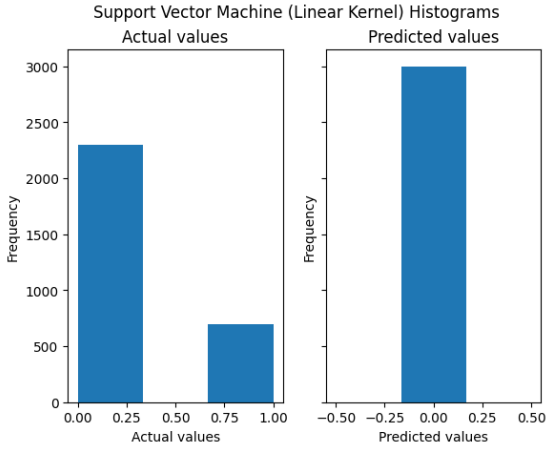
This shows our AdaBoost model for predicting country hits is our best quality model overall across all models we trained, while also yielding a high prediction accuracy. Interestingly enough, the models that performed well for other genres such as Gradient Boost and Random Forest had terrible F1 scores at 10.33% and 1.42%, each with a prediction accuracy of ~77%.

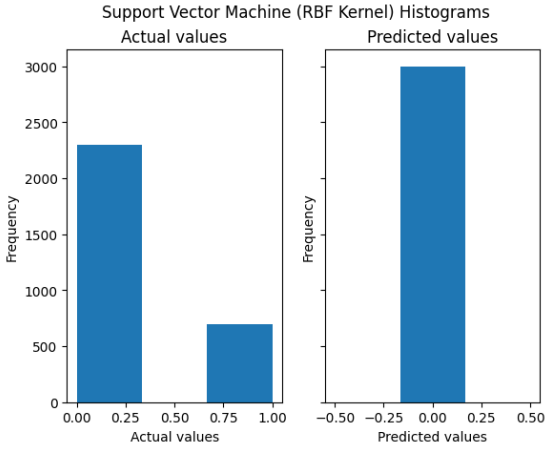


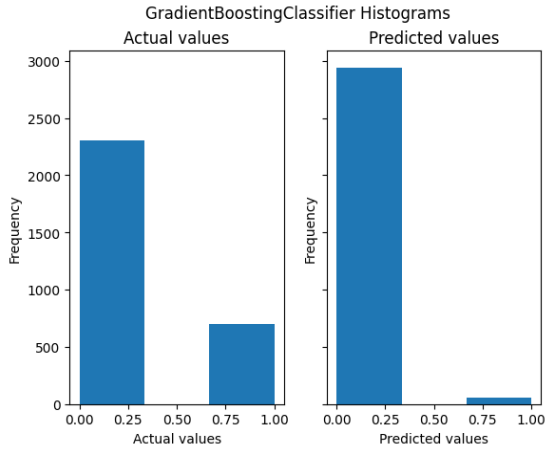


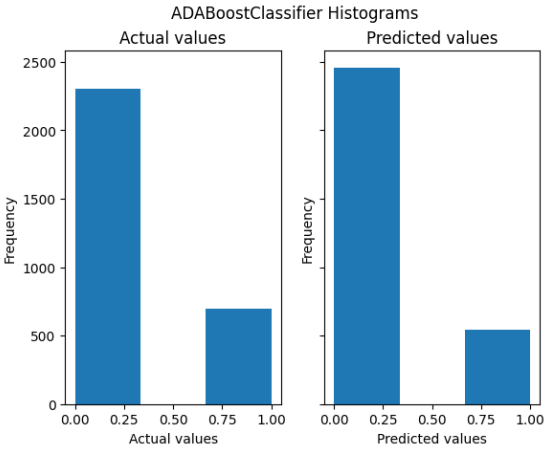





Support Vector Machine (Linear Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


Support Vector Machine (RBF Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


GradientBoostingClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


ADABoostClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values

Pop:

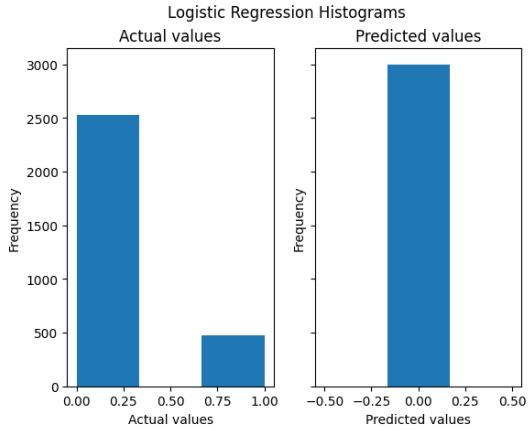
The highest performing models based on prediction accuracy were:

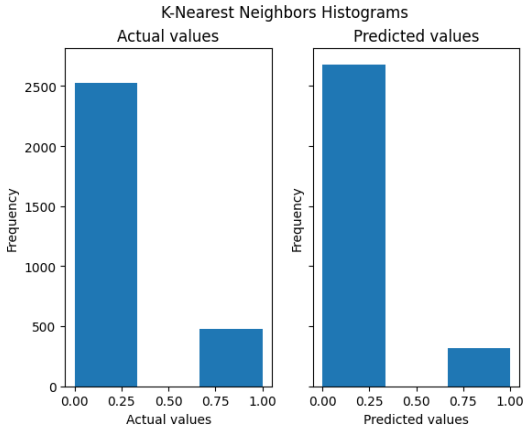
1. 93.50% - AdaBoost
2. 90.23% - Random Forest
3. 89.27% - Decision Tree

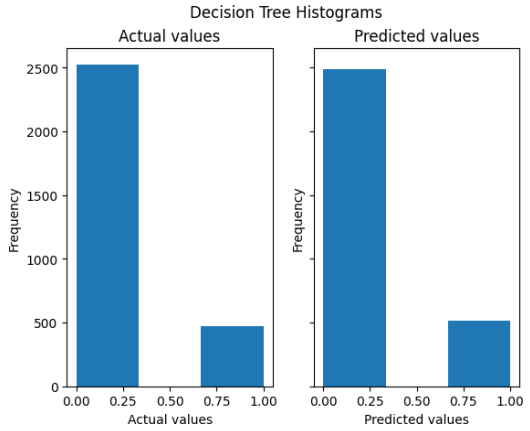
Ranking these three models by F1 score we see the following:

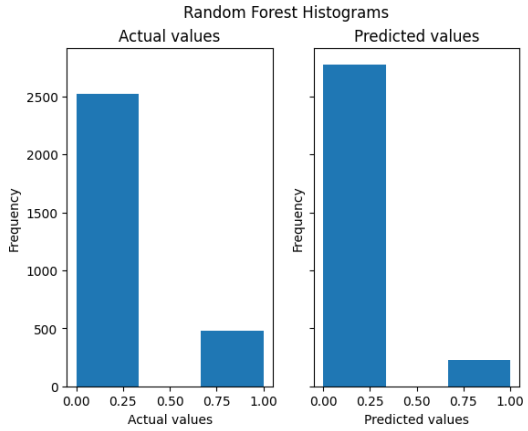
1. 75.96% - AdaBoost
2. 67.41% - Decision Tree
3. 58.08% - Random Forest

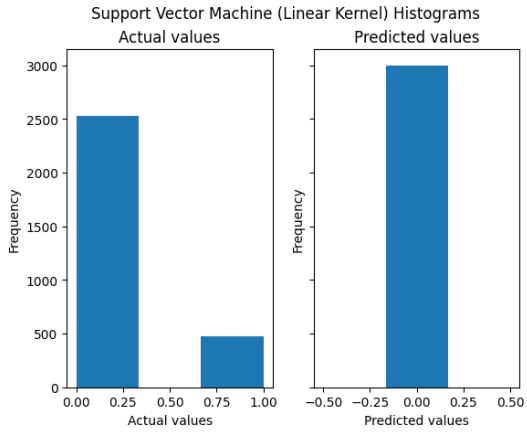
Once again, AdaBoost is the model of highest quality but also highest prediction accuracy, showing relatively strong results compared to the rest of our data when taking into account both prediction accuracy and F1 score. We consider our pop AdaBoost model to be the most legitimate predictor amongst all of our models as it has the highest prediction accuracy given a relatively strong F1 score.

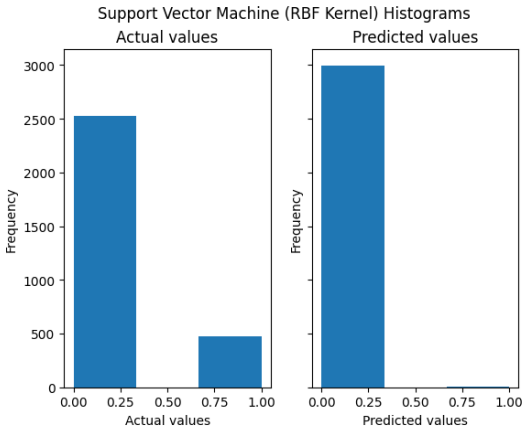


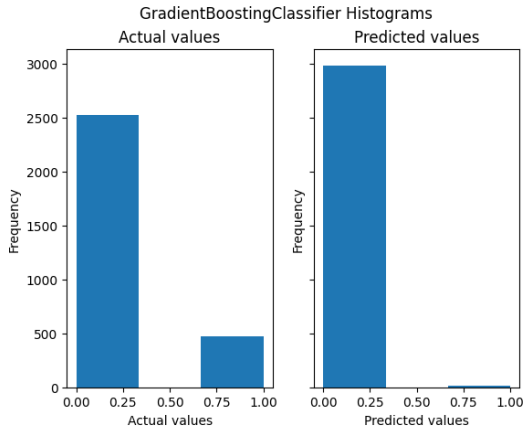


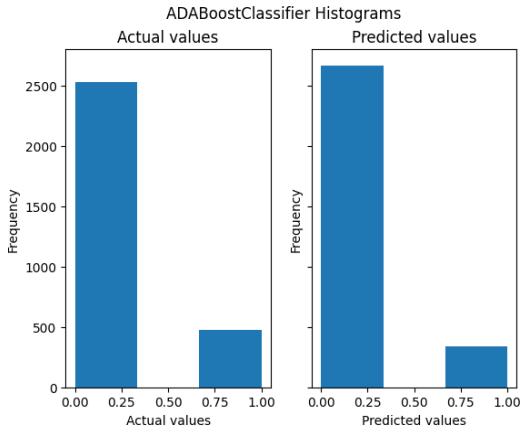





Support Vector Machine (Linear Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


Support Vector Machine (RBF Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


GradientBoostingClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


ADABoostClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values

Mixed Genres:

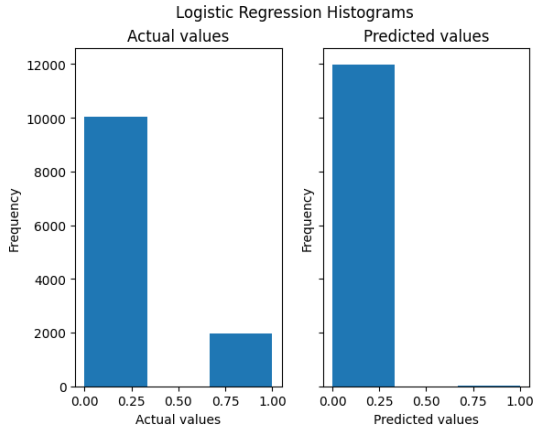
The two models that stood out based on prediction accuracy were:

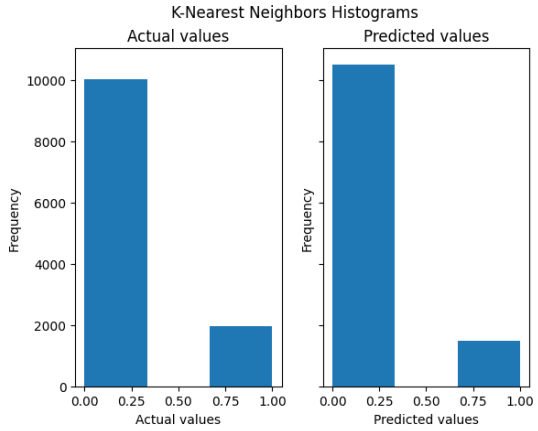
1. 91.49% - AdaBoost
2. 89.40% - Decision Tree

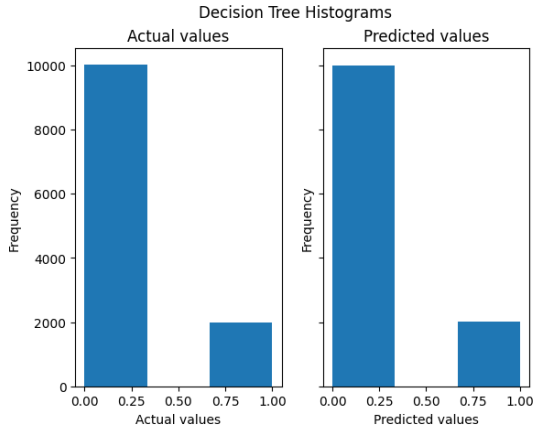
These models had nearly identical F1 scores:

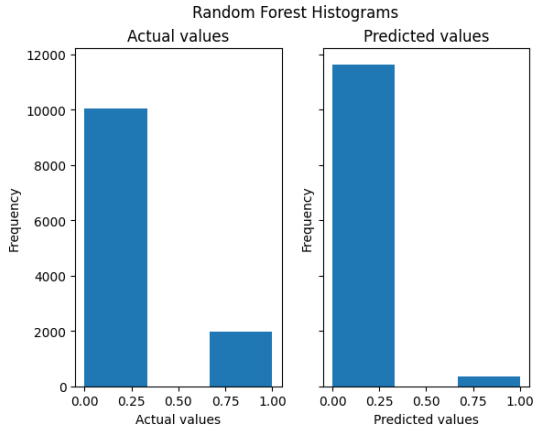
1. 68.07% - Decision Tree
2. 68.06% - AdaBoost

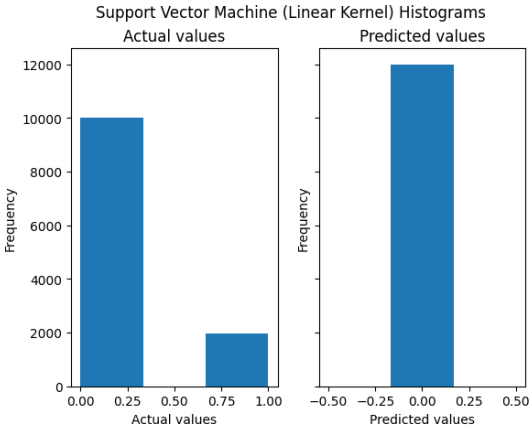
While the prediction accuracy for these models were high, they were nearly identically poor in terms of classification quality. Logically this makes sense in comparison to the quality of our genre-specific models, as it seems that models trained on individual genres performed better than those trained on all genres mixed together.

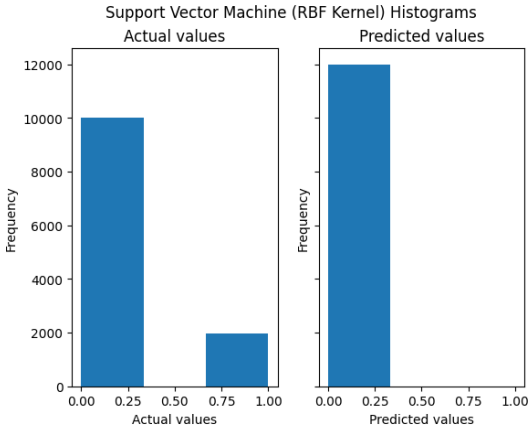


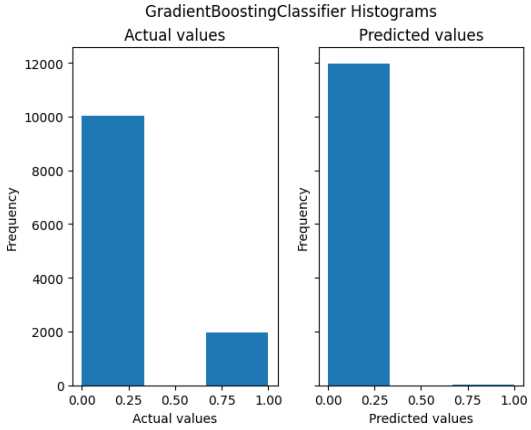


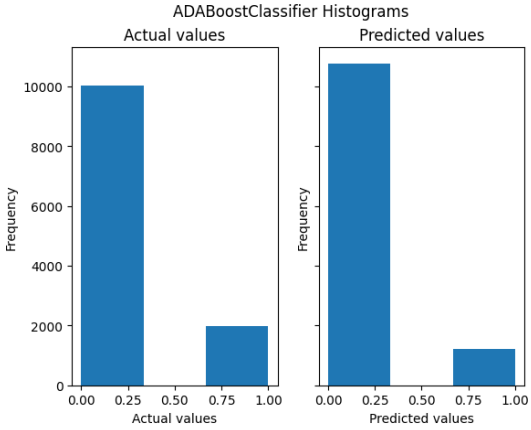





Support Vector Machine (Linear Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


Support Vector Machine (RBF Kernel) Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


GradientBoostingClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values


ADABoostClassifier Histograms
Actual values
Predicted values
Frequency
Actual values
Predicted values

Classic Model Analysis Overall (Sabeeh)

Model \ Genre	Rap	Pop	Jazz	Country	Mix
Logistic Regression	78.03%	84.17%	96.57%	76.63%	83.49%
K-Nearest Neighbors	77.33%	83.67%	96.20%	75.97%	82.45%
Decision Tree	86.43%	89.27%	97.03%	85.53%	89.40%
Random Forest	90.83%	90.23%	97.53%	76.93%	86.13%
Support Vector Machine (Linear Kernel)	78.50%	84.17%	96.60%	76.77%	83.55%
Support Vector Machine (Linear Kernel)	78.80%	84.23%	96.60%	76.77%	83.56%
GradientBoostingClassifier	90.50%	84.43%	97.10%	77.43%	83.65%
ADABoostClassifier	91.43%	93.50%	97.50%	90.17%	91.49%

Consistently, for all of our datasets, the linear regression model, K-nearest neighbor model, and the Support Vector Machines failed to yield a high F1 score as well as a high accuracy percentage. With respect to our linear regression model, it may be that there is no linear relationship between hit tracks and the features we provided the model. This seems to be likely due to the fact that our model almost always predicted that the song is not a hit. This indicates that the model is not truly learning and is more likely returning an almost constant value. This is also complemented by the fact that many of the “hit” and “no hit” distributions in the datasets were 16% split. So the model could’ve predicted “no-hit” majority of the time and still yield a high accuracy. This habit is also shown in the high accuracys and low F1 scores.

In regard to our K-nearest Neighbors model, the reason that this model performed poorly is likely due to the fact that we have imbalanced data. As shown above when discussing our data, in all of our datasets, we have significantly more non-hits in our dataset as opposed to hits. Furthermore, the data does not have expected clusterning patterns that would allow a KNN model to work efficiently using vector metrics. This is assumed to be from the diversity of music

even within a genre and beyond. KNN models are also highly sensitive to imbalanced datasets because of their reliance on majority voting to make predictions, thus the KNN model failed to predict hits properly.

The failure of the SVMs could occur for two reasons, firstly, non-linearly separable data, and secondly imbalanced data. The argument for non-linear separability could be due to a lack of existence of a kernel that allows for us to transform our data to be linearly separable. However, more likely is the fact that our data is imbalanced, as described above and in previous paragraphs.

While creating our models we based the model success purely based on accuracy. We failed to consider that this was simply due to the imbalanced data that we were using. During our final analysis of our data, we began to look more closely at indicators such as F1 score and saw this mistake. To improve this we may opt to tune our models based on both accuracy and F1 score in the future.

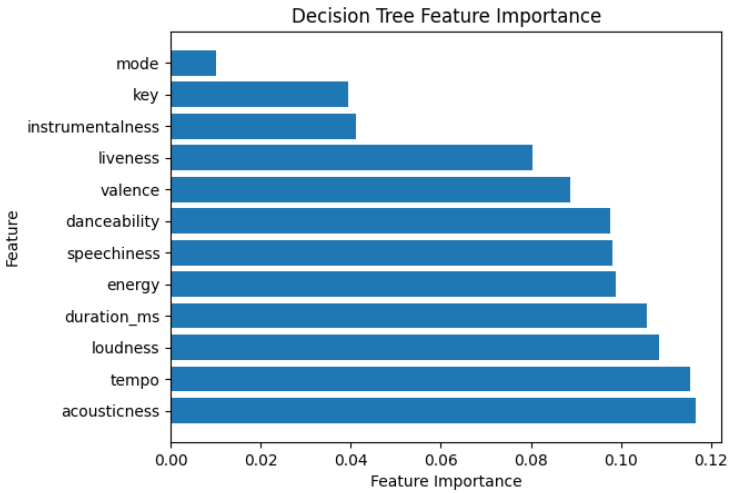
Feature Importance Analysis: (Thomas)

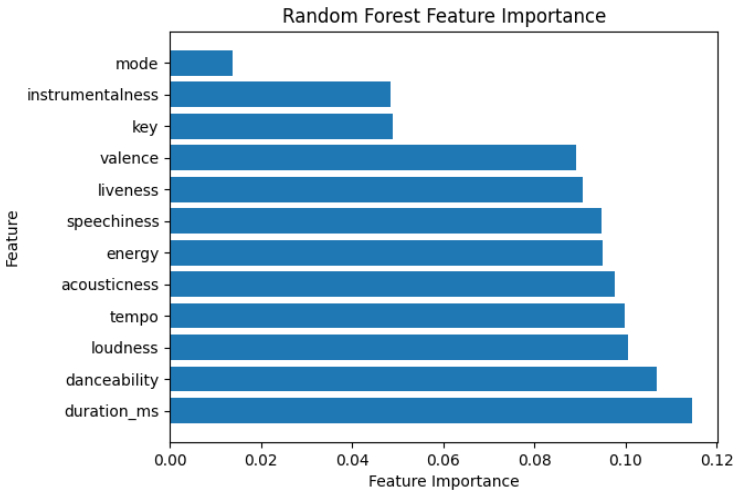
Once all of our models were trained, we analyzed the importance of each feature with respect to the individual models, giving significantly higher regard to models with higher F1 scores. Generally speaking, we found that the feature that had the highest impact on whether a song is a hit is shockingly duration.

Behind duration, most models saw loudness, speechiness, and tempo to be important features of a hit song, while key and mode consistently have little relevance to whether a song is a hit or not. This makes sense as it seems that what key or mode an artist chooses to use for their track doesn't seem to determine whether people enjoy a song or not. From genre to genre, we see that rap places a lot of importance on acousticness, country places a lot of weight on loudness and tempo, pop tends to give a high weight value to tempo, and jazz has a high weight for liveness and danceability.

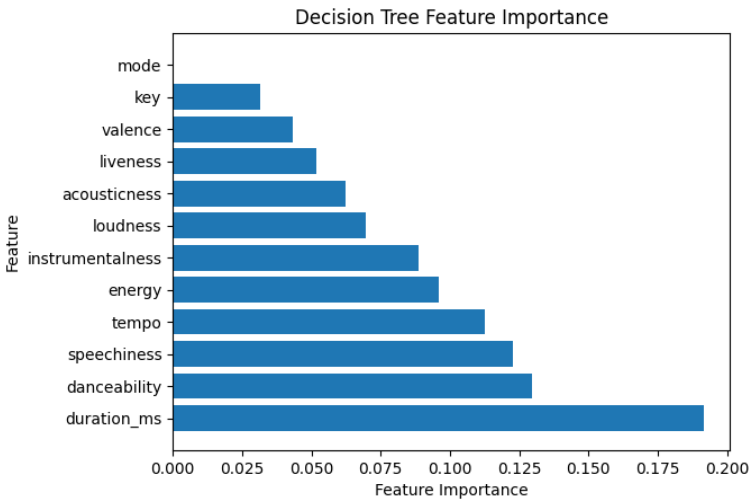
Genre was surprisingly unimportant for the mixed data set with our two highest rated models, namely Decision Tree model and AdaBoost model having that feature ranked at the 4th and 2nd least important features, having weight values of 6.0796% and 3.4685% respectively. Although not the lowest value, it is interesting to note that genre did not play a significant role in classifying the tracks in the mixed dataset.

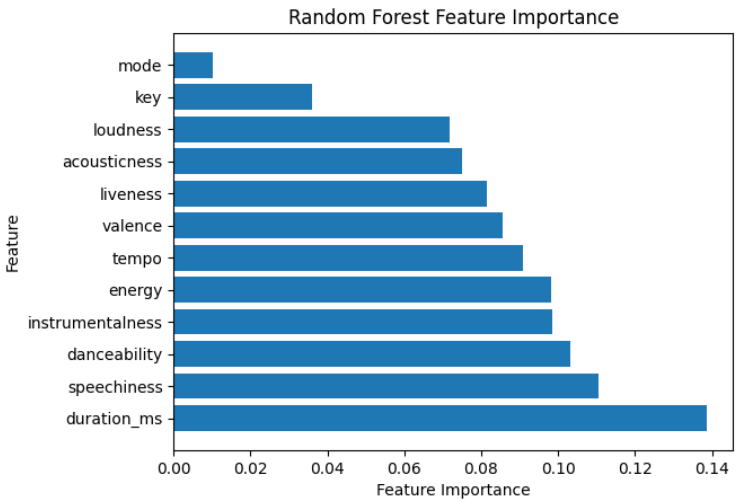
Rap:

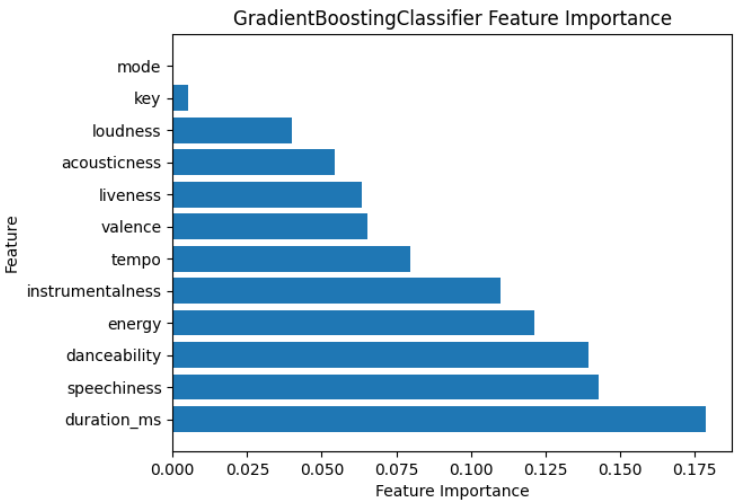


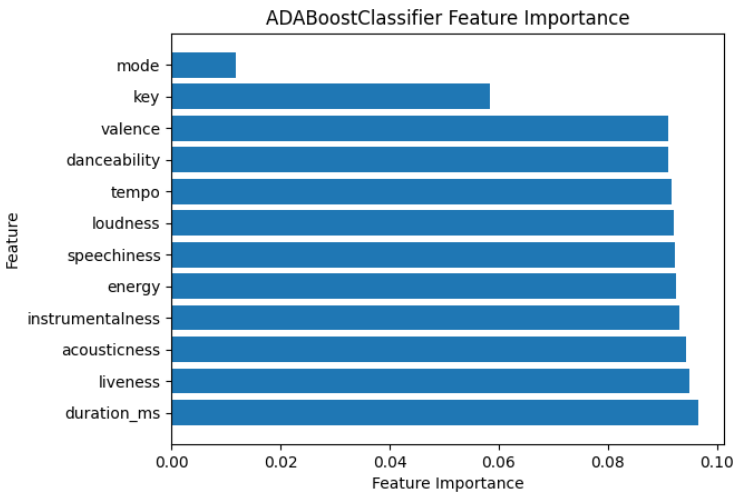


Jazz:

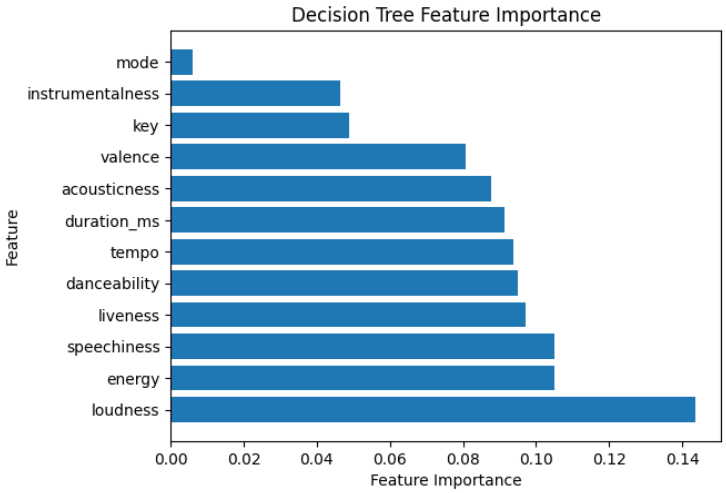


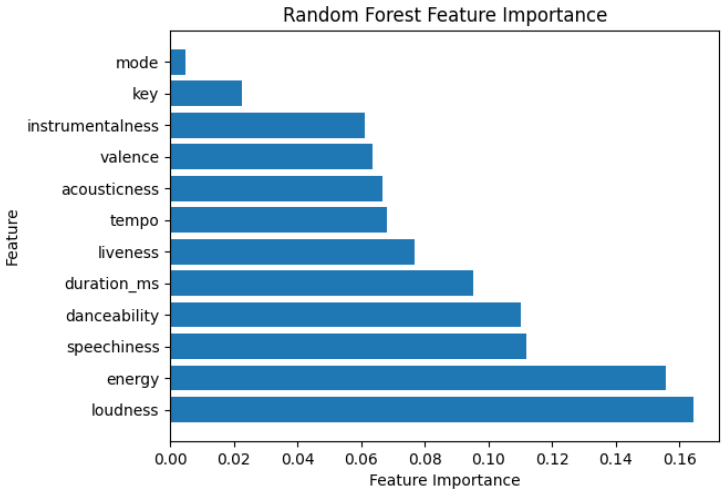


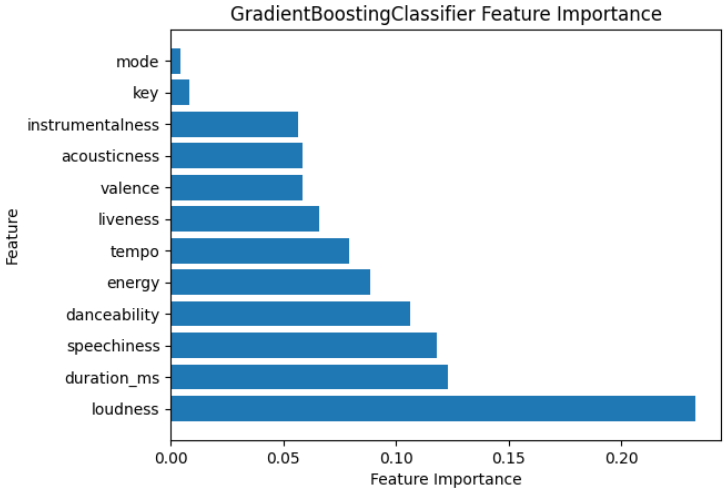


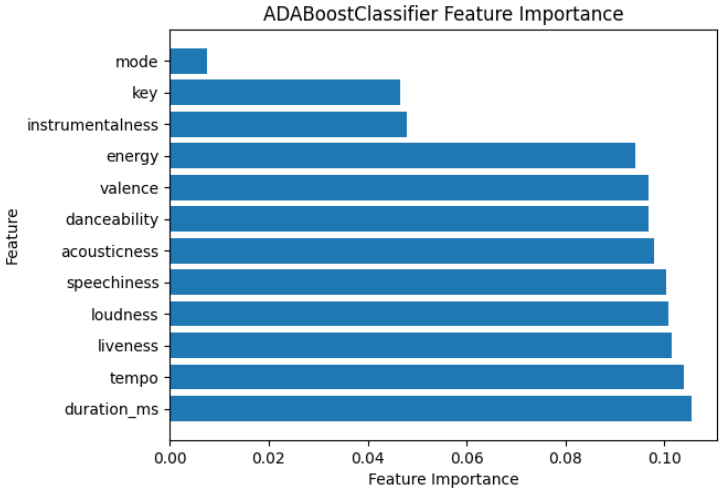


Country:

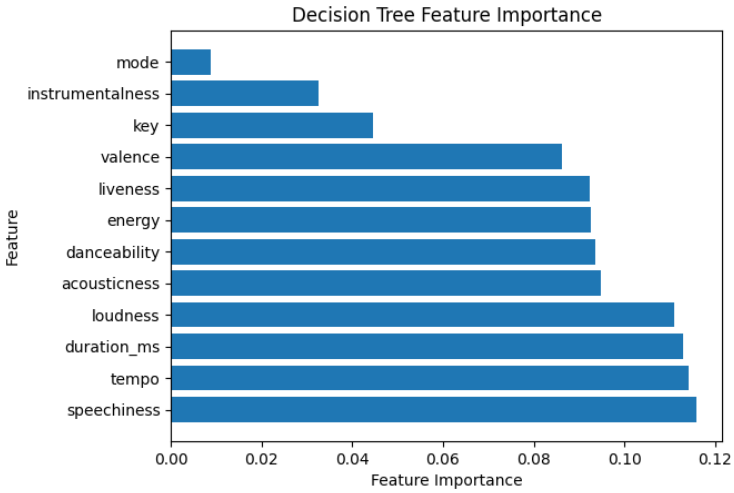


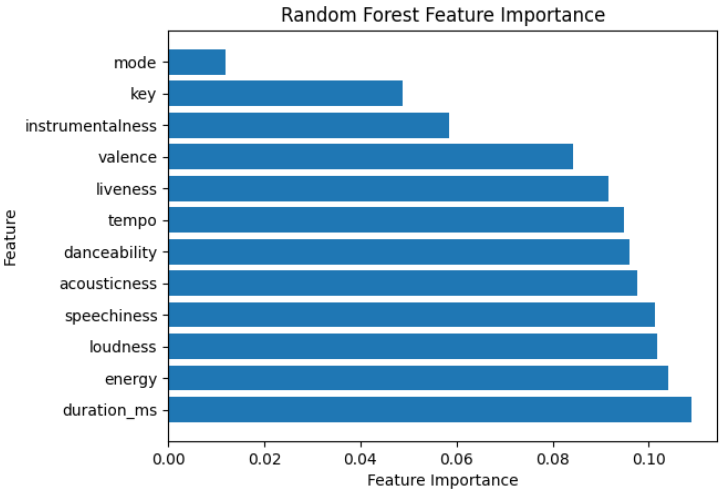


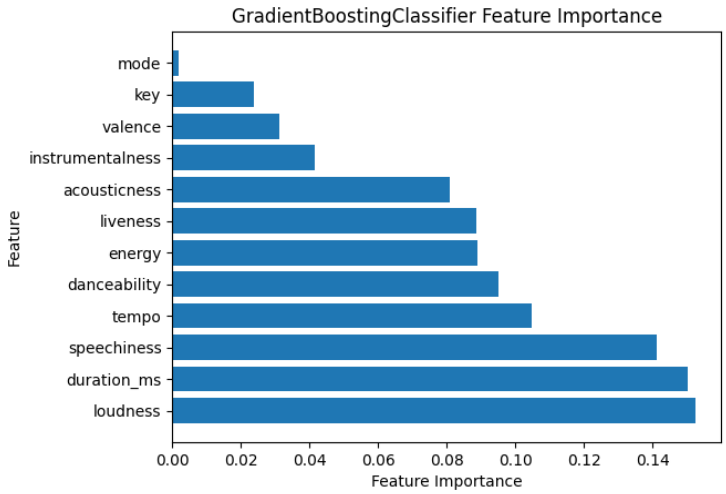

GradientBoostingClassifier Feature Importance
mode
key
instrumentalness
acousticness
valence
liveness
tempo
energy
danceability
speechiness
duration_ms
loudness
Feature
0.00
0.05
0.10
0.15
0.20
Feature Importance

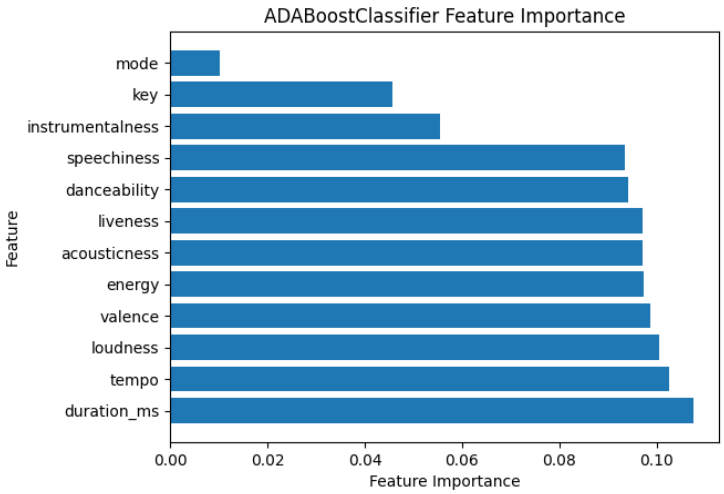

ADABoostClassifier Feature Importance
mode
key
instrumentalness
energy
valence
danceability
acousticness
speechiness
loudness
liveness
tempo
duration_ms
Feature
0.00
0.02
0.04
0.06
0.08
0.10
Feature Importance

Pop:

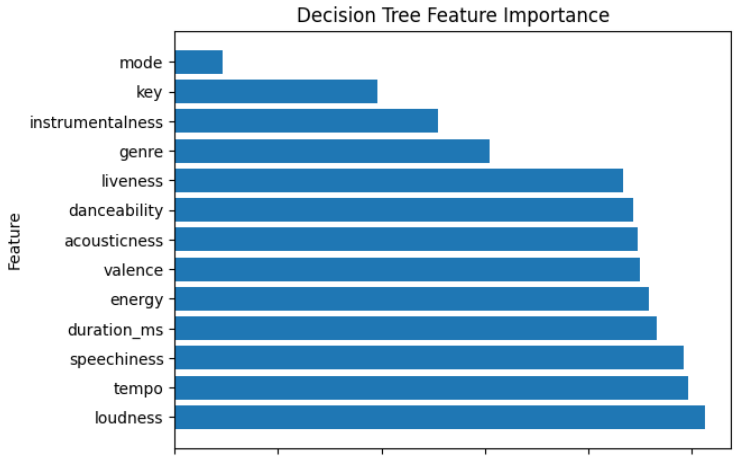

Decision Tree Feature Importance
mode
instrumentalness
key
valence
liveness
energy
danceability
acousticness
loudness
duration_ms
tempo
speechiness
Feature
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Feature Importance

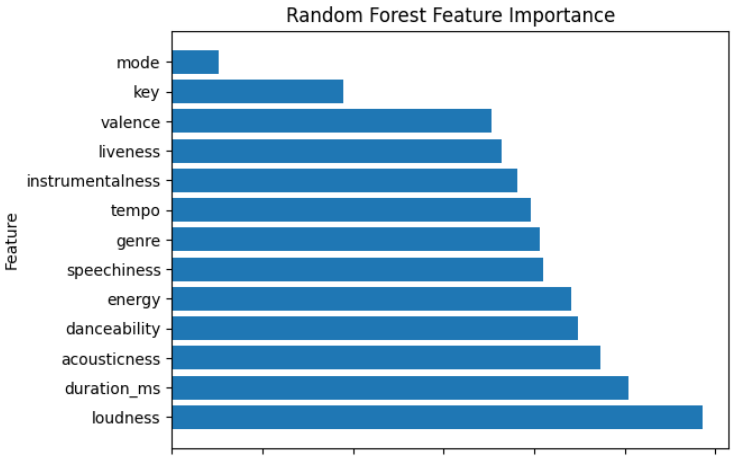

Random Forest Feature Importance
mode
key
instrumentalness
valence
liveness
tempo
danceability
acousticness
speechiness
loudness
energy
duration_ms
Feature
0.00
0.02
0.04
0.06
0.08
0.10
Feature Importance

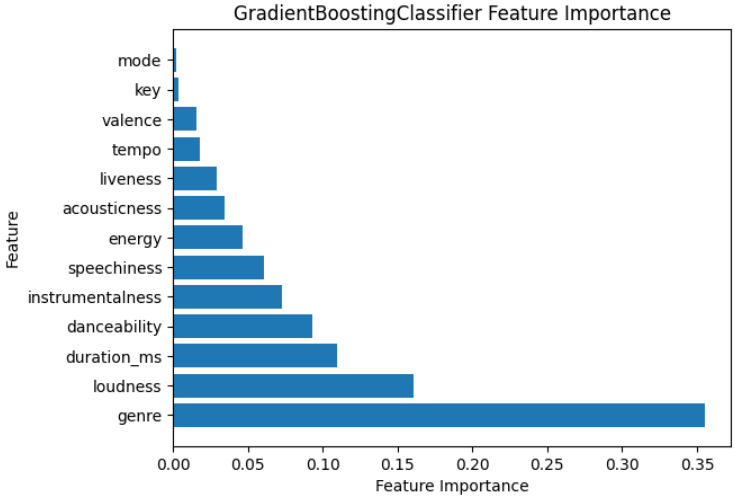

GradientBoostingClassifier Feature Importance
mode
key
valence
instrumentalness
acousticness
liveness
energy
danceability
tempo
speechiness
duration_ms
loudness
Feature
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
Feature Importance

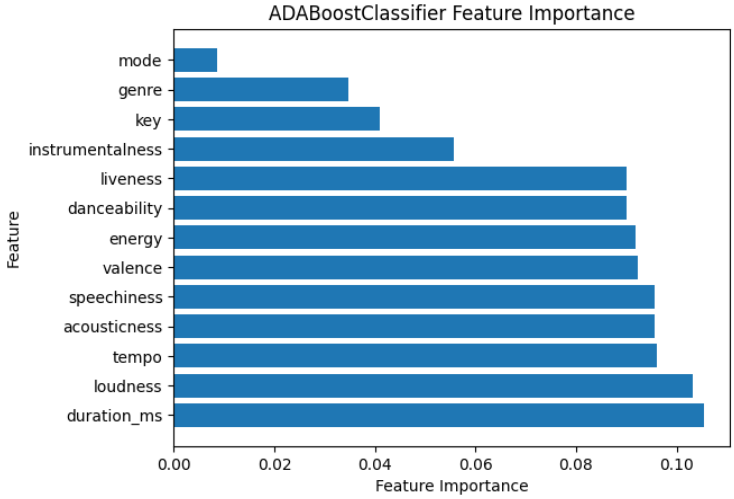

ADABoostClassifier Feature Importance
mode
key
instrumentalness
speechiness
danceability
liveness
acousticness
energy
valence
loudness
tempo
duration_ms
Feature
0.00
0.02
0.04
0.06
0.08
0.10
Feature Importance

Mixed:


Decision Tree Feature Importance
Feature
mode
key
instrumentalness
genre
liveness
danceability
acousticness
valence
energy
duration_ms
speechiness
tempo
loudness


Random Forest Feature Importance
Feature
mode
key
valence
liveness
instrumentalness
tempo
genre
speechiness
energy
danceability
acousticness
duration_ms
loudness


GradientBoostingClassifier Feature Importance
Feature
mode
key
valence
tempo
liveness
acousticness
energy
speechiness
instrumentalness
danceability
duration_ms
loudness
genre
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Feature Importance


ADABoostClassifier Feature Importance
Feature
mode
genre
key
instrumentalness
liveness
danceability
energy
valence
speechiness
acousticness
tempo
loudness
duration_ms
0.00
0.02
0.04
0.06
0.08
0.10
Feature Importance

Neural Network Analysis: (Jason)

Table 1: Accuracy of neural network classification models

Genre	Accuracy
Rap	90.65%
Pop	90.70%
Jazz	96.40%
Country	87.35%
All Genres	90.19%
All Genres (with genre as a feature)	91.74%

Based on the results outlined in Table 1, our neural network performed best for our Jazz music dataset. However, this may be attributed to the Jazz dataset being the least balanced of the datasets (3.21% hits, 96.79% non-hits). Nonetheless, our model always performed with greater than 87% accuracy, which indicates that our model was not overfitting. Additionally, it was interesting that in our two all genre datasets, the dataset where genre is included as a feature performed slightly better. This may be explained by certain genres of songs (such as rap and pop) being more likely to be a hit than other genres (country and jazz).

Like stated before, we chose to use Adam Optimizer because it is a further extension of stochastic gradient descent that adjusts the weights of the neural network in real-time. It is a good optimizer because it only requires first-order gradients with little memory requirement. We chose to use binary cross-entropy loss function because we are doing a binary classification task. Finally, we used the Sigmoid function as our activation function to make our final classification of hit or not a hit. In the future, we can experiment with the numbers of layers, the optimizer, the activation functions, and adding dropout layers.

Future Actions: (Thomas & Jackson)

- An analysis of changes in time period would be interesting. We could evaluate how effective the model is from music in past years. We could also create a model on music from the past and analyze it on music of today.
- Another parameter that could be indicative in predicting a hit or not is the number of followers or monthly listeners the artist has on social media. Although this does not affect the song itself, it would be interesting to analyze whether a model would show a correlation between follower count and hit probability.
- Analyze how effective our models are on music from genres we have not analyzed, ie model pop hits based on models generated by rap hits etc.
- Create a more balanced positive/negative labeled dataset for each genre and see how that impacts F1 score. Doing so would yield a more balanced data set, however it would fail to accurately portray the ratio of hits to non-hits in the music industry.

Conclusion: (Jackson)

From the analysis of each of our models in the context of each evaluation we performed, it is clear that we were naive in our approach to create high quality models that predicted whether or not a song will be a hit. It seems as though our process of randomly fetching songs that are or are not historically hits produced a largely negatively-labeled dataset.

While this is reflective of the real world, as most songs produced are not hits, we have come to the conclusion that it may have been more effective to train our models on a more evenly distributed dataset in terms of positive/negative labels so that the models better learn the qualities of a hit song. We also have reflected on our process of tuning the classic models, as we did so focusing solely on improving prediction accuracy. We believe this led to some hyperparameters that produced sub-par F1 scores, while other hyperparameter combinations may have yielded significantly better F1 scores.

Overall, the process of training and analyzing these models has taught us a lot about building quality datasets and models for a desired outcome. We are able to see now what potential changes can be made to improve the quality of our results, as well as how to better focus our efforts as we formulate an approach.

Contributions:

Report:

- All written contributions are indicated by the names in parentheses next to each subtitle throughout the report.

Video:

- Thomas: Audio Recording
- Jackson: Audio Recording
- Sabeeh: Slideshow, Editing
- Jason: Audio Recording, Slideshow, Editing

Code:

- Thomas:
    - Data Sourcing (fetching track features)
    - Gathering/choosing model statistics/metrics (live discussion with Jackson)
    - Visualization of Model Results
    - AdaBoost implementation
- Jackson:
    - Data Sourcing (API querying, normalization, joining data from the two APIs)
    - Classical Model Training/Hypertuning
    - Gathering/choosing model statistics/metrics (live discussion with Thomas)
- Sabeeh:
    - Organizing Group Direction
    - Data Preprocessing
    - Classical Model Training/Compiling/Hypertuning
    - Researching Models
- Jason
    - Building datasets
    - Neural network
    - Sklearn models