Spotify Hit Predictor

Christopher El-Khouri

625.740: Data Mining

November 17, 2021

1. Introduction

Spotify is one of the most popular music streaming services in the world with 381 million active monthly users as of October 2021 (Spotify Technology S.A, 2021). In addition to being a digital music streaming platform, Spotify created a platform called Spotify for Developers (Spotify AB, 2021) that provides users with a set of tools to retrieve Spotify measured audio features for the different tracks on the application. Audio features such as danceability, energy, instrumentalness, tempo, and more can be extracted.

In this project, we are going to develop a classification model that uses Spotify audio feature data to predict whether or not a track was a hit i.e. achieved mainstream popularity. The anticipated challenges are as follows:

- Different audio features may have different effects on popularity depending on the time of release of a musical track. For example: The 1980s may have had a higher mainstream preference for more danceable music compared to the 1970s so a "dancey" track released in the 70s may have not been a hit but would have been if it were released in the 1980s.
- Different audio features may have different effects on popularity depending on the genre of a musical track. For example: A rock song with a high danceability measure may not be a hit as much as a Hip hop song with the same danceability.
- A track could be a hit because of the artist and album it was released with, regardless of the audio features and this may skew our analysis and modeling.

The performance of 4 algorithms will be assessed on the same dataset; Logistic Regression, Decision Trees, K-Nearest Neighbors, and Feedforward Neural Networks.

2. Background

This report continues my previous research conducted in the Spring of 2020 titled "Music Taste Across Decades: Spotify Audio Feature Analysis of 33,354 Musical Pieces" (El Khouri, 2020). The research paper was for the Data Visualization course (605.662) and it performs a statistical and visual analysis of audio feature data of 33,354 musical tracks whose audio features have been measured by Spotify. Hypothesis testing was performed testing the relationships between genres, popularity, audio features, and time. The audio features; instrumentalness, danceability, valence, and energy were found to be correlated with popularity. Based on the results, I expect to build a reliable model that is able to predict the hit status of an arbitrary track using its audio features, decade, and genre.

3. Algorithms and Experimental Methods

The sections below elaborate upon the algorithms and experimental methods used in this project.

3.1. Logistic Regression

In the field of machine learning, linear models are the simplest of algorithms and function by generating a line of best fit to successfully predict the value of the target variable using the feature attributes. A typical linear model has the following form:

$$f(x) = w_0 + \sum_{i=1}^{d} w_i x_i = w_0 + \boldsymbol{w}_i^T \boldsymbol{x} \qquad (1)$$

Logistic Regression is a supervised linear machine learning algorithm that uses the logistic distribution to classify a dependent variable. In equation (2), $P(C_i|x)$ refers to the probability that input feature data x belongs to a class C_i.

Taking $f(x)$ from (1):

$$P(C_i|x) = \frac{\exp(f(x_i))}{\sum_{j=1}^{K} \exp(f(x_j))}, i = 1, \dots, K \qquad (2)$$

The Logistic Regression algorithm used in this project is the statsmodels Logit algorithm (Seabold & Perktold 2010).

Backward Elimination

The Logistic Regression model is initially run with all the features from the dataset. Based on the resulting p-values of the individual features, backward elimination is implemented where the feature with the highest p-value greater than 0.05 is eliminated. The algorithm is run on the modified dataset again until there are no more features with p-values greater than 0.05. The final modified dataset is the dataset that will be used to assess the performance of the rest of the algorithms.

3.2. Decision Trees

Decision Trees are a type of supervised learning algorithms that can be used for classification and regression. When used for classification, these trees have a target variable that takes discrete values. The branches are observations about the attributes and the leaves are the different class labels that our data could hold.

An example of a classification tree is shown Figure 1 (Colton, 2006), the branches are based on the following attributes: Parents Visiting, Weather, and Money. The leaves are based on the following class labels: Cinema, Play Tennis, Stay in, and Shopping. The decision tree is used by starting from the top, “Parents Visiting” in this case, and traverses until we arrive at a leaf.

The Decision Tree algorithm used in this project is the scikit-learn tree algorithm (Pedregosa et al., 2011).

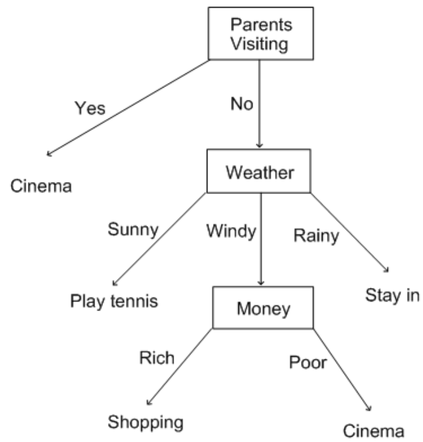

Figure 1: Classification Tree Example

Minimal Cost-Complexity Pruning

Decision Trees can be pruned to avoid overfitting and improve model performance. The scikit-learn algorithm implements Minimal Cost-Complexity Pruning (Pedregosa et al., 2011) and is demonstrated by equation (3) below:

$$R_\alpha(T) = R(T) + \alpha|\widetilde{T}| \quad (3)$$

α: *Complexity Parameter*

$|\widetilde{T}|$: *Number of leaves in tree T*

$R(T)$: *Total misclassification rate of the leaves*

Minimal Cost-Complexity Pruning finds the subtree that minimizes $R_\alpha(T)$. 10% of our data is extracted to form a Tuning Set. The decision tree algorithm is implemented on our Tuning Set with α varying from 0 to 0.5 to fine tune the complexity parameter and maximize model performance. The α corresponding to the maximal performance on the Tuning Set is implemented for the remaining 90% of the data.

3.3. K-Nearest Neighbors

K-Nearest Neighbors (K-NN) is a non-parametric method initially proposed by Thomas Cover (1967) that can be used for classification or regression. When used for classification, the algorithm takes the input data and returns the most common class among its k nearest neighbors from the training data. The figure below (Yadnesh, 2020) shows the working principle of K-NN.

The K-NN classification algorithm used in this project is the scikit-learn KNeighborsClassifier (Pedregosa et al., 2011). 10% of our data is extracted to form a Tuning Set. The K-NN algorithm is implemented on our Tuning Set with varying values of k to fine tune the number of neighbors and maximize model performance. The k corresponding to the maximal performance on the Tuning Set is implemented for the remaining 90% of the data.

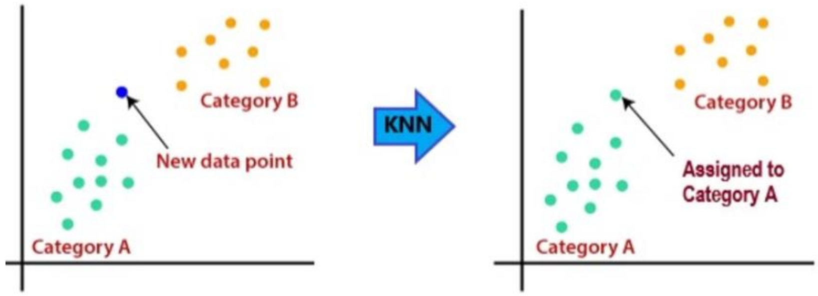

Figure 2: K-NN Example

3.4. Feedforward Neural Networks

Feedforward Neural Networks are a type of Artificial Neural Networks (ANN) that generally utilize perceptrons, which are supervised linear learning algorithms that can be used for classification or regression.

Figure 3 below shows a simple perceptron:

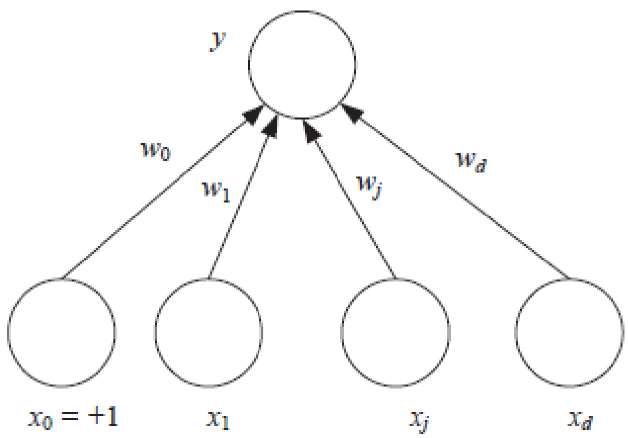

Figure 3: Simple Perceptron

The output y has the form of a typical linear model:

$$(4) \qquad y = w_0 + \sum_{i=1}^{d} w_i x_i$$

Multilayer Perceptrons (MLP) are perceptrons with 1 or more hidden layers, where the neurons in one layer are connected to the neurons in the other in a feedforward type structure. Figure 4 shows a 1-layer feedforward neural network with multiple outputs, $z_{1 \ldots h}$ are the activation functions that typically take sigmoid, tanh, or gaussian transformations.

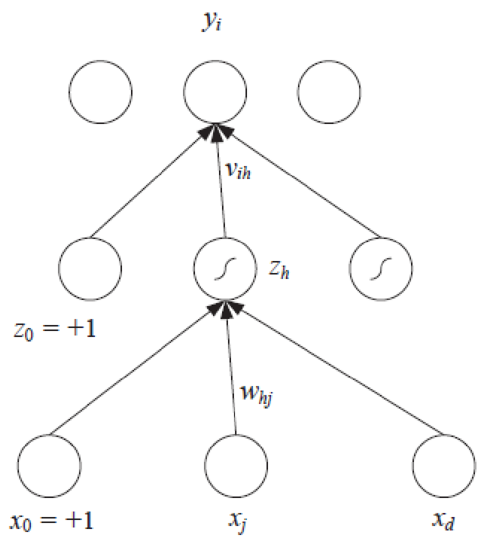

Figure 4: 1-layered MLP

The Multilayer Perceptron classification algorithm used in this project is the scikit-learn MLPClassifier (Pedregosa et al., 2011). 10% of our data is extracted to form a Tuning Set. The MLP algorithm is implemented on our Tuning Set with varying numbers of hidden dimensions to maximize model performance. The hidden dimensions corresponding to the maximal performance on the Tuning Set is implemented for the remaining 90% of the data.

3.5. 5-fold cross-validation

For each of the models, the data underwent 5-fold cross-validation, where the data is split into five parts (or folds) and the model is trained on each of the folds accordingly. In each run, one of the five folds is used as a test set and the remaining four folds are concatenated and used as the training set. The performance of each model is determined by calculating the classification errors of each of the folds by dividing the number of incorrect predictions over the total size of the folds, then calculating the mean of the errors.

4. Dataset

The dataset used is the same data set used in my previous research paper Music Taste Across Decades: Spotify Audio Feature Analysis of 33,354 Musical Pieces (El Khouri, 2020), which is based on The Spotify Hit Predictor Dataset (1960-2019) (Ansari, 2020). In my research paper, I preprocessed the dataset to group the tracks by genre and removed outliers accordingly. Further details to the preprocessing can be found on the paper.

The dataset used in this project initially contained 33,354 tracks, 31 features, and 2 classes: Hit (denoted by '1') and Flop (denoted by '0'). Further preprocessing was performed in order to prepare our data for modeling which resulted in the final dataset containing 58 features. The feature attributes; loudness, tempo, duration_ms, chorus_hit, and sections were scaled to have their values between 0 and 1. Dummy variables were created for the features time_signature, genres, Decade, key_full. The features of the final dataset are listed below.

```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
0   danceability            33354 non-null  float64
1   energy                  33354 non-null  float64
2   loudness                33354 non-null  float64
3   speechiness             33354 non-null  float64
4   acousticness            33354 non-null  float64
5   instrumentalness        33354 non-null  float64
6   liveness                33354 non-null  float64
7   valence                 33354 non-null  float64
8   tempo                   33354 non-null  float64
9   duration_ms             33354 non-null  float64
10  chorus_hit              33354 non-null  float64
11  sections                33354 non-null  float64
 12  target                  33354 non-null  int64
 13  time_signature_1        33354 non-null  uint8
 14  time_signature_3        33354 non-null  uint8
 15  time_signature_4        33354 non-null  uint8
 16  time_signature_5        33354 non-null  uint8
 17  genres_Caribbean        33354 non-null  uint8
 18  genres_Classical        33354 non-null  uint8
 19  genres_Country          33354 non-null  uint8
 20  genres_Easy listening   33354 non-null  uint8
 21  genres_Electronic       33354 non-null  uint8
 22  genres_Folk             33354 non-null  uint8
 23  genres_Hip hop          33354 non-null  uint8
 24  genres_Jazz             33354 non-null  uint8
 25  genres_Latin            33354 non-null  uint8
 26  genres_Metal            33354 non-null  uint8
 27  genres_Pop              33354 non-null  uint8
 28  genres_R&B              33354 non-null  uint8
 29  genres_Rock             33354 non-null  uint8
 30  Decade_10s              33354 non-null  uint8
 31  Decade_60s              33354 non-null  uint8
 32  Decade_70s              33354 non-null  uint8
 33  Decade_80s              33354 non-null  uint8
 34  Decade_90s              33354 non-null  uint8
 35  key_full_A Minor        33354 non-null  uint8
 36  key_full_B Major        33354 non-null  uint8
 37  key_full_B Minor        33354 non-null  uint8
 38  key_full_Bb Major       33354 non-null  uint8
 39  key_full_Bb Minor       33354 non-null  uint8
 40  key_full_C Major        33354 non-null  uint8
 41  key_full_C Minor        33354 non-null  uint8
 42  key_full_C# Major       33354 non-null  uint8
 43  key_full_C# Minor       33354 non-null  uint8
 44  key_full_D Major        33354 non-null  uint8
 45  key_full_D Minor        33354 non-null  uint8
 46  key_full_E Major        33354 non-null  uint8
 47  key_full_E Minor        33354 non-null  uint8
 48  key_full_Eb Major       33354 non-null  uint8
 49  key_full_Eb Minor       33354 non-null  uint8
 50  key_full_F Major        33354 non-null  uint8
 51  key_full_F Minor        33354 non-null  uint8
 52  key_full_F# Major       33354 non-null  uint8
 53  key_full_F# Minor       33354 non-null  uint8
 54  key_full_G Major        33354 non-null  uint8
 55  key_full_G Minor        33354 non-null  uint8
 56  key_full_G# Major       33354 non-null  uint8
 57  key_full_G# Minor       33354 non-null  uint8
```

However, after performing backward elimination, the dataset that will proceed to be used with the remaining models contains the following features:

```
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
0    danceability           33354 non-null  float64
1    energy                 33354 non-null  float64
2    loudness               33354 non-null  float64
3    speechiness            33354 non-null  float64
4    acousticness           33354 non-null  float64
5    instrumentalness       33354 non-null  float64
6    liveness               33354 non-null  float64
7    valence                33354 non-null  float64
8    tempo                  33354 non-null  float64
9    chorus_hit             33354 non-null  float64
10   genres_Caribbean       33354 non-null  uint8  
11   genres_Classical       33354 non-null  uint8  
12   genres_Country         33354 non-null  uint8  
13   genres_Easy listening  33354 non-null  uint8  
14   genres_Electronic      33354 non-null  uint8  
15   genres_Folk            33354 non-null  uint8  
16   genres_Hip hop         33354 non-null  uint8  
17   genres_Latin           33354 non-null  uint8  
18   genres_Metal           33354 non-null  uint8  
19   genres_Pop             33354 non-null  uint8  
20   genres_R&B             33354 non-null  uint8  
21   genres_Rock            33354 non-null  uint8  
22   Decade_10s             33354 non-null  uint8  
23   Decade_60s             33354 non-null  uint8  
24   Decade_70s             33354 non-null  uint8  
25   Decade_80s             33354 non-null  uint8  
26   Decade_90s             33354 non-null  uint8  
27   key_full_A Minor       33354 non-null  uint8  
28   key_full_B Minor       33354 non-null  uint8  
29   key_full_Bb Major      33354 non-null  uint8  
30   key_full_C Major       33354 non-null  uint8  
31   key_full_C# Major      33354 non-null  uint8  
32   key_full_C# Minor      33354 non-null  uint8  
33   key_full_D Minor       33354 non-null  uint8  
34   key_full_E Major       33354 non-null  uint8  
35   key_full_E Minor       33354 non-null  uint8  
36   key_full_Eb Major      33354 non-null  uint8  
37   key_full_F Major       33354 non-null  uint8  
38   key_full_F# Major      33354 non-null  uint8  
39   key_full_G# Major      33354 non-null  uint8  
40   target                 33354 non-null  int64  
```

A total of 40 features including the target variable. Further details can be found in the supporting documents.

5. Results

The results from our different models are shown on the table below:

Table 1: Results Table

Model	Measure	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean
Logistic Regression	Error	17.3%	17.0%	17.8%	16.6%	17.3%	**17.2%**
	False Positives	22.9%	23.2%	23.5%	22.5%	23.4%	**23.1%**
	False Negatives	11.5%	10.6%	11.8%	10.7%	11.0%	**11.1%**
Logistic Regression with Backward Elimination	Error	17.4%	16.9%	18.0%	16.5%	17.4%	**17.2%**
	False Positives	22.9%	23.2%	23.7%	22.3%	23.8%	**23.2%**
	False Negatives	11.8%	10.4%	12.0%	10.7%	10.9%	**11.2%**
Decision Tree (alpha=0.0001)	Error	18.6%	18.9%	17.8%	19.0%	19.7%	**18.8%**
	False Positives	20.5%	20.2%	19.5%	19.4%	21.9%	**20.3%**
	False Negatives	16.7%	17.6%	16.1%	18.6%	17.4%	**17.2%**
K-NN (n=5)	Error	17.7%	18.5%	17.5%	17.4%	18.8%	**18.0%**
	False Positives	23.8%	25.1%	23.1%	23.5%	24.1%	**23.9%**
	False Negatives	11.7%	11.7%	11.6%	11.1%	13.3%	**11.9%**
MLP (Hidden Layer Sizes=[3,4])	Error	14.8%	15.3%	15.8%	14.5%	16.3%	**15.3%**
	False Positives	17.5%	19.0%	19.1%	19.4%	18.0%	**18.6%**
	False Negatives	12.1%	11.4%	12.4%	9.4%	14.6%	**12.0%**

The results can be summarized as follows:

Table 2: Summary of Results

Model	Measure	Mean
Logistic Regression	Error	17.2%
	False Positives	23.1%
	False Negatives	11.1%
Logistic Regression with Backward Elimination	Error	17.2%
	False Positives	23.2%
	False Negatives	11.2%
Decision Tree (alpha=0.0001)	Error	18.8%
	False Positives	20.3%
	False Negatives	17.2%
K-NN (n=5)	Error	18.0%
	False Positives	23.9%
	False Negatives	11.9%
MLP (Hidden Layer Sizes=[3,4])	Error	**15.3%**
	False Positives	**18.6%**
	False Negatives	**12.0%**

6. Discussion

Since the goal was to predict the hit status of a track, the measures of error and false positives/negatives are important. Having a low false positive rate is more important than a low false negative rate because incorrectly predicting that a track is a hit rather than incorrectly predicting that a track is not a hit may have costly implications.

Referring to Table 2, the MLP algorithm gave us the strongest model for this data, with an error rate of 15.3 %, this model has an accuracy of approximately 85% and a false positive rate of only 18.6%. This means that the model will correctly predict a hit ~80% of the time. Since the hit status of a musical track is a sort of psychological/social science phenomenon, as it is a matter of subjective opinion, an accuracy of 85% is outstanding.

Although K-NN and Logistic Regression had slightly superior error rates over Decision Trees, Decision Trees had superior false positive rates. The slight raise in error rate in return for a superior false positive rate make Decision Trees the next best model for this case in my opinion.

7. Conclusion

In conclusion, I believe that the goal of this project was achieved with a model accuracy of 85%. Referring to our anticipated challenges mentioned in the introduction of this report, I believe that the first two challenges were accounted for by incorporating genre and decade into our models. However, the third was not, neither 'artist' nor 'album' was a feature incorporated into our models. Moreover, a track may have the technical audio features required to be a 'hit' but the artist or record label may not have the marketing strategy necessary for it to achieve that.

Further modelling analysis can be performed incorporating artist, album, and possibly record label. Furthermore, additional models can be assessed in an effort to develop more accurate predictive models.

Regardless, the results so far are significant enough to encourage further developments in the understanding of herd music preference with respect to scientifically measured audio data. This can lead to implications regarding the marketing of music from a platform, record label, and a musician's perspective.

8. References

Spotify Technology S.A.
Spotify Q2 Press Release 2021, page 1.
2021.

Spotify AB.
Spotify for Developers.
2021.

Christopher El Khouri.
Music Taste Across Decades: Spotify Audio Feature Analysis of 33,354 Musical Pieces.
2020.

Skipper Seabold and Josef Perktold.
statsmodels: Econometric and statistical modeling with python.
In *Proceedings of the 9th Python in Science Conference*,
pages 92-96, 2010.

Simon Colton.
Lecture 6: Decision Tree Learning.
Artificial Intelligence in Computational Art and Design, School of Interactive Arts and Technology.
Simon Fraser University, British Columbia, Canada, 2006.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay.
Scikit-learn: Machine Learning in Python.
Journal of Machine Learning Research, 12(85):2825−2830, 2011.

Thomas Cover and Peter Hart.
Nearest neighbor pattern classification
IEEE Transactions on Information Theory, 13(1):21-27, 1967.

Yadnesh.
k-Nearest Neighbors in Machine Learning (k-NN).
2020.

Farooq Ansari.
The Spotify Hit Predictor Dataset (1960-2019).
2020.