DSC 478 Final Project

Spotify Data Analysis

Chien Lin Yang

November 16, 2022

Summary

In this project, I obtained a large sample of tracks that were scraped from this API via Kaggle (Prediction of music genre | Kaggle), and I hope to use this sample to perform my data analysis that could resemble those used by modern digital applications. I do data analysis by using classification, clustering, and regression. Using machine learning, I hope to understand what popular music looks like and predict whether a particular style will be popular in the future.

For classification, in addition to basic approaches such as KNN, decision trees, Naïve Bayes, and Linear discriminant analysis (LDA), to also try 4 ensemble methods such as Random Forest, AdaBoost, Gradient Boosted Decision Trees (GBDT), and Bagging. The hyperparameters of these models have also been optimized through various experiments. I also perform unsupervised machine learning such as cluster analysis. In addition, I also try to do clustering using the reduced dimensional data after PCA and compare it with the original clustering. In addition to common regression analysis, I also use feature selection to try to reduce the dimensionality to achieve the purpose of optimizing the model. Furthermore, in the course of parameter optimization for the model, I compare the performance of the model on the training data (with a specific set of hyper-parameter values) to its performance on cross-validation. Since complex models (lots of parameters) are often prone to overfitting, overfitting can be reduced by imposing a constraint on the overall magnitude of the parameters (i.e., by including coefficients as part of the optimization process). Therefore, I also use Regularization-Based Models, such as Ridge, Lasso, Stochastic Gradient Descent Regressor, and Elastic Net.

Overall, the models built by Random Forest and Ada Boost (best combination of parameters) have the best results in classification, with an accuracy of up to 83% and 85%, and they also solve the overfitting problems. In addition, by observing the important features, we can find that “music_genre_Rap”, “music_genre_Rock”, “music_genre_Anime”, and “Instrumentalness” are the most important features, accounting for 53%. In Clustering and Regression, after PCA and Feature Selection to reduce the dimensionality, the performance of the model is also improved.

Introduction

Over the past couple of decades, music streaming services have become the norm for casual listening while traditional methods of sharing and collection like cassettes, CDs, and even mp3s have significantly faded in popularity. Streaming platforms have not only accelerated the digitalization and accessibility of music but also are at the forefront of analyzing user listening behaviors and creating a repertoire of advanced predictive and clustering algorithms and applications that serve as digital concierges for their user bases. These algorithms run daily and deliver personalized recommendations and suggestions that are customized per user, allowing them to explore based on their tastes and revisit their collections without having to spend hours digging through bins of LPs at record stores like the DJs of yore. Spotify is one of the most popular of these platforms and offers a free public API that developers all over the world can use to access the technical data of Spotify's entire music library. I obtained a large sample of tracks that were scraped from this API via Kaggle (Prediction of music genre | Kaggle), and I hoped to use this sample to perform my data analysis that could resemble those used by modern digital applications.

The data contains about 50,000 tracks and 18 variables, including 11 numeric variables, 6 categorical variables, and 1 date time.

Name	Data Type	Description
instance_id	Categorical	Instance ID.
artist_name	Categorical	The name of artist.
track_name	Categorical	The name of track.
popularity	Numerical	A measure on an integer scale from 0-100 on popularity of the track.
acousticness	Numerical	A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
danceability	Numerical	Describes from 0 to 1 how suitable a track is for dancing.
duration_ms	Numerical	The duration of the track in milliseconds.
energy	Numerical	Measure from 0.0 to 1.0 of the perceptual measure of intensity and activity.
instrumentalness	Numerical	Predictive measure from 0.0 to 1.0 on whether a track contains no vocals.
key	Categorical	The key the track is in.
liveness	Numerical	A detection measure from 0 to 1 on the presence of an audience in the track.
loudness	Numerical	The overall loudness of the track in decibels.

mode	Categorical	The modality of the track (major or minor).
speechiness	Numerical	Detects the presence of spoken words from 0 to 1.
tempo	Numerical	Estimated tempo of the track in beats-per-minute.
obtained_date	Date time	Obtained date.
valence	Numerical	A measure from 0 to 1 describing the musical positiveness.
music_genre	Categorical	The 10 genres are 'Electronic', 'Anime', 'Jazz', 'Alternative', 'Country', 'Rap', 'Blues', 'Rock', 'Classical', 'Hip-Hop'.

Data Analysis

1. Data Preprocessing

In the Knowledge Discovery Process (KDD Process), data preprocessing is important because in real world applications data can be inconsistent, incomplete, and noisy. Since this dataset has missing data, I use data cleansing to handle these missing data.

First, this dataset has 14 features and 50,005 samples by a simple shape function. Then, I used info() and isnull() to find the data with null values. Since these null values are null in every row and column and only 5 rows, I removed these 5 records. Second, I notice that the "tempo" attribute has the special symbol “?” exists, and there are 4,980 records. Since it accounts for 10% of the total data set, it cannot be removed directly. I use the mean value of "tempo" to fill these records so that the structure of the data will not be broken. In music-related fields, key and mode are observed together, so I merge these two attributes in one column, called “keyMode”.

2. Data Exploration and Visualization

Using data exploration to understand what is happening is important throughout the pipeline. I consider the distributions of each variable and some of the relationships between pairs of variables. Also, data visualization presents the characteristics of these variables.

For the numerical attributes, I used bot plots and histograms. We can see that the “instrumentalness”, “liveness”, “speechiness”, “loudness”, and “duration_ms” are relatively widely distributed, while “Popularity” and “danceability” have normal distributions. For the categorical attributes, we can see from the bar charts that each genre has 5,000 records, and the main key and mode are “G Major”, “C Major”, “D Major”, and “C# Major”. In the correlation

analysis, we can also observe that “acousticness” has a high negative correlation with “energy” and “loudness”, which are 79% and 73% respectively. In addition, “loudness” has an 84% positive correlation with “energy”, while “loudness” has a 53% negative correlation with “instrumentalness”.

On the other hand, Since the target attribute “popularity” is a numeric variable, I use the data discretization method to make it a categorical variable called “popClass”. I use the mean value of “popularity” to distinguish between “Popular” label and “Unpopular” label. I also observe “popClass” and “music_genre” with bar charts and found that the most popular ones are Rock, Rap, and Hip-Hop.

3. Data Preparation

First, I separate the target attribute and the attributes used for model training. Second, I create the dummy variables because there are categorical variables. Then, I divide the data into randomized training and test partitions. Finally, I perform min-max normalization to rescale numeric attributes, and this would be important for methods such as KNN that relies on computing distance or similarities among vectors, but it is not necessary for other methods such as classification tree learning.

4. Classification

For classification, in addition to basic approaches such as KNN, decision trees, Naïve Bayes, and Linear discriminant analysis (LDA), to also try 4 ensemble methods such as Random Forest, AdaBoost, Gradient Boosted Decision Trees (GBDT), and Bagging. The hyperparameters of these models have also been optimized through various experiments.

a. KNN

In the KNN method, experiment with different values of K (say from 5 to 100) and the weight parameter (i.e., with or without distance weighting) to see if we can improve accuracy of the KNN classifier. Overall, there is no significant difference between distance weighting and uniform weighting, but I choose the distance weighting with the highest correct rate(=0.8487). Next, using "distance" weights, compare the accuracy of the KNN classifier across the different values of K on the training and the test data to check for the potential overfitting or underfitting. I find that the highest accuracy is obtained with K=93. Then I use three evaluation methods to evaluate KNN, including the classification report, the confusion matrix (visualize it using Matplotlib), and the average accuracy scores. Overall, the results are good, with a prediction of 85% for the test data. However, there might be overfitting because the prediction for the

training data is as high as 99%.

b. Decision Tree

In the Decision Tree method, due to significant overfitting with the default parameters, the difference between the training and test data is 0.2. Therefore, we are going to use Pruning the tree to help in reducing overfitting. I explore various decision tree parameters and the use of cross-validation for evaluation. The parameters of the final model are criterion='entropy', min_samples_leaf=3, max_depth=4, and the accuracy of the training data is 0.8457, and the accuracy of the training data after cross-validation is also 0.8457, while the accuracy of the test data is 0.8155. Overall, the accuracy is very good and overfitting is reduced. I present the final model with a visual decision tree , and I also use the bar charts with important features to notice that “music_genre_Rock”, “music_genre_Rap”, “music_genre_Hip-Hop”, and “music_genre_Alternative”.

c. Naive Bayes (Gaussian) and Linear discriminant analysis (LDA)

There is no particular overfitting with these two methods, but LDA is better with 84% accuracy, and Naive Bayes is only 60%.

d. Random Forest

I use the "calc_params" function to explore the impact of individual parameters using cross-validation, including n_estimators, min_samples_leaf, and max_depth. Final model with these parameters is n_estimators=35, min_samples_leaf=9, max_depth=3, and the accuracy of the training data is 0.8294, and the accuracy of the test data is 0.8305. Also, the bar charts with important features show that “music_genre_Rock”, “music_genre_Rap”, and “music_genre_Anime” are the most important features.

e. Ada Boost

I use grid search to explore the parameter space more systematically, and the parameters of the final Ada Boost model are n_estimators=75 and learning_rate=0.9. The accuracy of the training data is 0.8465 and the accuracy of the test data is 0.8469.

f. Gradient Boosted Decision Trees (GBDT) and Bagging

The performances of both ensemble methods are similar, but GBDT has reduced overfitting.

In general, Random Forest and Ada Boost of the ensemble methods have the best performances in classification, not only with 83%-85% accuracy but also without overfitting.

5. Clustering

In the cluster analysis of unsupervised learning, I use the kmeans method. I compared the results obtained with and without PCA and used three evaluation methods, Silhouette, Completeness, and Homogeneity. Overall, there was no improvement in performance after PCA, but its mean Silhouette value improved from 0.1374 to 0.1420.

6. Regression

In the regression analysis, I use the numerical variable “popularity” as the target attribute and evaluate the model with RMSE and MAE. In addition to common regression analysis, I also use feature selection to try to reduce the dimensionality to achieve the purpose of optimizing the model. However, the result of feature selection is the same as the original regression, both with MAE=0.072. Furthermore, in the course of parameter optimization for the model, I compare the performance of the model on the training data (with a specific set of hyper-parameter values) to its performance on cross-validation. The cross-validation is performed using partitions of the training data, and I use one approach to perform cross-validation using the KFold function in sklearn.model_selection. However, the 10-fold cross-validation does not get the better performance.

Since complex models (lots of parameters) are often prone to overfitting, overfitting can be reduced by imposing a constraint on the overall magnitude of the parameters (i.e., by including coefficients as part of the optimization process). Therefore, I also use Regularization-Based Models, such as Ridge, Lasso, Stochastic Gradient Descent Regressor, and Elastic Net. However, because there is no particularly obvious overfitting problem, so each Regularization-Based Model does not particularly reduce overfitting.

7. Conclusion

Among the different machine learning methods, the ensemble methods of Random Forest and Ada Boost have the best results with an accuracy of up to 83% and 85%, and they also solve the overfitting problems. I think that it might be because ensemble methods use a combination of models to increase accuracy. Also, Random Forest is one of the most accurate learning algorithms available for most data sets, and it can handle lots of variables without variable deletion.