Music Genre Classification

An Exploratory Research Project for CS490

Created By

Harrison Margalotti, Undergrad Class of 2020

and

Benjamin Cordova, Undergrad Class of 2020

Ithaca College

2019

Abstract

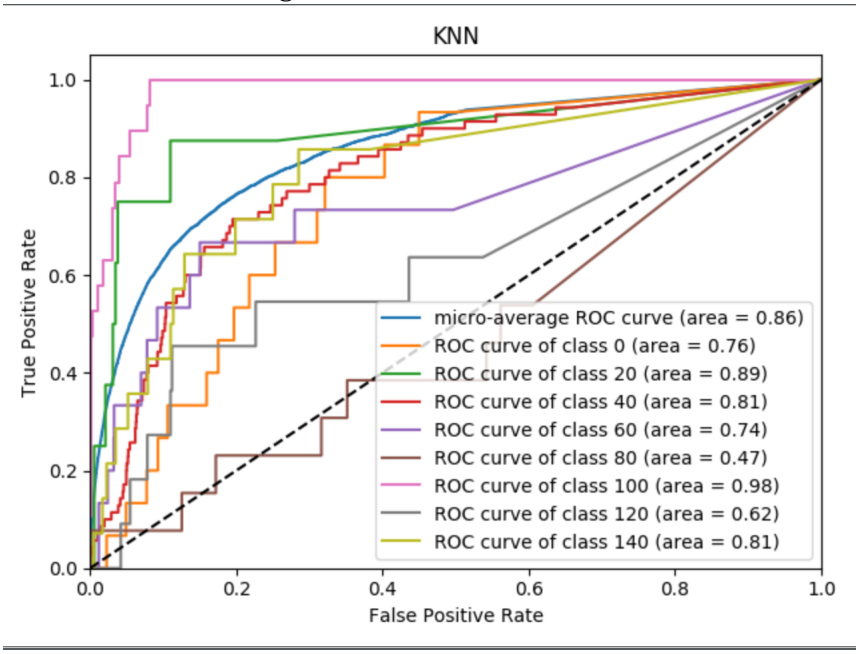
This machine learning project involved the classification of music. More specifically, the task at hand was to sort through thousands of songs, spreading across all genres, and create a system that could correctly identify and categorize any new song data it was fed through audio signal analysis. This is accomplished by creating a pipeline for the task at hand using classifiers from SciKit learn and then adjusting it to the specific situation. Through a process of research, data collection, and internal analysis, the most successful algorithm was developed. The best algorithm was classifying each genre 4 times and creating a mapping to the classifier that performed the best on internal metrics. Once the best classifier per genre was established, the calibrated algorithm ran on test data and internal metrics showed an auc_roc score of 0.887.

Setup

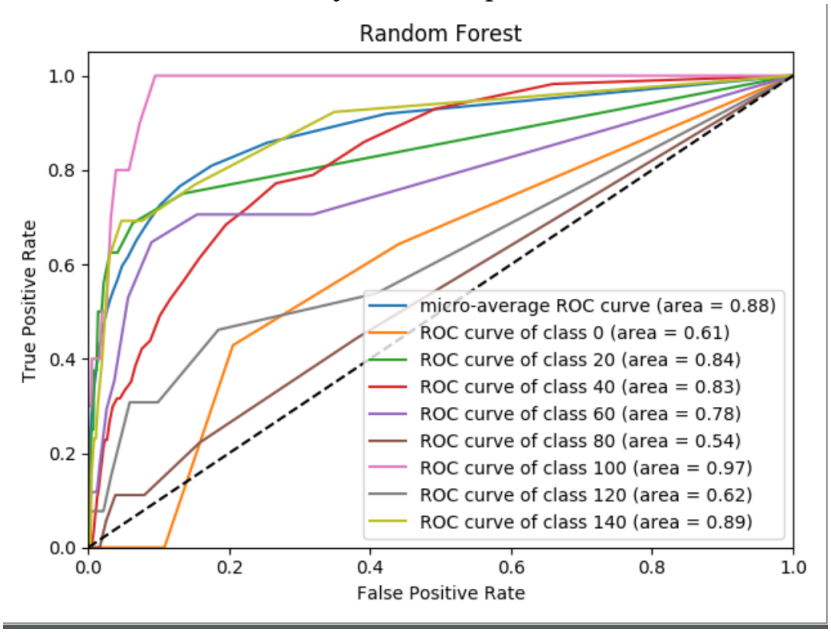
When first presented with the project problem and criteria, the most popular route appeared to be to start with Professor Turnbulls suggestion of logistic regression. After fully analyzing the situation at hand, it was decided that a problem of this level of realism must have a more complex, and efficient, solution than a simple logistic regression algorithm. The knowledge that Comp 490 instilled created the confidence to be able to responsibly research and choose another possible algorithm. This turned the direction of the project into online research, combing through research papers and academic articles for other possible solutions not specifically covered in class. After collecting a list of promising algorithms, the next step was to implement each of them and observe their performance as well as their potential for the project. The library used to implement the classifiers is called SciKit learn. All of the classifiers used throughout this experiment share the same API allowing the project to have low coupling and high cohesion. The metric used for internal testing is the area under the Receiver Operating Characteristic curve. The visuals shown throughout express the range of roc curves across genres as well as the micro-average of the rocs.

Methods

The first algorithm that was chosen for testing was "K nearest neighbors". This is a relatively simple algorithm but it had a few structural benefits that seemed promising. The basic principle of KNN is that it will look at clusters of data points, and select the nearest cluster based on distance to the new, uncategorized data point. Format wise, this seemed like it would transfer well onto the project. The graph produced would have a sufficient amount of training data points from the data given. Barring song data points that are extremely close to multiple categories, it seemed that KNN would have a high rate of success with more easily identifiable songs. Small adjustments would be able to account for the harder to classify data points. KNN was not a complete failure, but was not very successful. After classifying all genres with KNN, an auc_roc score of 0.79 was produced. Some genres were very easy for this classifier to identify while others were very difficult (statistically random). This algorithm was like not very successful due to the fact that some genres had very few songs in them, making them unlikely neighbors. Additionally, the number of neighbors used was 147 which correlates to one neighbor per genre. The result of this was an algorithm which took a very long time to run, and struggled due to so many possible neighbors being considered. Below is the range of ROC curves produced when all genres were classified with KNN.

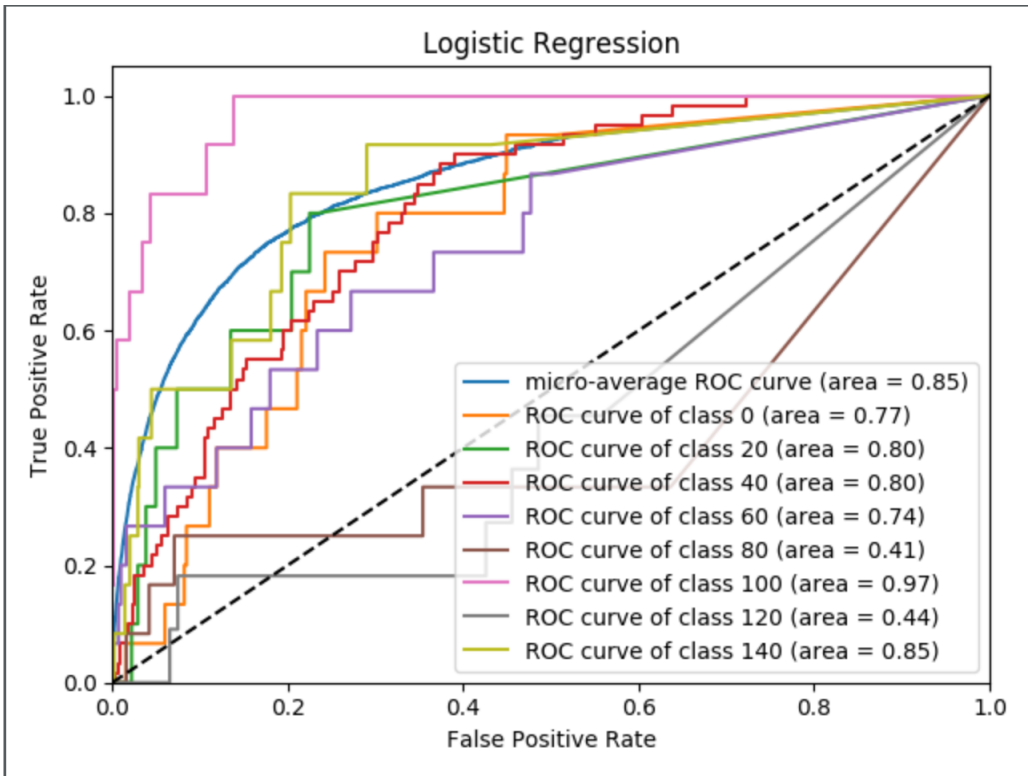


The second algorithm we tested was "Random Forest". This was described as a "flexible machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time." Another promising feature of RF was that it can be applied to both classification and regression taks. The process of RF is creating a tree system of comparisons. It will gather suggestions for classification from each node in a tree format. From there, it will taking the highest percentage classification choice. When a large amount of sample data is available, this algorithm should work very well, providing more comparisons, and in theory a higher accuracy decision. This list of benefits is why this algorithm was chosen second. Classifying every genre with the random forest classifier produced an auc_roc score of .79. This is a good start, but not the desired result. One downside is that this algorithm quickly scales in runtime complexity it generates X number of decision trees that each have a depth of N. Larger trees allow for more accurate predictions but take a very long time to process. Perhaps, this is the reason as to why this attempt was unsuccessful.



After two fairly unsuccessful tests, the third algorithm we chose to use seemed very promising for finding an innovative and efficient solution. This algorithm was called "Tree-based Pipeline Optimization Tool", or TPOT for short. This was described as a new approach to machine learning problems.

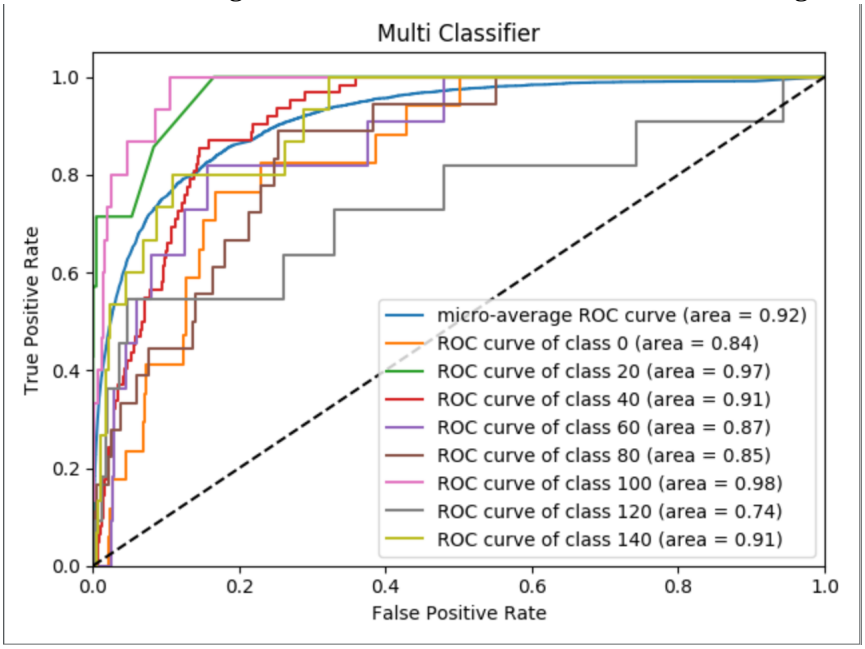
The idea is that there are many possible choices for which algorithm to use for any given problem. TPOT is designed to "optimize a series of feature preprocessors and machine learning models with the goal of maximizing classification accuracy on a supervised classification task". This seemed like the perfect solution to the problem at hand. This would produce for us a pipeline of the most efficient strategies for classifying the data. TPOT ran for four days using an intel 8700K with 12 threads at 4.9Ghz for processing along with 16gb of memory. The final result was a pipeline utilizing the Naive Bayes classifier. This pipeline which was thought to be the best according to the genetic learning algorithm showed an actual auc_roc score of a 0.75. TPOT is an impressive algorithm, but one downside to genetic learning besides the massive processing power required is the inability to perform well on a variety of problems. After attempting three different "outside" algorithms with little success, it was decided to return to square one and begin with basic logistic regression. This decision was made considering the amount of time left to complete and submit the project. The first step taken towards tweaking the basic logistic regression algorithm was to apply Principal Component Analysis. PCA is a dimensionality reduction procedure which is designed to compare all of your set features, evaluate them individually and in relation to the overall dataset, then choose only the critical features needed to identify the dataset entirely. Reducing the number of features from 76 to the 70 most important showed no a decrease in performance by 0.01. Reductions to 60 features, 25 features, and 12 features followed a trend in decreases in performance. Inversely, the next feature manipulation tactic employed was polynomial expansion. Reduction of the number of features proved to be detrimental so the next logical test was increasing the number of features. This proved to be even more detrimental, decreasing the auc_roc score by 0.03.



Results Analysis

Analyzing the results from the range of rocs from KNN shows that it had genres score as high as .98, and genres scoring as low as .47 which is worse than random. This shows that the are subsets of the data, or specific genres, that is shows true promise for, while others it did not. While it did not score very well overall, it was not to be discarded. Likewise, RF had scores ranging from 0.54 to 0.97. More interestingly, where KNN scored a low 0.47 which was worse than guessing, RF was able to bring this score to 0.54. That is not coincidental, it is progress in its infancy. Seeing an algorithm be able to make sense of data where another did worse than guessing really highlights how data makes different algorithms perform better. Looking at genre 80, which scored 0.54 with RF, and 0.47 with KNN, it scored 0.41 with logistic regression, a new low between the classifiers. While this was not the desired result, it is more data, allowing the discovery of correlation between data and the similarities that KNN and logistic regression share. This analysis is what guided the project into its final phase of using different

classifiers for different genres. Each genre was classified using 4 different classifiers: SVM's, Logistic Regression, KNN, and RF. The highest scoring (according to internal metrics) per genre was the classifier that would be used on that genre when predicting new test data. After classifying each genre four times, a mapping from genre to ideal classifier was created which allowed for a calibrated version of the model to be created where it would pick the best classifier to use per genre without having to re-classify and compare score which would be unable to work on test data. The results from internal testing with the calibrated model were the following:



Using the multi-classifier method limited the range of scores produced to only range from 0.74-0.98. Genre 80, which had a previous max score of 0.54 when using a single classifier across all data, has now jumped up to 0.85, an increase of 0.31. The visual above shows that this is a very promising algorithm and a great final implementation building upon the other tested algorithms and what was learned between all of them. To continue improvement, more classifiers would be likely to help as the algorithm would be able to target the data more specifically. The concern with this would be making sure to take precautions to avoid overfitting the data which would only harm the algorithm. This algorithm was not submitted to the competition to receive a true auc_roc score, but internal metrics which

have been tuned according to the true scores showed an auc_roc score of 0.887 for this algorithm.

Conclusion

As a first real-world problem, developing a machine learning algorithm for music genre classification, similar to tasks that tech giants Pandora or Spotify may encounter; this project was very successful due to the translation of the skills which were learned during Comp 490. Due to the realism of the task in the world, it was decided that there would be extensive research and work done on this specific type of classification problem. This is why the development team began by researching the topic further, looking for an algorithm that is specifically designed for music classification, or something very similar. Research resulted in 3 strong contenders for this project. These algorithms were K nearest neighbors, Random Forest, Tree-based Pipeline Optimization Tool, and logistic regression which was decided to be held off on implementing. Studies showed that each was able to perform very well in a variety of situations, which was true and some were able to classify specific genres very well where others could not. After analysis over why certain classifiers struggled with data, and identifying trends amongst the data, it was discovered that the most optimal solution was to allow the algorithms to work together as a team to accomplish the task; a far more creative solution that felt both satisfying to see perform well, as well as being very fitting for a real world task. This project was not perfect, however. K nearest neighbors is structured in a way that could be easily applied to the given task, and likely to succeed because of what appeared to be a large data set, or so it seemed. Random Forest will take the highest "voted" category to place the test data in. Again, because of the decently large data set, it was hoped that this algorithm would perform well. The final algorithm which was tested was TPOT. The hope was that this genetic learning algorithm would produce a pipeline of the most efficient path to take. While the "one perfect algorithm" that was sought out was never discovered, there were many strong benefits to expanding the search to strategies not taught in class. Comp 490 was a great introduction to machine learning that allowed students to become competent in machine learning algorithms. Due to this strong foundation

which was developed, students are knowledgeable enough to apply what was learned to make informed decisions regarding other solutions. This led to a much larger world of machine learning, and introduced the team to three significant algorithms that were previously unknown to the members. It has demonstrated that with the base of information received, the expansion, and progression of knowledge and skills through self exploration and research is insurmountable.

References

J. M. Keller, M. R. Gray and J. A. Givens, "A fuzzy K-nearest neighbor algorithm," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-15, no. 4, pp. 580-585, July-Aug. 1985. doi: 10.1109/TSMC.1985.6313426 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6313426&isnumber=6313410

Olson, Randal S., and Jason H. Moore. "TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning." PMLR, 4 Dec. 2016, proceedings.mlr.press/v64/olson_tpot_2016.html.

"A Comparative Study on Content-Based Music Genre Classification." ACM Digital Library, ACM, dl.acm.org/citation.cfm?id=860487.l