Nate Boldt
COMP479
Professor Dligach
10/24/2021

Homework 3 Report

This report will detail the steps, observations, and experiments taken for homework 3. First, we need some data to analyze – the data chosen for this is a set of music tracks with 16 features that have the label of music genre. Each assignment requirement has a (#) in it to indicate where in the report each is addressed. The models all performed quite poorly on this data set, which means I will find a different data set for the project.

To start, the pre-processing performed after splitting the data into training, development, and test sets involved converting string features into float representations and normalizing the data. Additionally, filling in missing features with a '0' value and separating the features from the labels.

In attempting to fit a logistic regression classifier to the music genre data, I started by using the default values for everything. With these defaults, the model did not converge, and I could not then use it for any predictions. Tampering with the number of iterations was then my next objective – upping it to 500, then 1000, then 1500 – all these values failed to converge. However, at 2000 iterations the model converges and is able to make predictions. This logistic regression model had poor performance on the development set – the accuracy count was 3%(1). The F1 score was not calculated for this multi-class model, because the purpose for this assignment is to increase overall model performance as opposed to performance for specific classes. In a real world example, I would most likely choose the F1 score because this data set is partially imbalanced.

To increase the performance of this model in classification, next the model was run at 3000 iterations. Originally, I had thought that more iterations would create a better performance, but the output accuracy was exactly the same – this makes sense as the iteration parameter is just *how many attempts* of training we perform until convergence, so 'extra' iterations aren't going to improve the model. After this, I tweaked the CV value from the default of 5 to 10 to get the 10 k-fold increased data training. This tweak required more iterations in order to converge up to 2500, but did not increase the accuracy at all(2).

After the accuracy still failed to increase, I explored using the SVM model. The setup is mostly the same with preprocessing using the same training data / development data to test. The SVM model did have an improvement in performance, but not by much – clocking in at 3.7% accuracy.

Comparing these sci-kit built in models to my implementation of KNN shows the KNN has advantages over the logistic regression and SVM attempts – with accuracy boosting to 4.4% using default values. This means that the data could show slight clumping patterns, in that similar songs would have similar traits. In tweaking the K value, we see valuable performance improvements by up to 5% for a K of 10 and nearly 6% with a K of 20(3).

Comparing all of these models to the base line 'dummy classifier' from ScitKit shows that the models are indeed preforming very poorly. As you can see below, the default run from the dummy classifier has the best accuracy of any of the models attempted, which is achieved also by the 'most frequent' setting. This proves that the data is likely imbalanced, especially as

we cycle through different classes in the ‘constant setting’ – a constant setting of class 3 vs class 10 is almost a full 6% difference(4). The only redeeming quality of the ScitKit models and KNN implementation is that it beats out the ‘uniform’ setting of making random guesses on labels.

Default	6.2%
Stratified	3.1%
Most_Frequent	6.2%
Uniform	1.9%
Constant, #10	6.2%
Constant, #3	0.5%

In a final comparison using the test data shows a 4% accuracy on the SVM model, a 6% accuracy on the dummy classifier, and the KNN has a 5.5%. These ratings show either very poor performing models or perhaps more likely a data set without many discernable patterns.