CSC475 / SENG480B Music Retrieval Techniques Assignment 4 Fall 2020

J. Shier

October 28, 2020

1 Introduction

In this assignment we will explore machine learning topics including supervised learning for genre classification with audio features as well as with song lyrics. There are three levels of engagement: **Minimum**, **Expected**, **Advanced**. **Minimum** is the least amount of work you can do to keep up with the course. If you are not able to complete the minimum work suggested then this course is probably not for you. **Expected** is what a typical student of the course should work on - the goal is to provide a solid foundation and understanding but not go deeper into more interesting questions and research topics. **Advanced** is what students who are particularly interested in the topic of MIR, students who want a high grade, students interested in graduate school, and graduate students should work on. Each level of engagement includes the previous one so a student who is very interested in the topic should complete all three levels of engagement (*Minimum*, *Expected*, *Advanced*).

The assignment is worth 10% of the final grade. There is some variance in the amount of time each question probably will require. Please provide your answers as a single Jupyter Python notebook (.ipynb) uploaded through the BrightSpace website for the course. Also please answer the questions in order and explicitly mention if you have decided to skip a question. For the questions that do not require programming use the ability to write Markdown in Jupyter notebooks for your answer. You are also welcome to do the assignment in any other programming language/framework. If you do so please include a PDF file with plots and figures for questions that require them, as well as answers for questions that don't require coding. Also include the code itself with instructions on how to run it in a README file.

2 Problems

The coding problems mention specific libraries/frameworks for Python that can help you with implementation. However you should be able to find corresponding libraries or code for most programming languages if you want to try using a different programming language or environment.

1. This question will build on the audio feature extraction using spectral centroid question from assignment 3. We'll perform experiments on three different genres: classical, disco, and reggae. There are 300 audio files in total, 100 for each genre. These audio files are available in the GTZAN folder in assignment resources. The assignment template contains a solution to the last question from assignment 3 (updated with new genres), which computes the mean and standard deviation of the spectral centroid for each track and plots them on a scatter plot. We'll use these results for audio classification. If you're not using Jupyter notebooks you will have to copy/translate this code into the language or environment that you are using.

    - **a)** Use sckit-learn to report the 10-fold cross-validation classification accuracy for a linear support vector machine and a naive bayes classifier trained on the two features calculated in the Jupyter template (mean centroid and std centroid) to predict the three genres. Show the confusion matrix for each case. **(Minimum: 1 point)**
    - **b)** Compute the MFCCs for each recording using the default settings of librosa. Then summarize the entire recording by taking the mean of the MFCCs across the recording as well as the mean and standard deviation across each recording. The resulting configurations will be just the mean (20 features per recording) and the mean and std (40 features per recording). Report on the 10-fold cross-validation classification accuracy and confusion matrix for these two configurations using the linear support vector machine and naive bayes classifier. **(Minimum: 1 point)**
    - **c)** Use t-SNE[1] to reduce the dimensionality of the 300 by 40 feature matrix of mean and std mfccs to a 300 by 2 feature matrix. Visualize the corresponding scatter plot with coloring of

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html`

the points based on genre. How does the visual separation/overlap compare to the scatter plot of just the centroid? Run the 10-fold cross-validation classification accuaracy and show a confusion matrix for the same configurations as the full feature set, but now using only the 2 dimensions returned from t-SNE. How much accuracy is lost compared results from the previous question? **(Expected: 1 point)**

- **d)** Let's forget about the genre labels and do some unsupervised learning. In this question we'll perform clustering on the two-dimensional results from t-SNE using K-Means[2]. Pretend that you forgot that the files in the dataset are actually from three different genres and experiment with a few different values for number of clusters. Plot the results of clustering using three different choices for number of clusters. Make sure to plot each sample colored by the label assigned to it by K-Means. What value for number of clusters gives you the best result in your opinion? How does your plot compare to the results from the previous question?

  Now, randomly select three points from two different clusters and play the associated audio files (6 in total). This is where keeping track of the audio file names during audio feature extraction would have been helpful (see code comment in the compute folder function above).

  Comment on the similarities between the audio files from the same clusters - do they sound like they are from the same genre? Does it make sense that they were clustered together?

  (Expected: 1 point)

2. This question will look at Naive Bayes classification with song lyrics. George covers material pretty much identical to this question in the first video of the Machine Learning for MIR Kadenze course, which you can use if you need to review this material before completing the questions.

   Our goal will be to build a simple Naive Bayes classifier for the MSD dataset, which uses lyrics to classify music into genres. More complicated approaches using term frequency and inverse document frequency weighting and many more words are possible but the basic

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans`

concepts are the same. The goal is to understand the whole process, so do not use existing machine learning packages but rather build the classifier from scratch.

We are going to use the musicXmatch[3] dataset which is a large collection of song lyrics in bag-of-words format for some of the tracks contained in the Million Song dataset (MSD). The corresponding genre annotations, for some of the song in the musicXmatch dataset, is provided by the MSD Allmusic Genre Dataset[4]. For the purpose of this course, in order to simplify the problem, we are going to use a reduced version of the musicXmatch dataset. Three genres are considered, namely: Rap, Pop Rock, and Country. The resulting genre annotated dataset is obtained by an intersection of musicXmatch and MAGD, where we select 1000 instances of each genre, such that the three classes are balanced and easy to handle. In addition, we also reduce the cardinality of the dictionary of words used for the bag-of words lyrics representation (originally equal to 5000), to the 10 best words for each genre. Intuitively, the best words are the most frequent words for a particular genre that are not frequent among all the genres.[5]

The resulting dictionary of words is

Genre										
rap	de	hood	ya	und	yall	ich	fuck	shit	yo	bitch
rock	end	wait	again	light	eye	noth	lie	fall	our	away
country	gone	good	night	blue	home	long	littl	well	heart	old

For answering this question we provide you with:

- **data.npy** - the three genres dataset (not binarized - you will need to binarize)
- **labels.npy** - the genre labels where Rap=12, Pop Rock=1, and Country=3

[3] `https://labrosa.ee.columbia.edu/millionsong/musixmatch`

[4] `http://www.ifs.tuwien.ac.at/mir/msd/partitions/msd-MAGD-genreAssignment.cls`

[5] The best genre words maximize the Term Frequency (TF) and Inverse Document Frequency (IDF) product. More details available at `https://en.wikipedia.org/wiki/Tf-idf`

- **dictionary.pickle** - the full 5000 words dictionary
- **words.npy** the 30 best word indexes with respect to the full dictionary
- **tracks.pickle** - the track IDs of songs used (not needed for assignment, but included for those interested).

This is available on BrightSpace in the data folder in the assignment resources download. You will need to use python pickle and numpy load to load the pickle and npy files respectively.

- **a)** Write code that calculates the probabilities for each dictionary word given the genre. For the purposes of this assignment we are considering only the tracks belonging to the three genres: Rap, Rock/Pop, Country. Use add-one additive smoothing[6] to handle the case that there is no instance of a particular word in a genre. **(Minimum 1pt)**
- **b)** One can consider the Naive Bayes classifier a generative model that can generate binary feature vectors using the associated probabilities from the training data. The idea is similar to how we do direct sampling in Bayesian Networks and depends on generating random number from a discrete distribution (the unifying underlying theme of this assignment question). Describe how you would generate random genre lyrics consisting solely of the words from the dictionary using your model. Code that and show 5 examples of randomly generated tracks for each of the three genres: Rap, Rock pop, and Country; each example should consist of a subset of the words in the dictionary. **(Minimum: 1 point)**
- **c)** Explain how these probability estimates can be combined to form a Naive Bayes classifier. Code it and calculate the classification accuracy and confusion matrix that you would obtain using the whole data set for both training and testing. Do not use any libraries such as scikit-learn but write the code directly. **(Expected: 2 point)**
- **d)** Read the Wikipedia page about cross-validation in statistics[7]. Calculate the classification accuracy and confusion matrix using the kfold cross-validation, where k = 10. Note that you will need

[6] `https://en.wikipedia.org/wiki/Additive_smoothing`
[7] `https://en.wikipedia.org/wiki/Cross-validation_(statistics)`

to generate your own splits. Do not use any libraries such as scikit-learn but write the code directly. **(Advanced: 2 points)**