ISTANBUL TECHNICAL UNIVERSITY ELECTRICAL-ELECTRONICS FACULTY

MUSIC GENRE CLASSIFICATION VIA MACHINE LEARNING

MACHINE LEARNING FOR SIGNAL PROCESSING PROJECT FINAL REPORT

Faik Yusuf Ayan 040150239

Instructor: Asst. Prof. Ender Mete EKŞİOĞLU

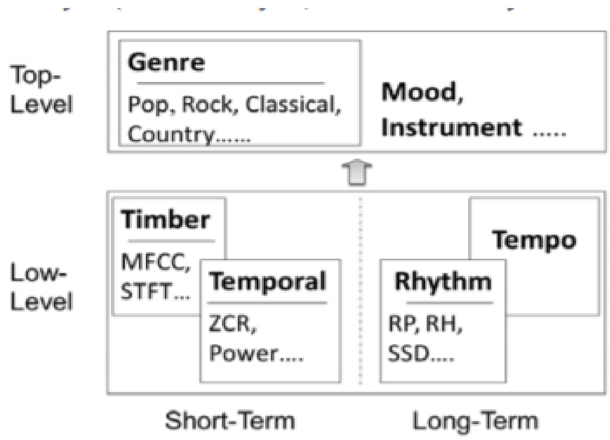
INTRODUCTION

Music genre prediction is one of the topics that digital music processing is interested in. Music styles are classified by a variety of parameters such as beat and timbre. Humans automatically categorize music types with the senses, but this separation is not straightforward for machines. To overcome this challenge, a discipline called MIR (Music Information Retrieval) was created. MIR includes research on the extraction of features from music signals to decide the music genre. Extracting the features relevant to the task from the audio is a very important step in many music information retrieval (MIR) appliances, and the choice of functions has a significant effect on performance. Over the last few decades, many features have been introduced and successfully applied to many diverse types of MIR systems.

Briefly, the aim of this work is to predict the genres of songs by using machine learning techniques. For this purpose, feature extraction is done by using signal processing techniques, then machine learning algorithms are applied to do a multiclass classification for music genres. I decided to choose 4 music genres that are blues, metal, country, and pop.

FEATURE EXTRACTION

Audio features can be mainly divided into two levels as top-level and low-level according to perspective of music understanding . The top level labels provide information on how listeners interpret and understand music using different genres, moods, instruments, etc. Low-level audio features can also be categorized into short-term and long-term features on the basis of their time scale. This figure characterizes audio features from different levels and perspectives.



Most of the features that have been proposed in the literatures are short-time timbre features, which only consider the immediate frequencies and extract the characteristics of the audio signal in a 10-30ms duration small sized window. Long-term features such as rhythm and beat features contain the structural information and normally extracted from the local windows on the large time-scale full song or a sound clip.

Audio data can be decoded and transformed into series of digital samples to represent the waveform. But this data cannot be used directly by machine learning algorithms because pattern matching algorithms cannot deal with such an amount of information. So, it is necessary to extract some features that describe the audio wave using a compact representation. Different features works well for different purposes and for classifying music genres the most useful features are listed below.

1-Zero Crossing Rate (ZCR): The ZCR measures the noisiness of the sound by computing the number of times the audio waveform crosses the zero axis per time unit. A zero crossing occurs when adjacent audio samples have different signs. The following equation shows the mathematical calculation of Time Domain Zero Crossings.

$$Z(i)= \frac{1}{2N} \sum_{n=0}^{N-1} |sgn[x_i(n)]-sgn[x_i(n-1)]|$$

2-Root Mean Square Energy (RMSE): The energy of a signal corresponds to the total magntiude of the signal. For audio signals, that roughly corresponds to how loud the signal is. The root-mean-square energy (RMSE) in a signal is defined as

$$\sqrt{\frac{1}{N} \sum_{n=1}^{N} |x(n)|^2}$$

3-Spectral Centroid: Spectral Centroid (SC) is commonly associated with the measure of the shape or brightness of a sound by calculating the weighted average frequency of every time frame. The spectral centroid is defined as the "center of gravity" of a Short Time Fourier Transform (STFT) using the Fourier transform's frequency and magnitude information. The following equation shows the calculation of Spectral Centroid.

$$C(i)= \frac{\sum_{N-1}^{k=0} k|Xi(k)|}{\sum_{N-1}^{k=0} |Xi(k)|}$$

4-Spectral Roll-off: Spectral roll-off point is defined as the boundary frequency where 85% of the energy distribution in the spectrum is below this point. A measure of the skewness of the spectral shape.

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n]$$

5-Mel-Frequency Cepstral Coefficients: Besides from being the most important audio feature MFCC are compact, short time descriptors of the spectral envelope audio feature set and typically computed for audio segments of 10-100ms. MFCC are one of the most popular set of features used in pattern recognition. Although this feature set is based on human perception analysis but after calculated features it may not be understood as human perception of rhythm, pitch, etc. 13 MFCC features is selected for the solution of genre specification. MFCCs are commonly derived as follows:

- Take the Fourier transfrom of a windowed signal.
- Map the powers of the spectrum obtained above onto the Mel scale, using triangular overlapping windows.
- Take the logs of the powers at each of the Mel frequencies.
- Take the discrete cosine transform of the list of Mel log powers, as if it were a signal.
- The MFCCs are the amplitudes of the resulting spectrum.

```
notefolder='genres\blues\';
listname=dir (fullfile([notefolder,'*.wav']));
coeffs1=zeros(length(listname),186);

for k=1:length(listname)
    file_name=strcat(notefolder,listname(k).name);
    [x, fs]=audioread(file_name);

    aFE=audioFeatureExtractor('SpectralDescriptorInput',"melSpectrum",'SampleRate',fs,...
        "mfcc",true,"spectralCentroid",true,"spectralRolloffPoint",true);


    z=ZCR(x);%zero crossing rate
    rms=sqrt(mean(x.^2, 1));%rmse value
    features=extract(aFE,x);%mfcc and spectral features
    kovaryans=cov(features(:,1:13));
    coeffs1(k,:)=[mean(features,1) reshape(kovaryans,1,[]) z rms];

end
```

Above figure of our code illustrates feature extraction of one music genre consist of 100 songs. Each genre has its own folder. Since there are 4 different genres this process is repeated 4 times for blues, metal, country, and pop.

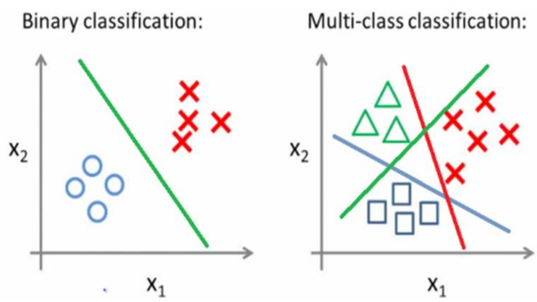
```
%%
X=[coeffs1;coeffs2;coeffs3;coeffs4]; %combining feature vectors
y=[1*ones(100,1);2*ones(100,1);3*ones(100,1);4*ones(100,1)]; %labeling

%split into train and test
indices=randperm(400);
Xtrain=X(indices(1:360),:);
Xtest=X(indices(361:end),:);
ytrain=y(indices(1:360),:);
ytest=y(indices(361:end),:);

```

For each song 186 features are extracted and combined. Size of feature matrix (X matrix) is 400x186. Once extraction part is done, I moved on to labeling and splitting data as train and test sets. At that point I was ready to train our model and test it. Data is splitted into %90 train and %10 test since I have small data set.

METHODS AND RESULTS

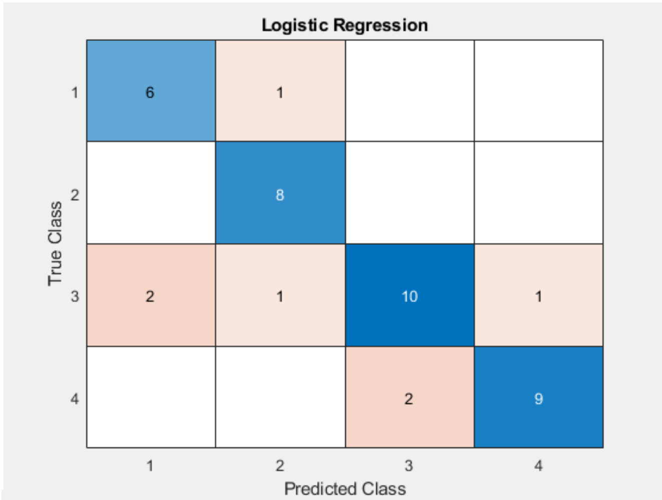
2 different supervised classification algorithms are used for our classification problem.

1-One vs All Logistic Regression: It is a form of logistic regresssion used to predict a variable have more than 2 classes. It is built upon existing model and turned it into a multi-class classifier. Is is a method which involves training N distinct binary classifiers (N=4 for our case). Then those N classifiers are used as demonstrated:



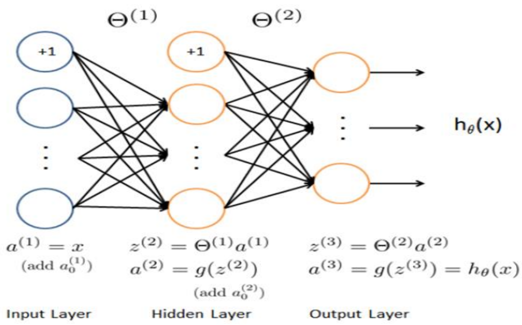
```
%%
%OnevsAll Logistic Regression with Regularization

num_labels=4;
lambda1=1;%for regularization

[all_theta] = oneVsAll(Xtrain, ytrain, num_labels, lambda1);
%iterasyon sayısı değiştikçe sonuç değişiyor

predLR1 = predictOneVsAll(all_theta, Xtest);
fprintf('\nTest Accuracy: %f\n', mean(double(predLR1 == ytest)) * 100);

C2=confusionmat(ytest,predLR1);
cm2=confusionchart(C2);
cm2.Title='Logistic Regression';
```

```
Iteration   4989 | Cost: 1.006330e-01
Iteration   4990 | Cost: 1.006330e-01
Iteration   4991 | Cost: 1.006330e-01
Iteration   4992 | Cost: 1.006330e-01
Iteration   4993 | Cost: 1.006330e-01
Iteration   4994 | Cost: 1.006329e-01
Iteration   4995 | Cost: 1.006329e-01
Iteration   4996 | Cost: 1.006329e-01
Iteration   4997 | Cost: 1.006329e-01
Iteration   4998 | Cost: 1.006329e-01
Iteration   4999 | Cost: 1.006329e-01
Iteration   5000 | Cost: 1.006329e-01


Test Accuracy: 82.500000
```



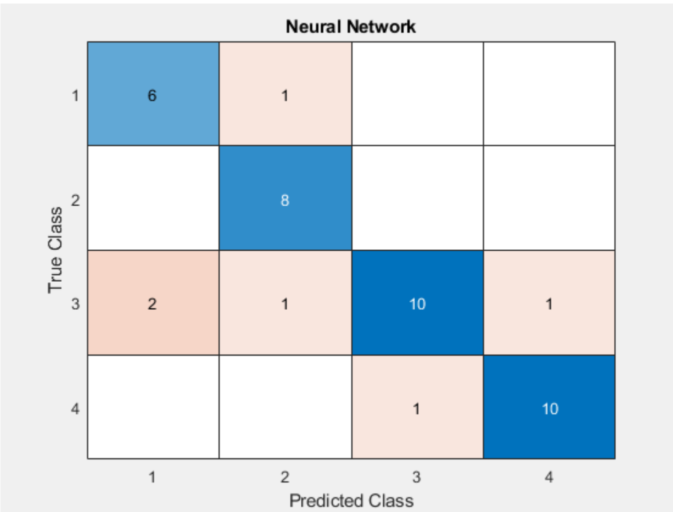
OnevsAll Logistic Regression model gave us pretty good results as it can seen from confusion matrix and numerical evaluation. Optimization function fmincg is used to get optimum theta values by reducing LRcostfunction. Sigmoid function is used in costfunction for classification purposes.

2-Neural Networks: It is a dynamic structure that seek to replicate the creation of classification rules by the human brain. A neural network consists of various neuron layers, each layer receiving inputs from previous layers and transferring outputs to next layers. The manner in which each layer output becomes the input for the next layer depends on the weight given to that particular element, which depends on the cost function. The cost function alters the internal mechanics of the network such as the weights based on the information provided by the cost function, until the cost function is minimized.



```
%%
%Neural Networks

input_layer_size=186;
hidden_layer_size=20;
num_labels=4;

initial_Theta1 = randInitializeWeights(input_layer_size, hidden_layer_size);
initial_Theta2 = randInitializeWeights(hidden_layer_size, num_labels);

initial_nn_params = [initial_Theta1(:) ; initial_Theta2(:)];

options = optimset('MaxIter', 20000);%we can try different iteration number
lambda = 1;%we can try different lambda values

costFunction = @(p) nnCostFunction(p, ...
                                   input_layer_size, ...
                                   hidden_layer_size, ...
                                   num_labels, Xtrain, ytrain, lambda);

[nn_params, cost] = fmincg(costFunction, initial_nn_params, options);

Theta1 = reshape(nn_params(1:hidden_layer_size * (input_layer_size + 1)), ...
                 hidden_layer_size, (input_layer_size + 1));

Theta2 = reshape(nn_params((1 + (hidden_layer_size * (input_layer_size + 1))):end), ...
                 num_labels, (hidden_layer_size + 1));



predNN1=predict(Theta1,Theta2,Xtest);
fprintf('\nTest Accuracy: %f\n', mean(double(predNN1 == ytest)) * 100);


C1=confusionmat(ytest,predNN1);
cm=confusionchart(C1);
cm.Title='Neural Network';
```

```
Iteration  19988 | Cost: 9.935718e-01
Iteration  19989 | Cost: 9.935717e-01
Iteration  19990 | Cost: 9.935717e-01
Iteration  19991 | Cost: 9.935715e-01
Iteration  19992 | Cost: 9.935714e-01
Iteration  19993 | Cost: 9.935712e-01
Iteration  19994 | Cost: 9.935712e-01
Iteration  19995 | Cost: 9.935703e-01
Iteration  19996 | Cost: 9.935600e-01
Iteration  19997 | Cost: 9.935593e-01
Iteration  19998 | Cost: 9.935575e-01
Iteration  19999 | Cost: 9.935518e-01
Iteration  20000 | Cost: 9.935475e-01


Test Accuracy: 85.000000
```



Neural networks also worked well on our dataset. Results on confusion matrix and evaluation score are satisfying enough for our project. Only 1 hidden layer was used and every layer size is given in the code. Optimization problem is solved with the same function as logistic regression which is fmincg. Again, neurons are selected as sigmoid functions for classification goals.

CONCLUSION

In summary, objective of the project was to get accuracy rate over 80% for determining the type of four different music genres. If the training data increases, accuracy will increase and if number of classes rise, accuracy may drop. I could have achieved even better results by increasing the iteration, but I left it because I knew that the process would increase and I already had good results.

REFERENCES

[1] https://www.coursera.org/learn/machine-learning
[2]Automatic Music Genre Classification of Audio Signals with Machine Learning Approaches"GSTFInternational Journal on Computing (JoC), Vol. 3 No.2, July 2013
[3]" Music Genre Classification using Machine Learning Techniques" HareeshBahuleyan
[4]https://ch.mathworks.com/help/audio/ref/audiofeatureextractor.html
[5]http://mirlab.org/jang/books/audioSignalProcessing/appNote/musicGenreClassification/html/goTutorial.html
[6]https://www.udemy.com/course/machine-learning-for-datascience-using-matlab/