Music genre recognition from audio file

To be able to implement a solution for our problem, firstly we need to understand the features/characteristics of an audio file, more specifically, features of certain music genres. The features that can be used in our solution are separated into temporal and spectral. They are different for each genre.

Temporal features are features obtained in the time domain. The ones that could be useful for us are BPM (Beats Per Minute) and ZCR (Zero Crossing Rate).

- BPM measures rhythmic intensity and varies from genre to genre (Rock 110 – 140, Pop 100 – 130, Hip Hop 85 – 115, Reggae 60 – 90).
- ZCR measures the frequency of signal sign changes (for example, in speech ZCR is high because of frequent amplitude changes, whereas in jazz ZCR is low, because amplitudes change smoothly).

Spectral features are features obtained in the frequency domain.

- MFCC (Mel-Frequency Cepstral Coefficients) are numerical features that describe the "shape" of sound, optimized for how the human ear perceives sound. They are used for speech recognition, music genre classification, and audio analysis.
- Spectral Centroid measures the "center of mass" of the frequency spectrum (in Hz)
- Spectral Contrast is an audio feature that measures differences between the strongest (peaks) and weakest (valleys) parts of a sound across different frequency ranges. It helps identify textures and dynamics in audio signals. (similar to ZCR)

We will use the GTZAN dataset, but we will need to shorten the audio files to 5 seconds. We can either do that in code (using sampling) or manually in a program such as Audacity. When we adapt our dataset and calculate the features above, that can be used as data that we can use to develop a model using machine learning, that will perform classification of music into genres.

We can use different approaches to solving our problem – K-NN algorithm (K – nearest Neighbours), Naive Bayes Classifiers, RNN (Recurrent Neural Networks), and many more.

K-NN algorithm

K-NN algorithm is an algorithm used in supervised learning. It works through 4 steps:

1. Selecting the value of K - K represents the number of nearest neighbors that needs to be considered while making prediction.
2. Calculating distance - To measure the similarity between target and training data points Euclidean distance is used. Distance is calculated between data points in the dataset and target point.
3. Finding Nearest Neighbors - The K data points with the smallest distances to the target point are nearest neighbors.
4. Voting for Classification - When you want to classify a data point into a category the K-NN algorithm looks at the K closest points in the dataset. These closest points are called neighbors. The algorithm then looks at which category the neighbors belong to and picks the one that appears the most. This is called majority voting.

In our case, if we choose the K-NN, we will need to separate data for training and data for testing. Every datapoint represents a certain song/audio file. If we do a good calculation of features mentioned above and do the necessary pre-processing of the input data (for example scaling) we can apply the K-NN algorithm.

Naïve Bayes algorithm

Naive Bayes is a probabilistic classifier based on Bayes' Theorem, which describes how to update probabilities based on new evidence. It treats the problem mathematically, assuming independence between features. It follows four steps:

1. Calculate Prior Probabilities - Determine the overall probability of each genre occurring in the dataset.
2. Compute Feature Likelihoods - For each genre, calculate how often each extracted feature (BPM, ZCR, MFCCs, etc.) appears.
3. Applying Bayes Theorem - Use probability rules to update the likelihood of each genre given the extracted features.
4. Select the Most Probable Genre - The genre with the highest final probability is chosen as the prediction.

If we choose Naive Bayes, we will need to separate the data for training and testing. Each data point represents a song/audio file. After performing feature extraction and pre-processing we can apply the Naive Bayes algorithm. Naive Bayes is simple and fast, and works well with high-dimensional data, but it assumes feature independence, which can lead to poor performance if features are highly correlated.

Recurrent Neural Networks (RNN)

RNN’s is a deep learning model made to process sequential data. RNN’s are designed to handle data where the order of components is crucial (for example the order of words in a sentence or translating text from one language to another). Their ability to process sequential data is based on a hidden state. It acts as a memory, retaining information about previous inputs in the sequence and using it to influence the processing of the current input. For music recognition, the best choice would be a many to one RNN model. This architecture will process a sequence of inputs and generate a single output. The main advantage of using RNN is that music is time dependent data. For example, a specific drum beat at the beginning of the song can indicate a particular genre.