

Feature extraction



In [3]:

```
1 header = 'filename chroma_stft_mean chroma_stft_var rms_mean rms_var spectral_centroid_
2 for i in range(1, 21):
3     header += f'mfcc_mean{i} '
4     header+= f'mfcc_var{i} '
5 for j in range(1,129):
6     header += f'mel_specgram_mean{j} '
7     header += f'mel_specgram_var{j} '
8 header += ' tempo'
9 header += ' label'
10 header = header.split()
```

In [4]:

```
1 len(header)
```

Out[4]:

343

In [7]:

```

1 file = open('music.csv', 'w', newline='')
2 with file:
3     writer = csv.writer(file)
4     writer.writerow(header)
5 genres = 'final_blues final_classical final_country final_disco final_hiphop final_jazz
6 for g in genres:
7     for filename in os.listdir(f'./final_genres/{g}'):
8         songname = f'./final_genres/{g}/{filename}'
9         y, sr = librosa.load(songname, mono=True, duration=30)
10        chroma_stft = librosa.feature.chroma_stft(y=y, sr=sr)
11        rms = librosa.feature.rms(y=y)
12        spec_cent = librosa.feature.spectral_centroid(y=y, sr=sr)
13        spec_bw = librosa.feature.spectral_bandwidth(y=y, sr=sr)
14        rolloff = librosa.feature.spectral_rolloff(y=y, sr=sr)
15        zcr = librosa.feature.zero_crossing_rate(y)
16        mfcc = librosa.feature.mfcc(y=y, sr=sr)
17        harmonics, perceptual = librosa.effects.hpss(y)
18        tempo=librosa.beat.tempo(y=y,sr=sr)
19        spectral_contrast=librosa.feature.spectral_contrast(y,sr=sr)
20        #Spectral flatness
21        spectral_flatness=librosa.feature.spectral_flatness(y=y)
22        mel_spectrogram=librosa.feature.melspectrogram(y=y,sr=sr)
23        to_append = f'{filename} {np.mean(chroma_stft)} {np.var(chroma_stft)} {np.mean(
24        for e in mfcc:
25            to_append += f' {np.mean(e)}'
26            to_append += f' {np.var(e)}'
27        for j in range(0,7):
28            to_append += f' {np.mean(spectral_contrast[j])}'
29            to_append += f' {np.var(spectral_contrast[j])}'
30            j+=1
31
32        to_append += f' {np.mean(spectral_flatness)}'
33        to_append += f' {np.var(spectral_flatness)}'
34        y_harmonic=librosa.effects.harmonic(y=y)
35        tonnetz=librosa.feature.tonnetz(y=y_harmonic,sr=sr)
36        for k in range(0,6):
37            to_append += f' {np.mean(tonnetz[j])}'
38            to_append += f' {np.var(tonnetz[j])}'
39            k+=1
40        for l in range(0,128):
41            to_append += f' {np.mean(mel_spectrogram[j])}'
42            to_append += f' {np.var(mel_spectrogram[j])}'
43        to_append += f' {float(tempo)}'
44        to_append += f' {g}'
45        file = open('music.csv', 'a', newline='')
46        with file:
47            writer = csv.writer(file)
48            writer.writerow(to_append.split())

```

C:\Users\PROMIT\anaconda3\envs\tensorflow\lib\site-packages\librosa\core\spectrum.py:222: UserWarning: n_fft=1024 is too small for input signal of length=1011

warnings.warn(

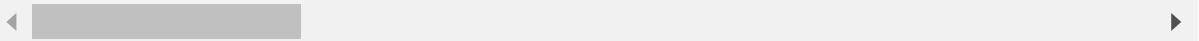
In [8]:

```
1 import pandas as pd
2 data=pd.read_csv('music.csv')
3 data.head()
```

Out[8]:

	filename	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var	spectral_centroid_
0	final_blues00.wav	0.335434	0.091088	0.130405	0.003521	1773.2
1	final_blues01.wav	0.343020	0.086142	0.112699	0.001450	1816.1
2	final_blues02.wav	0.346838	0.092210	0.132002	0.004620	1788.6
3	final_blues03.wav	0.363671	0.086856	0.132562	0.002447	1654.9
4	final_blues04.wav	0.335927	0.088291	0.143289	0.001701	1630.7

5 rows × 343 columns



In [9]:

```
1 data.shape
```

Out[9]:

(10000, 343)