FMA: A Dataset For Music Analysis

Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, Xavier Bresson, EPFL LTS2.

Usage

1. Go through the paper (https://arxiv.org/abs/1612.01840) to understand what the data is about.
2. Download some datasets from https://github.com/mdeff/fma (https://github.com/mdeff/fma).
3. Uncompress the archives, e.g. with `unzip fma_small.zip` .
4. Load and play with the data in this notebook.

In [1]:
```
%matplotlib inline

import os

import IPython.display as ipd
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn as skl
import sklearn.utils, sklearn.preprocessing, sklearn.decomposition, sklearn.svm
import librosa
import librosa.display

import utils

plt.rcParams['figure.figsize'] = (17, 5)
```

In [2]:
```
# Directory where mp3 are stored.
AUDIO_DIR = os.environ.get('AUDIO_DIR')

# Load metadata and features.
tracks = utils.load('tracks.csv')
genres = utils.load('genres.csv')
features = utils.load('features.csv')
echonest = utils.load('echonest.csv')

np.testing.assert_array_equal(features.index, tracks.index)
assert echonest.index.isin(tracks.index).all()

tracks.shape, genres.shape, features.shape, echonest.shape
```

Out[2]:
```
((106574, 52), (163, 4), (106574, 518), (13129, 249))
```

1 Metadata

The metadata table, a CSV file in the `fma_metadata.zip` archive, is composed of many colums:

1. The index is the ID of the song, taken from the website, used as the name of the audio file.
2. Per-track, per-album and per-artist metadata from the Free Music Archive website.
3. Two columns to indicate the subset (small, medium, large) and the split (training, validation, test).

In [3]:

```
ipd.display(tracks['track'].head())
ipd.display(tracks['album'].head())
ipd.display(tracks['artist'].head())
ipd.display(tracks['set'].head())
```

track_id	bit_rate	comments	composer	date_created	date_recorded	duration	favorites	genre_top	genres	genres_all
2	256000	0	NaN	2008-11-26 01:48:12	2008-11-26	168	2	Hip-Hop	[21]	[21]
3	256000	0	NaN	2008-11-26 01:48:14	2008-11-26	237	1	Hip-Hop	[21]	[21]
5	256000	0	NaN	2008-11-26 01:48:20	2008-11-26	206	6	Hip-Hop	[21]	[21]
10	192000	0	Kurt Vile	2008-11-25 17:49:06	2008-11-26	161	178	Pop	[10]	[10]
20	256000	0	NaN	2008-11-26 01:48:56	2008-01-01	311	0	NaN	[76, 103]	[17, 10, 76, 103]

track_id	comments	date_created	date_released	engineer	favorites	id	information	listens	producer	tags	title
2	0	2008-11-26 01:44:45	2009-01-05	NaN	4	1		6073	NaN	[]	AWOL - A Way Of Life
3	0	2008-11-26 01:44:45	2009-01-05	NaN	4	1		6073	NaN	[]	AWOL - A Way Of Life
5	0	2008-11-26 01:44:45	2009-01-05	NaN	4	1		6073	NaN	[]	AWOL - A Way Of Life
10	0	2008-11-26 01:45:08	2008-02-06	NaN	4	6	NaN	47632	NaN	[]	Constant Hitmaker
20	0	2008-11-26 01:45:05	2009-01-06	NaN	2	4	"spiritual songs" from Nicky Cook	2710	NaN	[]	Niris

track_id	active_year_begin	active_year_end	associated_labels	bio	comments	date_created	fav
2	2006-01-01	NaT	NaN	A Way Of Life, A Collective of Hip-Hop from...	0	2008-11-26 01:42:32	
3	2006-01-01	NaT	NaN	A Way Of Life, A Collective of Hip-Hop from...	0	2008-11-26 01:42:32	
5	2006-01-01	NaT	NaN	A Way Of Life, A Collective of Hip-Hop from...	0	2008-11-26 01:42:32	
10	NaT	NaT	Mexican Summer, Richie Records, Woodsist, Skul...	<span style="font-family:Verdana, Geneva, A...	3	2008-11-26 01:42:55	

track_id	active_year_begin	active_year_end	associated_labels	bio	comments	date_created	fav
20	1990-01-01	2011-01-01	NaN	Songs written by: Nicky Cook \n VOCALS...	2	2008-11-26 01:42:52	

track_id	split	subset
2	training	small
3	training	medium
5	training	small
10	training	small
20	training	large

2 Genres

The genre hierarchy is stored in `genres.csv` and distributed in `fma_metadata.zip`.

In [4]:
```
print('{} top-level genres'.format(len(genres['top_level'].unique())))
genres.loc[genres['top_level'].unique()].sort_values('#tracks', ascending=False)
```

```
16 top-level genres
```

Out[4]:

genre_id	#tracks	parent	title	top_level
38	38154	0	Experimental	38
15	34413	0	Electronic	15
12	32923	0	Rock	12
1235	14938	0	Instrumental	1235
10	13845	0	Pop	10
17	12706	0	Folk	17
21	8389	0	Hip-Hop	21
2	5271	0	International	2
4	4126	0	Jazz	4
5	4106	0	Classical	5
9	1987	0	Country	9
20	1876	0	Spoken	20
3	1752	0	Blues	3
14	1499	0	Soul-RnB	14
8	868	0	Old-Time / Historic	8
13	730	0	Easy Listening	13

In [5]:
```
genres.sort_values('#tracks').head(10)
```

Out[5]:

genre_id	#tracks	parent	title	top_level
175	0	86	Bollywood	2
178	0	4	Be-Bop	4
377	1	19	Deep Funk	14
173	4	86	N. Indian Traditional	2
493	4	651	Western Swing	9
374	9	20	Banter	20
808	12	46	Salsa	2
174	17	86	South Indian Traditional	2
465	18	20	Musical Theater	20
176	23	2	Pacific	2

3 Features

1. Features extracted from the audio for all tracks.
2. For some tracks, data colected from the Echonest (http://the.echonest.com/) API.

In [6]:

```
print('{1} features for {0} tracks'.format(*features.shape))
columns = ['mfcc', 'chroma_cens', 'tonnetz', 'spectral_contrast']
columns.append(['spectral_centroid', 'spectral_bandwidth', 'spectral_rolloff'])
columns.append(['rmse', 'zcr'])
for column in columns:
    ipd.display(features[column].head().style.format('{:.2f}'))
```

```
518 features for 106574 tracks
```

statistics																	
number	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17
track_id																	
2	3.86	1.54	0.00	0.33	0.12	-0.34	-0.26	0.15	0.41	-0.16	-0.03	0.43	-0.23	-0.30	-0.19	-0.05	-0.15
3	4.30	1.40	0.11	-0.21	0.03	-0.02	0.15	0.05	0.03	-0.06	0.51	0.37	0.21	0.10	0.48	0.27	0.11
5	2.62	2.42	0.44	-0.78	-0.77	-0.72	0.09	0.15	0.26	-0.61	0.10	-0.25	0.16	0.64	0.19	0.29	-0.07
10	5.08	1.16	2.10	1.37	-0.20	-0.35	-0.53	0.56	0.28	-0.15	-0.05	-0.19	0.02	0.11	0.18	-0.12	-0.03
20	11.88	4.09	0.00	1.52	0.18	0.34	0.37	0.07	-0.02	0.03	0.25	0.13	0.08	0.04	-0.05	0.06	0.32

statistics													kurtosis					
number	01	02	03	04	05	06	07	08	09	10	11	12	01	02	03	04	05	06
track_id																		
2	7.18	5.23	0.25	1.35	1.48	0.53	1.48	2.69	0.87	1.34	1.35	1.24	0.69	0.57	0.60	0.63	0.57	0.44
3	1.89	0.76	0.35	2.30	1.65	0.07	1.37	1.05	0.11	0.62	1.04	1.29	0.68	0.58	0.58	0.58	0.45	0.46
5	0.53	-0.08	-0.28	0.69	1.94	0.88	-0.92	-0.93	0.67	1.04	0.27	1.13	0.61	0.65	0.49	0.45	0.47	0.45
10	3.70	-0.29	2.20	-0.23	1.37	1.00	1.77	1.60	0.52	1.98	4.33	1.30	0.46	0.54	0.45	0.65	0.59	0.51
20	-0.19	-0.20	0.20	0.26	0.78	0.08	-0.29	-0.82	0.04	-0.80	-0.99	-0.43	0.65	0.68	0.67	0.60	0.65	0.70

statistics							kurtosis						max					mean
number	01	02	03	04	05	06	01	02	03	04	05	06	01	02	03	04	05	06
track_id																		
2	2.30	0.98	1.03	1.67	0.83	8.46	0.10	0.16	0.21	0.32	0.06	0.07	-0.00	0.02	0.01	0.07	0.01	0.02
3	2.00	1.38	0.87	2.02	0.43	0.48	0.18	0.11	0.27	0.21	0.07	0.07	0.00	0.01	0.05	-0.03	0.02	-0.00
5	10.77	0.92	-0.19	0.53	0.15	7.70	0.25	0.09	0.19	0.18	0.07	0.08	-0.01	-0.02	-0.03	0.02	0.00	-0.00
10	0.50	2.95	0.09	3.00	4.28	0.35	0.06	0.10	0.32	0.19	0.12	0.06	-0.02	-0.02	-0.00	-0.07	0.01	0.01
20	1.11	4.17	0.25	0.16	0.52	0.46	0.17	0.19	0.35	0.29	0.10	0.07	0.01	0.01	-0.02	-0.08	0.01	-0.01

statistics								kurtosis						max				
number	01	02	03	04	05	06	07	01	02	03	04	05	06	07	01	02	03
track_id																	
2	2.27	0.45	0.76	0.08	0.01	7.25	3.26	50.83	41.39	39.33	31.51	33.35	47.27	54.69	18.01	15.36	17.13
3	3.21	0.24	-0.01	-0.28	0.25	1.28	3.73	59.70	40.60	42.14	31.47	38.78	37.24	58.20	15.73	15.05	17.37
5	1.48	0.36	0.46	0.08	-0.11	0.77	0.93	50.93	38.54	39.02	33.71	32.75	38.79	51.90	17.10	15.97	18.65
10	2.24	0.74	1.61	0.46	-0.14	1.70	-0.40	58.62	37.43	45.41	28.71	33.74	46.30	52.13	19.18	14.28	15.5
20	2.26	0.32	0.63	0.10	0.37	0.70	11.45	46.74	42.15	39.38	33.53	36.35	28.78	48.10	16.04	16.16	17.70

feature																	spectral_b
statistics	kurtosis		max	mean	median	min	skew	std	kurtosis		max	mean	median	min	skew		

feature	spectral_centroid												
statistics	kurtosis	max	mean	median	min	skew	std	kurtosis	max	mean	median	min	skew
number	01	01	01	01	01	01	01	01	01	01	01	01	01
track_id													
2	2.41	5514.05	1639.58	1503.50	0.00	1.08	719.77	3.87	3451.11	1607.47	1618.85	0.00	-0.8
3	3.52	6288.43	1763.01	1517.99	0.00	1.65	972.76	2.38	3469.18	1736.96	1686.77	0.00	0.4
5	1.32	5648.61	1292.96	1186.51	0.00	0.94	665.32	0.90	3492.74	1512.92	1591.52	0.00	-0.6
10	9.73	5739.39	1360.03	1180.97	0.00	2.52	668.70	0.44	3962.70	1420.26	1301.81	0.00	0.8
20	2.18	5540.21	1732.97	1640.78	123.61	0.96	481.93	1.69	3556.88	2489.02	2467.10	677.70	-0.1

feature	rmse							zcr						
statistics	kurtosis	max	mean	median	min	skew	std	kurtosis	max	mean	median	min	skew	std
number	01	01	01	01	01	01	01	01	01	01	01	01	01	01
track_id														
2	2.50	14.75	3.19	2.65	0.00	1.57	2.54	5.76	0.46	0.09	0.07	0.00	2.09	0.06
3	-0.64	9.10	3.61	3.71	0.00	0.02	1.95	2.82	0.47	0.08	0.06	0.00	1.72	0.07
5	0.00	11.03	3.25	2.41	0.00	1.03	2.59	6.81	0.38	0.05	0.04	0.00	2.19	0.04
10	1.77	12.32	3.89	3.76	0.00	0.83	2.00	21.43	0.45	0.08	0.07	0.00	3.54	0.04
20	1.24	16.18	4.60	4.37	0.00	0.80	2.18	16.67	0.47	0.05	0.04	0.00	3.19	0.03

3.1 Echonest features

In [7]:

```
print('{1} features for {0} tracks'.format(*echonest.shape))
ipd.display(echonest['echonest', 'metadata'].head())
ipd.display(echonest['echonest', 'audio_features'].head())
ipd.display(echonest['echonest', 'social_features'].head())
ipd.display(echonest['echonest', 'ranks'].head())
```

```
249 features for 13129 tracks
```

track_id	album_date	album_name	artist_latitude	artist_location	artist_longitude	artist_name	release
2	NaN	NaN	32.6783	Georgia, US	-83.2230	AWOL	AWOL - A Way Of Life
3	NaN	NaN	32.6783	Georgia, US	-83.2230	AWOL	AWOL - A Way Of Life
5	NaN	NaN	32.6783	Georgia, US	-83.2230	AWOL	AWOL - A Way Of Life
10	2008-03-11	Constant Hitmaker	39.9523	Philadelphia, PA, US	-75.1624	Kurt Vile	Constant Hitmaker
134	NaN	NaN	32.6783	Georgia, US	-83.2230	AWOL	AWOL - A Way Of Life

track_id	acousticness	danceability	energy	instrumentalness	liveness	speechiness	tempo	valence
2	0.416675	0.675894	0.634476	0.010628	0.177647	0.159310	165.922	0.576661
3	0.374408	0.528643	0.817461	0.001851	0.105880	0.461818	126.957	0.269240
5	0.043567	0.745566	0.701470	0.000697	0.373143	0.124595	100.260	0.621661
10	0.951670	0.658179	0.924525	0.965427	0.115474	0.032985	111.562	0.963590
134	0.452217	0.513238	0.560410	0.019443	0.096567	0.525519	114.290	0.894072

track_id	artist_discovery	artist_familiarity	artist_hotttnesss	song_currency	song_hotttnesss
2	0.388990	0.386740	0.406370	0.000000	0.000000
3	0.388990	0.386740	0.406370	0.000000	0.000000
5	0.388990	0.386740	0.406370	0.000000	0.000000
10	0.557339	0.614272	0.798387	0.005158	0.354516
134	0.388990	0.386740	0.406370	0.000000	0.000000

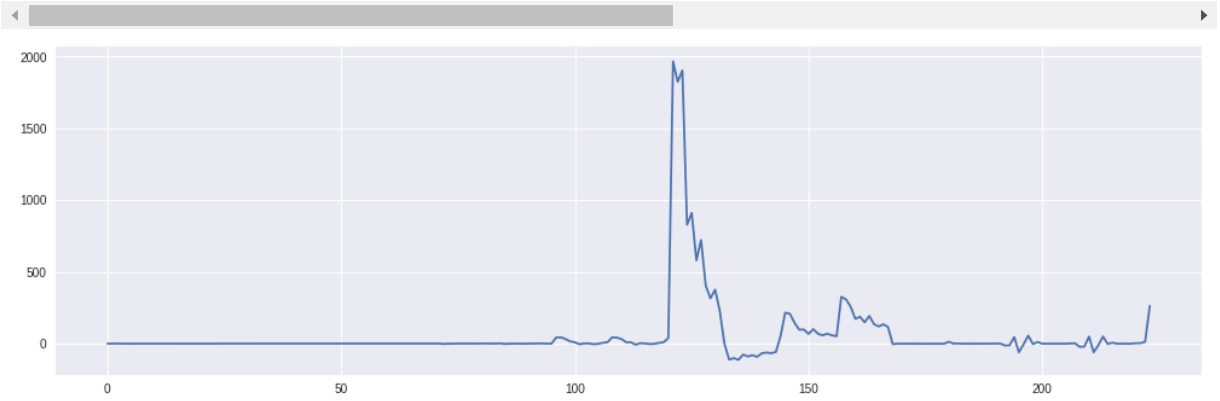
track_id	artist_discovery_rank	artist_familiarity_rank	artist_hotttnesss_rank	song_currency_rank	song_hotttnesss_rank
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN
10	2635.0	2544.0	397.0	115691.0	67609.0
134	NaN	NaN	NaN	NaN	NaN

In [8]:

```
ipd.display(echonest['echonest', 'temporal_features'].head())
x = echonest.loc[2, ('echonest', 'temporal_features')]
plt.plot(x);
```

track_id	000	001	002	003	004	005	006	007	008	009	...	
2	0.877233	0.588911	0.354243	0.295090	0.298413	0.309430	0.304496	0.334579	0.249495	0.259656	...	-1.992
3	0.534429	0.537414	0.443299	0.390879	0.344573	0.366448	0.419455	0.747766	0.460901	0.392379	...	-1.582
5	0.548093	0.720192	0.389257	0.344934	0.361300	0.402543	0.434044	0.388137	0.512487	0.525755	...	-2.288
10	0.311404	0.711402	0.321914	0.500601	0.250963	0.321316	0.734250	0.325188	0.373012	0.235840	...	-3.662
134	0.610849	0.569169	0.428494	0.345796	0.376920	0.460590	0.401371	0.449900	0.428946	0.446736	...	-1.452

5 rows × 224 columns



3.2 Features like MFCCs are discriminant

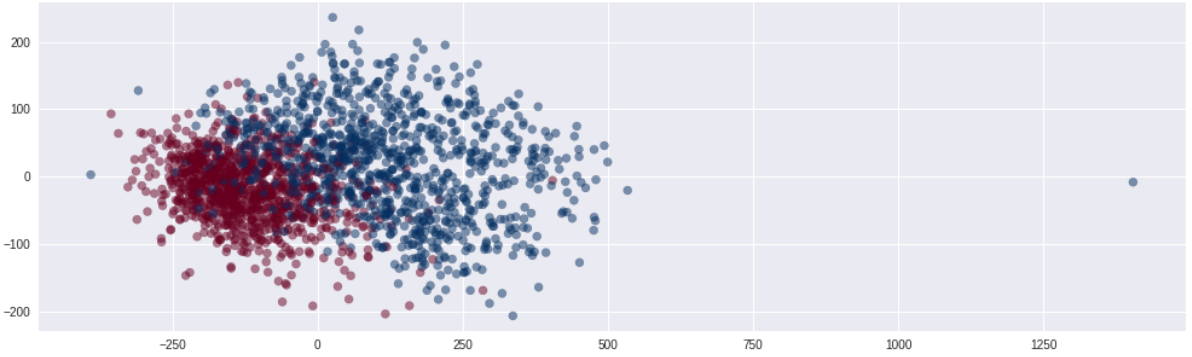
In [9]:
```
small = tracks['set', 'subset'] <= 'small'
genre1 = tracks['track', 'genre_top'] == 'Instrumental'
genre2 = tracks['track', 'genre_top'] == 'Hip-Hop'

X = features.loc[small & (genre1 | genre2), 'mfcc']
X = skl.decomposition.PCA(n_components=2).fit_transform(X)

y = tracks.loc[small & (genre1 | genre2), ('track', 'genre_top')]
y = skl.preprocessing.LabelEncoder().fit_transform(y)

plt.scatter(X[:,0], X[:,1], c=y, cmap='RdBu', alpha=0.5)
X.shape, y.shape
```

Out[9]:
```
((2000, 2), (2000,))
```



4 Audio

You can load the waveform and listen to audio in the notebook itself.

In [10]:
```
filename = utils.get_audio_path(AUDIO_DIR, 2)
print('File: {}'.format(filename))

x, sr = librosa.load(filename, sr=None, mono=True)
print('Duration: {:.2f}s, {} samples'.format(x.shape[-1] / sr, x.size))

start, end = 7, 17
ipd.Audio(data=x[start*sr:end*sr], rate=sr)
```

```
File: /media/share/fma/fma_full/000/000002.mp3
Duration: 167.86s, 7402752 samples
```

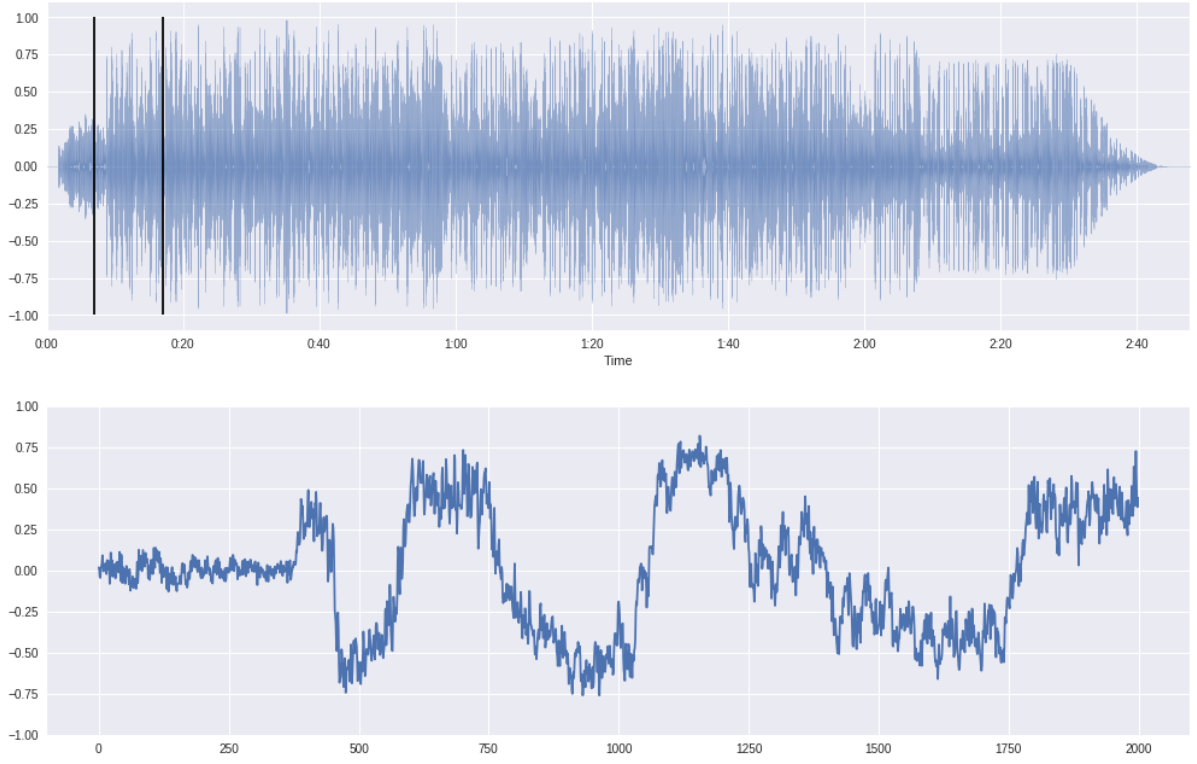
Out[10]:

0:00 / 0:10

And use librosa (https://github.com/librosa/librosa) to compute spectrograms and audio features.

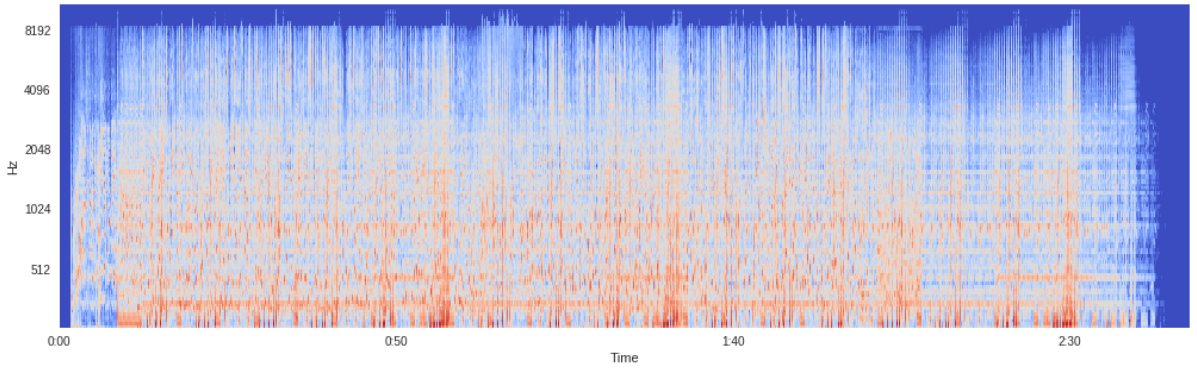
In [11]:
```
librosa.display.waveplot(x, sr, alpha=0.5);
plt.vlines([start, end], -1, 1)

start = len(x) // 2
plt.figure()
plt.plot(x[start:start+2000])
plt.ylim((-1, 1));
```



In [12]:
```
stft = np.abs(librosa.stft(x, n_fft=2048, hop_length=512))
mel = librosa.feature.melspectrogram(sr=sr, S=stft**2)
log_mel = librosa.logamplitude(mel)

librosa.display.specshow(log_mel, sr=sr, hop_length=512, x_axis='time', y_axis='mel');
```



In [13]:
```
mfcc = librosa.feature.mfcc(S=librosa.power_to_db(mel), n_mfcc=20)
mfcc = skl.preprocessing.StandardScaler().fit_transform(mfcc)
librosa.display.specshow(mfcc, sr=sr, x_axis='time');
```