Some Siri-ous Music Shuffler

Data science and Artificial Intelligence Society

Project Proposal

Topic: Machine Learning (Supervised)

Description: Web-scrape musical chord data / Data Preprocessing / Supervised Classification / Match

Expected Duration: 10 weeks

Team Member: Ikgyu Shin, Byunghoon Kwon, Sohyun Park

Why & What
this to expect

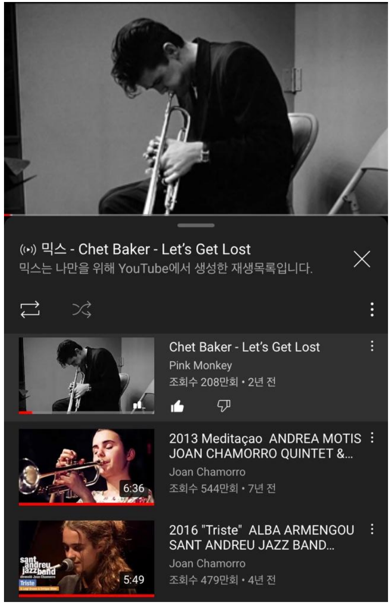
Digger

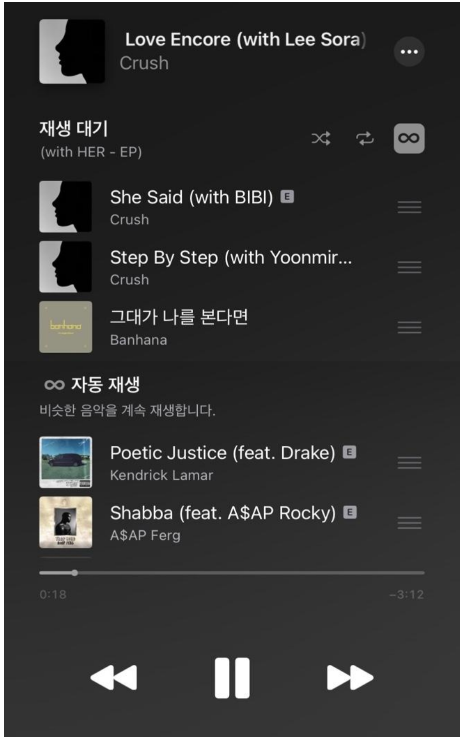
Hey Siri

What is my taste?

Chord itself?

Chord Progression





Algo-rhythm

Note	Hz	Note	Hz	Note	Hz	Note	Hz
C1	32.7	C2	65.4	C3	130.8	C4	261.6
C#1	34.6	C#2	69.3	C#3	138.6	C#4	277.2
D1	36.7	D2	73.4	D3	146.8	D4	293.7
D#1	38.9	D#2	77.8	D#3	155.6	D#4	311.1
E1	41.2	E2	82.4	E3	164.8	E4	329.6
F1	43.7	F2	87.3	F3	174.6	F4	349.2
F#1	46.2	F#2	92.5	F#3	185.0	F#4	370.0
G1	49.0	G2	98.0	G3	196.0	G4	392.0
G#1	51.9	G#2	103.8	G#3	207.7	G#4	415.3
A1	55.0	A2	110.0	A3	220.0	A4	440.0
A#1	58.3	A#2	116.5	A#3	233.1	A#4	466.2
B1	61.7	B2	123.5	B3	246.9	B4	493.9

Note Frequency Chart

	Octave 0	Octave 1	Octave 2	Octave 3	Octave 4	Octave 5	Octave 6	Octave 7	Octave 8
C	16.35	32.70	65.41	130.81	261.63	523.25	1046.50	2093.00	4186.01
C#	17.32	34.65	69.30	138.59	277.18	554.37	1108.73	2217.46	4434.92
D	18.35	36.71	73.42	146.83	293.66	587.33	1174.66	2349.32	4698.64
D#	19.45	38.89	77.78	155.56	311.13	622.25	1244.51	2489.02	4978.03
E	20.60	41.20	82.41	164.81	329.63	659.26	1318.51	2637.02	5274.04
F	21.83	43.65	87.31	174.61	349.23	698.46	1396.91	2793.83	5587.65
F#	23.12	46.25	92.50	185.00	369.99	739.99	1479.98	2959.96	5919.91
G	24.50	49.00	98.00	196.00	392.00	783.99	1567.98	3135.96	6271.93
G#	25.96	51.91	103.83	207.65	415.30	830.61	1661.22	3322.44	6644.88
A	27.50	55.00	110.00	220.00	440.00	880.00	1760.00	3520.00	7040.00
A#	29.14	58.27	116.54	233.08	466.16	932.33	1864.66	3729.31	7458.62
B	30.87	61.74	123.47	246.94	493.88	987.77	1975.53	3951.07	7902.13

TABLE 9.2 Frequencies of notes in tempered scale

C_0	16 352		C_3	130 81		C_6	1046 5	
		17 324			138 59			1108 7
D_0	18 354		D_3	146.83		D_6	1174.7	
		19.445			155 56			1244.5
E_0	20.602		E_3	164.81		E_6	1318 5	
F_0	21.827		F_3	174.61		F_6	1396 9	
		23 125			185.00			1480 0
G_0	24 500		G_3	196.00		G_6	1568.0	
		25.957			207 65			1661.2
A_0	27.500		A_3	220.00		A_6	1760 0	
		29 135			233.08			1864 7
	68		B_3	246.94		B_6	1975.5	
	03		C_4	261.63		C_7	2093 0	
		34 648			277.18			2217 5
	08		D_4	293 66		D_7	2349.3	
		38 891			311 13			2489.0
	03		E_4	329.63		E_7	2637 0	
	54		F_4	349.23		F_7	2793 8	
		46.249			369.99			2960 0
	99		G_4	392 00		G_7	3136.0	
		51 913			415 30			3322.4
	00		A_4	440 00		A_7	3520.0	
		58 270			466 16			3729.3
	35		B_4	493.88		B_7	3951 1	
	06		C_5	523.25		C_8	4186.0	
		69 296			554 37			4434.9
	16		D_5	587 33		D_8	4698 6	
		77.782			622.25			4978 0
	07		E_5	659.26		E_8	5274.0	
	07		F_5	698 46		F_8	5587.7	
		92 499			739 99			5919 9
	99		G_5	783 99		G_8	6271 9	
		103.83			830.61			6644 9
	00		A_5	880.00		A_8	7040.0	
		116 54			932 33			7458.6
	47		B_5	987.77		B_8	7902 1	

Data

```
In [42]: merged
```

Out[42]:

	url	name	decade	genre	chords	uuid
0	https://tabs.ultimate-guitar.com/tab/10000_man...	Dont Talk	1980s	Folk	['D', 'Dmaj7', 'D', 'Dmaj7', 'D', 'Dmaj7', 'D'...	c639eb23-fefd-4263-af20-3f78f110edcd
1	https://tabs.ultimate-guitar.com/tab/10000_man...	Whats The Matter Here	1980s	Folk%%Folk	['G', 'G', 'C', 'D', 'G', 'G', 'G', 'G', 'G', ...	828ba8e4-d791-4403-a4cc-5ca4a1c0cc1a
2	https://tabs.ultimate-guitar.com/tab/1002296	Limón Y Sal (ver 2)	2000s	Pop%%Folk	['F', 'G', 'Am', 'G', 'C', 'F', 'F', 'G', 'Am'...	bd5b9396-8d6a-49ee-a738-6577cf310783
3	https://tabs.ultimate-guitar.com/tab/1054759	Snälla Bli Min	2010s	Electronic	['Asus4', 'G', 'Bm7/A', 'Bm', 'D/F#', 'A', 'As...	ef979fca-e68a-4d8f-bd27-3d048516dc76
4	https://tabs.ultimate-guitar.com/tab/1055161	Time To Say Goodbye Con Te Partirò	1990s	Pop%%Classical%%Pop	['G', 'D', 'Em', 'C', 'G', 'D', 'Em', 'C', 'G'...	d42333bf-e925-4ad1-a4b9-bfe2e2df209e
...	...	...	...	...	...	...
14109	https://tabs.ultimate-guitar.com/tab/ziggy_mar...	Personal Revolution	2010s	Reggae	['Bm', 'A', 'Bm', 'A', 'Bm', 'A', 'Em', 'G', '...	089f0438-4e4d-44dd-83e5-5dd929753f0f
14110	https://tabs.ultimate-guitar.com/tab/ziggy_mar...	Shalom Salaam	2000s	Reggae	['Gm', 'Bb', 'F', 'Gm', 'Gm', 'Bb', 'F', 'Gm',...	149bb9e5-fb06-4d73-a69b-b5d9573d2993
14111	https://tabs.ultimate-guitar.com/tab/ziggy_mar...	True To Myself	2000s	Reggae%%Reggae%%Reggae	['A', 'E', 'Bm', 'D', 'A', 'E', 'Bm', 'D', 'A'...	c0d75c52-ae53-4b47-bc8e-a723ba542d00
14112	https://tabs.ultimate-guitar.com/tab/ziggy_mar...	True To Myself (ver 2)	2000s	Reggae	['A', 'E', 'Bm', 'A', 'E', 'Bm', 'A', 'E', 'Bm...	29047e58-5391-4875-ab94-3a5a7fdbcd0e
14113	https://tabs.ultimate-guitar.com/tab/ziggy_mar...	Wild And Free	2010s	Reggae	['G', 'A', 'E', 'B', 'E', 'B', 'E', 'B', 'E', ...	06c24412-8cf2-43bb-b0d5-5f3980270a14

BUT...

Analysis based on Musical Chord (obtained through web scraping & appliances in invented logic)

→ Cannot fully grasp complexity of music

Then recognized two useful libraries **"librosa" & "spotipy"**

Why & What
this to expect

Digger

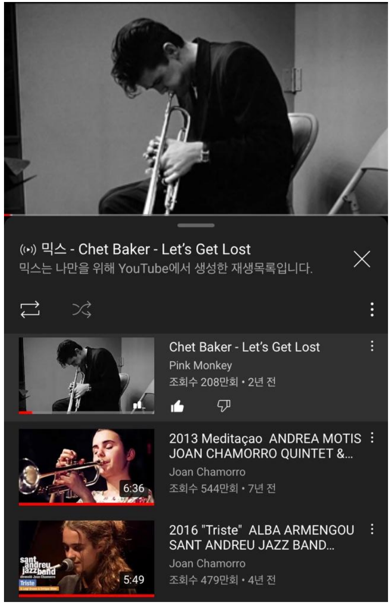
Hey Siri

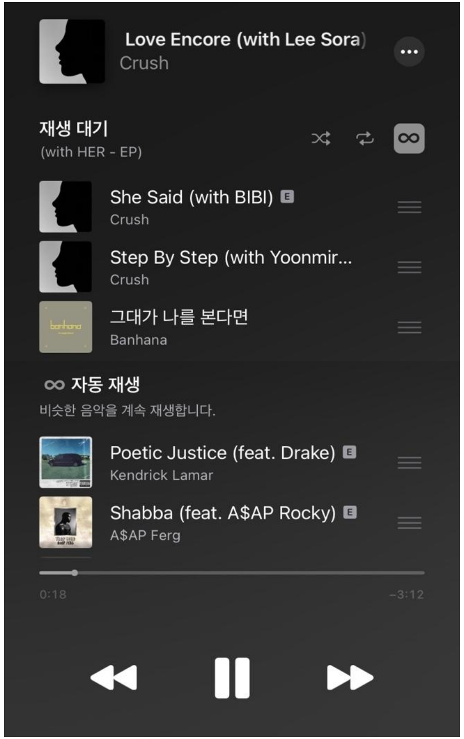
What is my taste?

~~Chord itself?~~

~~Chord Progression~~

Librosa





Music Sample Load

In [70]:
```
import librosa
y , sr = librosa.load('/Users/mac/Desktop/DAIS_VS/crush_sometimes.wav', duration=60.0)
```

In [72]:
```
pathAudio = "/Users/mac/Desktop/DAIS_VS/"
files = librosa.util.find_files(pathAudio, ext=['wav'])
#files = np.array(files)
# for y in files:
#     data = librosa.load(y, sr = 16000,mono = True)
#     data = data[0]
#     librosa.display.waveplot(data)
```

In [73]:
```
import os

path, filename = os.path.split(files[0])
root, ext = os.path.splitext(filename)
what_i_want, the_rest = root.rsplit("_", 1)


#what_i_want = os.path.splitext(os.path.split("/my/path/to/Planning_Group_20180108.ind")[1])
[0].rsplit("_", 1)
tada = []
tada.append(what_i_want)
tada.append(the_rest)
# what_i_want
# the_rest

tada
```

Out[73]:
```
['crush', 'sometimes']
```

Music sample default output

```
print(y)
print(len(y))
print('Sampling rate (Hz): %d' %sr)
print('Audio length (seconds): %.2f' % (len(y) / sr)) #음악의 길이(초) = 음파의 길이/Sampling rate
```

```
[0.         0.         0.         ... 0.18545943 0.15676436 0.17183484]
1323000
Sampling rate (Hz): 22050
Audio length (seconds): 60.00
```

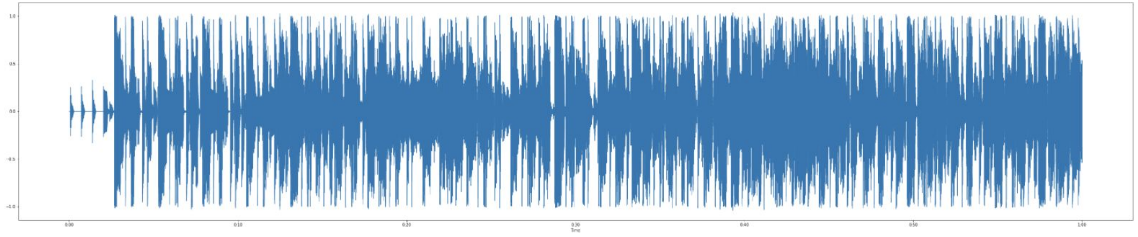
y (amplitude of the audio) = numeric

Audio length = amplitude / sr

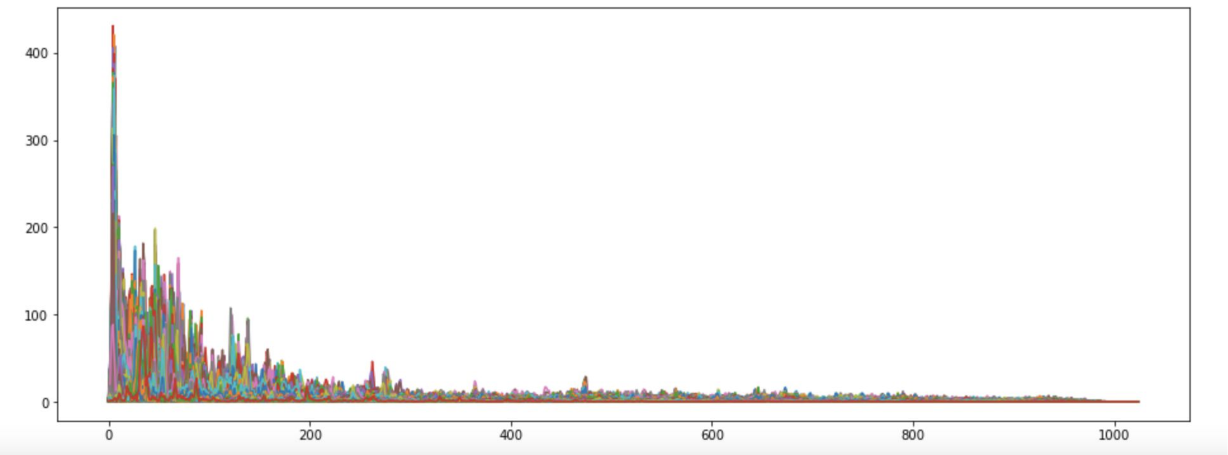
2D wave plot

```
import matplotlib.pyplot as plt
import librosa.display

plt.figure(figsize =(50,10))
librosa.display.waveshow(y=y,sr=sr)
```

```
<librosa.display.AdaptiveWaveplot at 0x7fd1f65b7ca0>
```



Fourier Transform

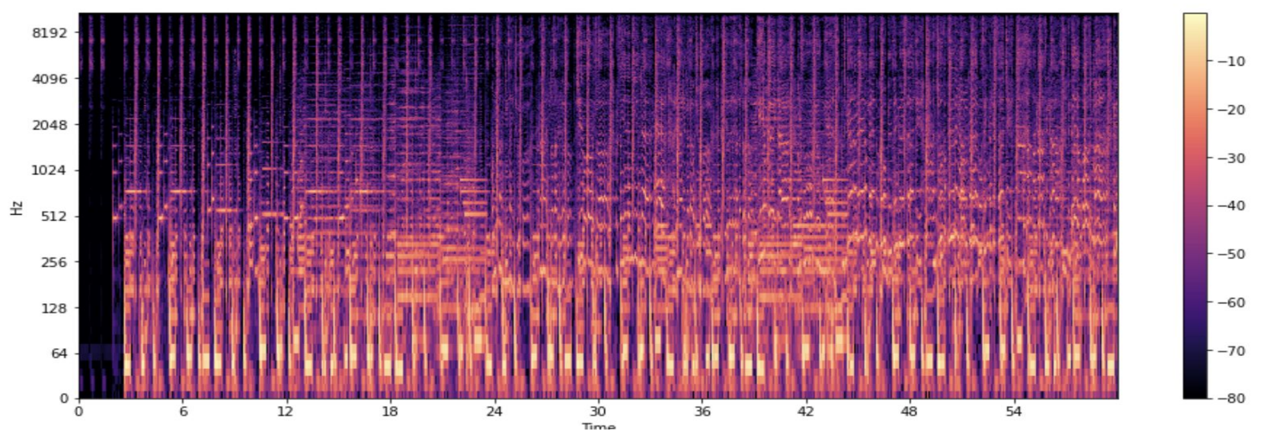


Conversion of **Time-domain** graph into **frequency-domain** graph

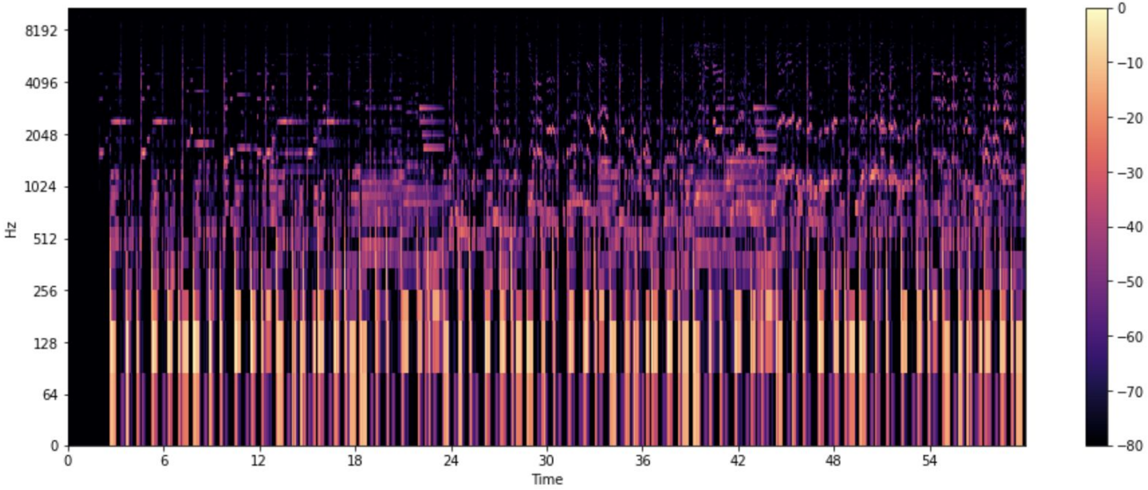
Many sin graphs of frequencies are yielded through Fourier Transformation

X-axis: **Frequency**, **Y**-axis: **Amplitude** → **Easier + Sophisticated analysis**

Spectogram (Sonographs|Voiceprints|Voicegrams)

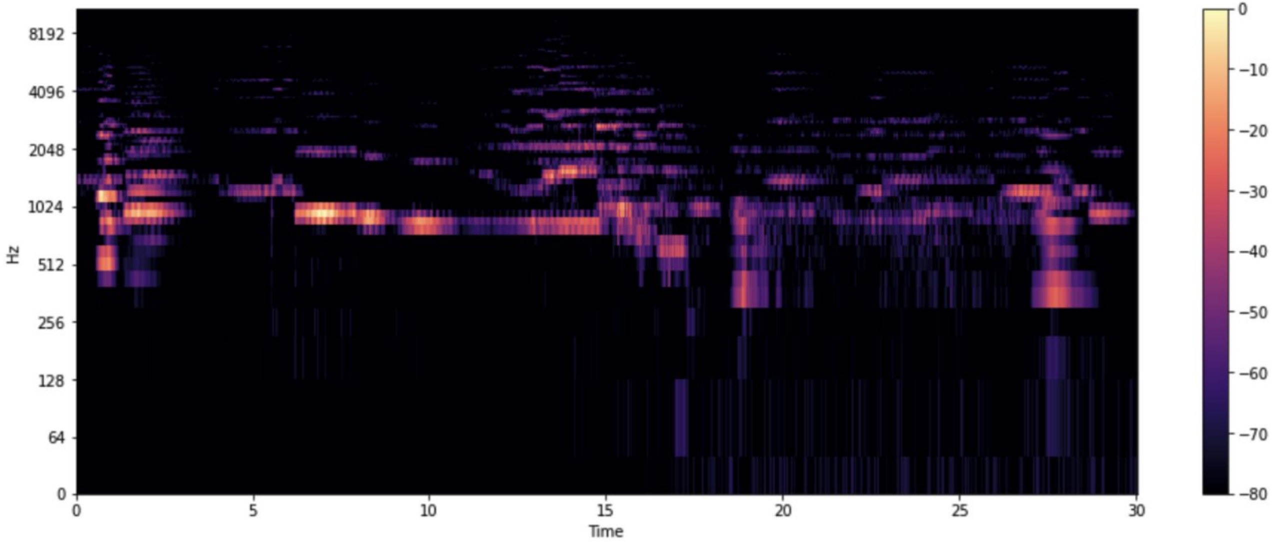


Mel Spectogram



Conversion of Spectogram's **y**-axis into **Mel Scale** (human-readable format) → **Non-linear Transformation**

(Sample) Classical music's Mel Spectogram



Tempo & Zero Crossing Rate Extraction

```
tempo , _ = librosa.beat.beat_track(y,sr=sr)
print(tempo)
```

```
<ipython-input-79-a4af5f969d2b>:1: FutureWarning: Pass y=[0.          0.          0.
... 0.18545943 0.15676436 0.17183484] as keyword args. From version 0.10 passing these as pos
itional arguments will result in an error
  tempo , _ = librosa.beat.beat_track(y,sr=sr)
```

```
92.28515625
```

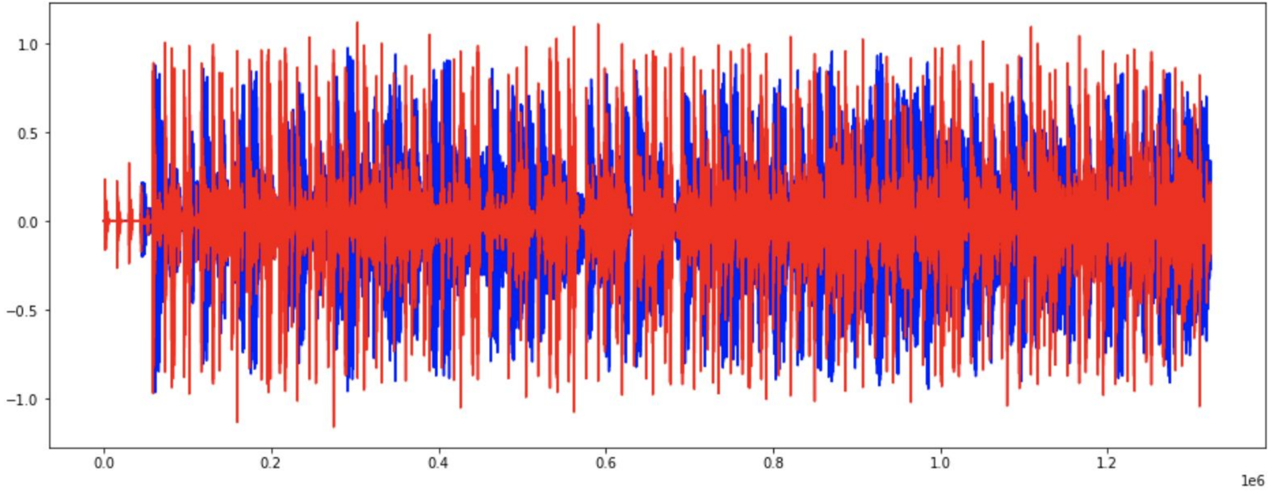
```
zero_crossings = librosa.zero_crossings(y, pad=False)

print(zero_crossings)
print(sum(zero_crossings)) # 음 <-> 양 이동한 횟수
```

```
[False False False ... False False False]
92760
```

The rate at which a sound wave converts from positive to negative or from negative to positive.

Often used for speech recognition or music analysis, and easy to analyze **Percussive feature**

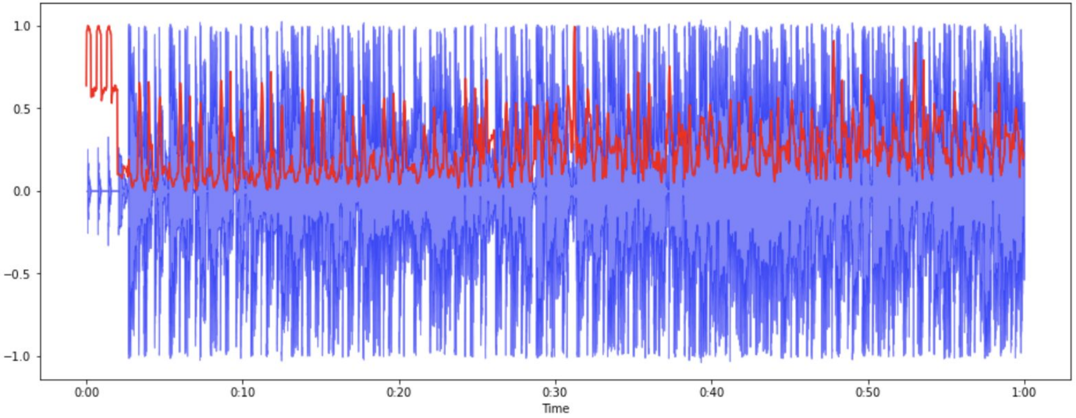
Harmonic & Percussives



Harmonic : Characteristics that cannot be distinguished by the human ear = the color of music

Blue: Harmonics | Red: Percussives

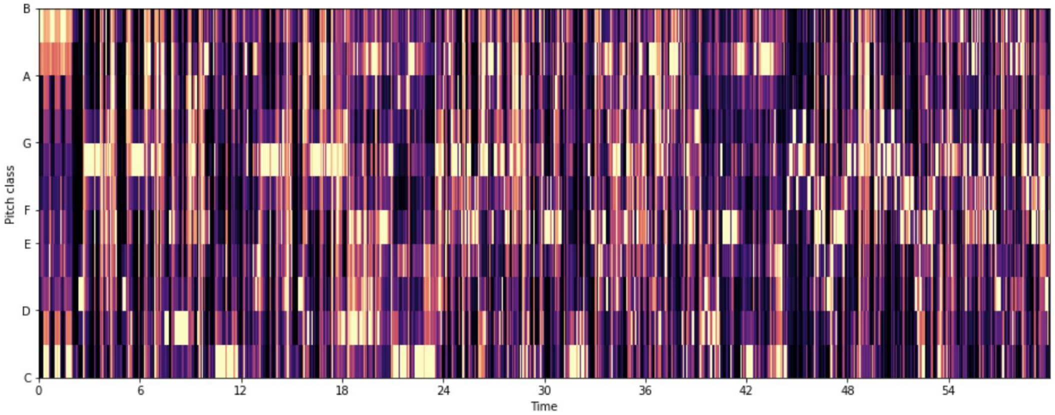
Spectral Centroid



Calculates the weighted average of frequencies from the sound expressed in frequency and tells where the **[center of gravity of the sound]** is.

Blues music: the center of gravity is centered
Metal music: the center of gravity is centered at the end → metal music runs at the end

Chroma Frequencies



Easy to recognize chords

Bases on the music theory that human hearing recognizes two sounds with frequencies with an octave difference as similar sounds

Feature Extraction & Save

```
from librosa import feature


fn_list_i = [
 feature.chroma_stft,
 feature.spectral_centroid,
 feature.spectral_bandwidth,
 feature.spectral_rolloff
]

fn_list_ii = [
 feature.rms,
 feature.zero_crossing_rate
]

def get_feature_vector(y,sr):
   feat_vect_i = [ np.mean(funct(y,sr)) for funct in fn_list_i]
   feat_vect_ii = [ np.mean(funct(y)) for funct in fn_list_ii]
   feature_vector = feat_vect_i + feat_vect_ii
   return feature_vector

feature_vector = get_feature_vector(y, sr)
tada.extend(feature_vector)
print(tada)
```

```python
import csv

header =[
 'song',
 'artist',
 'chroma_stft',
 'spectral_centroid',
 'spectral_bandwidth',
 'spectral_rolloff',
 'rmse',
 'zero_crossing_rate'
]

with open('sample.csv','w') as f:
 csv_writer = csv.writer(f, delimiter = ',')
 csv_writer.writerow(header)
 csv_writer.writerow(tada)
```

Spoti-py ?


```
{'acousticness': 0.446,
 'analysis_url': 'https://api.spotify.com/v1/audio-analysis/6y0igZArWVi6Iz0rj35c1Y',
 'danceability': 0.54,
 'duration_ms': 234910,
 'energy': 0.59,
 'id': '6y0igZArWVi6Iz0rj35c1Y',
 'instrumentalness': 0,
 'key': 0,
 'liveness': 0.14,
 'loudness': -4.359,
 'mode': 1,
 'speechiness': 0.0528,
 'tempo': 119.878,
 'time_signature': 4,
 'track_href': 'https://api.spotify.com/v1/tracks/6y0igZArWVi6Iz0rj35c1Y',
 'type': 'audio_features',
 'uri': 'spotify:track:6y0igZArWVi6Iz0rj35c1Y',
 'valence': 0.267}
```

Change of Plans

1. Decided to let go of **Spotipy**
    1. Features are not fully substantiated
2. **NO DROPS** considering **collinearity**
    1. K-means does not get BADLY affected

Genre Classification

song	artist	chroma_stft	spectral_centroid	spectral_bandwidth	spectral_rolloff	rmse	zero_crossing_rate
crush	sometimes	0.39172086	1853.81969	2189.910806	3960.317719	0.27187797	0.070059856

Masterplan

song	artist	chroma_stft	spectral_centroid	spectral_bandwidth	spectral_rolloff	rmse	zero_crossing_rate
crush	sometimes	0.39172086	1853.81969	2189.910806	3960.317719	0.27187797	0.070059856

1. Data Cleaning
    a. Drop unnecessary features (duplicate audio representation)
    b. sklearn - minmaxscaler
2. Model
    a. K-means clustering
    b. sklearn.metrics.pairwise - cosine_similarity

MinMaxScale

WHY: To deal with wide range of variables

- Preserves the shape of the original distribution
- Doesn't meaningfully change the information embedded in the original data
- Doesn't reduce the importance of outliers

HOW:

- Transform features by scaling each feature to a given range e.g. 0-1, 0-5

chroma_stft	spectral_centr	spectral_band	spectral_rollof	rms	zero_crossing	chroma_cens	spectral_contr	feature.mfcc	tempo
0.2647821	1307.3975	1841.24819	2865.7035	0.04887787	0.04442093	0.1872992	26.8413823	28.61816	103.359375
0.36767906	2850.43686	2654.35582	5998.95123	0.25452292	0.13377376	0.22510704	23.9193165	25.890448	129.199219
0.37814575	2131.03199	2361.64153	4791.12936	0.28312698	0.07083272	0.21760835	24.6722008	26.276594	99.3840144
0.44611198	2659.93974	2604.08014	5592.72868	0.31960315	0.11514311	0.22204307	22.143348	26.012596	89.1029095
0.2670146	2763.07473	2805.6133	6118.45467	0.12314584	0.11456129	0.17611174	27.2105459	27.691204	143.554688
0.29396346	2207.28426	2491.53302	4934.66183	0.22963063	0.09115119	0.20759115	27.1111021	26.148813	129.199219
0.2871341	661.930205	1111.47938	1032.38959	0.0858389	0.03012642	0.18684089	28.9984281	26.267736	107.666016

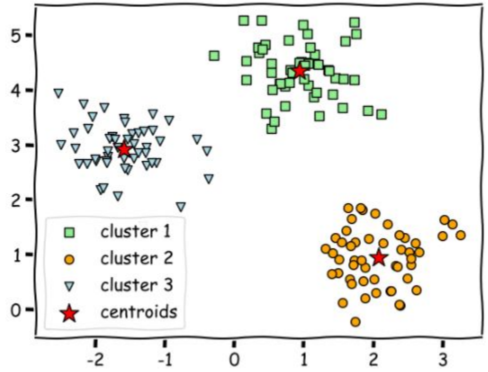
K-Means

WHY: Cluster musics with similar audio features

- Unsupervised learning that groups data with similar properties into a certain number of clusters
- It is characterized in that it is not possible to know how many clusters it will be divided into from the point of view of analysis.
- It is used when there is no correct answer for the independent variable and divides the data by attribute.

HOW:

- Find optimal number of group (k)
- Assign each data to the cluster corresponding to the nearest centroid



Cosine Similarity

WHY: Check how close and similar the values of the variables that make up the song

- Used as a metric to measure distance when the magnitude of the vector is not important.
- Meaning of similarity between two vectors that can be obtained using the cosine angle between the two vectors

HOW:

- If the directions of two vectors are exactly the same, it is 1, if it forms an angle of 90°, it is 0, if it has an opposite direction, it has a value of -1.
- That is, the range is greater than or equal to -1 and less than or equal to 1, and the closer the value is to 1, the higher the degree of similarity is judged.

Cosine Similarity : -1

Cosine Similarity : 0

Cosine Similarity : 1

```python
def find_similar_songs_(name, n=10):
  series = sim_ST[name].sort_values(ascending=False)

  series = series.drop(name)

  return series.head(n).to_frame()

#find_similar_songs(0)
names.loc[list(find_similar_songs_(0).index)]
```

	artist	song
781	MAMAMOO	AYA
41	AOA	Short Hair
78	Apink	NoNoNo
212	CNBLUE	Love
928	Primary, Gaeko, Zion.T	Seethru
436	GFRIEND	Me Gustas Tu
525	INFINITE	BTD (Before The Dawn)
373	EXID	DDD
274	Crush, ZICO	Oasis
11	2NE1	Clap Your Hands (박수쳐)

Music input: HANN by G-IDLE

Interpretation

k_cluster	chroma_stft	spectral_centroid	spectral_bandwidth	spectral_rolloff	rms
1	0.356622	2678.399002	2624.642919	5671.808134	0.179894

	artist	song	chroma_stft	spectral_centroid	spectral_bandwidth
8	2NE1, BIGBANG	Lollipop	0.440014	2686.520394	2554.849526
15	3LAU, Bright Lights	How You Love Me	0.326211	3227.416737	2902.293836
19	4Minute	Volume Up	0.345895	2744.915879	2461.621953
35	ADORA, EUNHA	MAKE U DANCE	0.358085	2494.736007	2387.220661
42	APRIL	LALALILALA	0.297457	2789.300312	2697.416208

Cluster #1 are grouped by k-pop (idols)
Nice!

```
print("What you want?: ")

x = int(input())
y = librosa_MM.iloc[x]['k_cluster']

print("Your pick is: " + names.iat[x,0] +"
print(librosa_MM[librosa_MM['k_cluster']==
rosa_MM[librosa_MM['k_cluster']==y]['artis
#print(group_MM.get_group(1)['playlist'].v
```

```
What you want?:
119
Your pick is: BLACKPINK - Kill This Love
0                  HANN (Alone) - (G)I-DLE
1                       LATATA - (G)I-DLE
9                        Kiss (Dara) - 2NE1
10      Please Don't Go (CL&Minzy) - 2NE1
11            Clap Your Hands (박수쳐) - 2NE1
13                       10 Out of 10 - 2PM
14                          Heartbeat - 2PM
dtype: object
```

Clusters are grouped by k-pop (idols)
Which is apparently similar to that of the input!

Interpretation

	song	chroma_stft	spectral_centroid	spectral_bandwidth
17	['Can_I_Love____feat__youra__Meego', '']	0.316272	1381.886737	1719.168721
21	['Easy']	0.419281	1225.363550	1630.064616
44	['Lucky_To_Be', 'Me']	0.353904	1323.944487	2024.150064
46	['Make_Me', 'Rainbows']	0.362804	967.686955	1690.989140
56	['On_a_Clear_Day__You_Can_See_Forever', '']	0.310390	1190.620190	1908.142794
63	['Regent_s', 'Park']	0.323171	973.677270	1386.411235
92	['When_I_fall_in', 'love']	0.358840	1538.971293	1850.365694

	song	chroma_stft	spectral_centroid
2	['AUTOMATIC']	0.378146	2131.031992
13	['Bon_voyage__feat_Beenzino', '']	0.427799	1769.541419
35	['Johnny']	0.374517	1986.597325
37	['Kiss_Me_More__feat__SZA', '']	0.328921	2007.317403
39	['LOVE']	0.340032	2041.087579
42	['Light', 'Switch']	0.411422	1763.400659
45	['MERRY-GO-ROUND__Feat__Zion_T__Wonstein___Pro...	0.401210	1867.283693
47	['Manila']	0.419320	1879.571839
50	['Nerdy', 'Love']	0.367930	2081.391252
53	['No_Make', 'Up']	0.332375	1921.996173
60	['Peaches__feat__Daniel_Caesar___Giveon', '']	0.353197	1870.755405
65	['Roses']	0.350775	2166.848208
79	['Sometimes']	0.390291	1863.375333
86	['Too', 'Good']	0.384841	1916.251260

Songs do actually have similar musical trait!

Limitation

1. Lack of understanding in audio features → attempted to many research papers & articles
2. Lack of data's credibility & accessibility
   a. **Spotipy**'s subjectivity
   b. **Spotipy**'s lack in user friendliness - limitation in crawling, etc
3. Lack of preciseness & accuracy test on model output
   a. The only way to know whether the model is successful or not is to let one individual to "**listen**" which, perhaps, yields extreme subjective result.
4. Lack of time in finding appropriate computation methods
   a. Graph image Deep Learning (CNN)
      i. Mean & SD can not fully substantiate the overall trend of a music

Thank You!

Data science and Artificial Intelligence Society