|  |  |  |  |
| --- | --- | --- | --- |
| 소프트웨어응용: 프로젝트 최종보고서 | | | |
| **기술개발**  **과 제** | 음원 파일의 차원 축소를 이용한 클러스터링 시각화 엔진  (영문: MIDI File Clustering Visualization Engine by Dimensionality Reduction) | | |
| **지도교수** | 황혜수 | | |
| **개발기간** | 2019 년 9 월 ~ 2019 년 12 월 (총 3개월) | | |
| **과제팀**  **구성원** | 이름 | 박인수 | 이효근 |
| 학번 | 2014920017 | 2014920045 |
| 연락처 | 010-9445-6838 | 010-3317-1032 |
| **소프트웨어응용 과제를 성실히 수행하고자 과제 제안서를 제출합니다.**  2019 년 12월 10 일  과제 수행자1 : 박 인 수 (인)  과제 수행자2 : 이 효 근 (인) | | | |

|  |
| --- |
| **1. 개발 과제의 개요** |

**가. 개발 과제 요약**

◇ MP3 Format의 음원(MIDI File)을 Dimensionality Reduction를 통해 음원 파일 간 거리를 측정하고, 각 음원 파일끼리 비교, 분석할 수 있게 한다.

◇ 각 음원들을 Clustering하여 분류하고, 항목별 Visualized Graph를 제공한다.

◇ 새로운 음원을 입력으로 넣어주면, 새 연산을 통해 이차원 그래프 위에 Clustering 한다.

**나. 개발 과제의 배경 및 효과**

◇ 평소 즐겨 듣는 음악과 유사한 곡이 있는지 찾을 수 있으며 추천할 수 있다.

◇ 음원의 표절 관련 시비를 음원 요소를 바탕으로 한 시스템적 분석이 가능하다.

◇ 음원 시장에서 새로 출시될 음원이 흥행할 수 있을지 예측해볼 수 있다.

**다. 개발 과제의 목표와 내용**

◇ 각 음원의 특징에 따른 분류를 정확히 할 수 있고 시각화를 구현하는 것을 목표로 한다.

◇ 시간적으로 여유가 있다면 데이터 유사도를 기반으로 한 추천 엔진까지 구현해본다.

◇ 내용 기반 필터링을 사용한 차원 축소를 사용하는데 있어, 측정할 데이터 차원의 개수를 늘려가는 것을 목표로 한다.

**라. 기술개발 일정 및 설계사양**

◇ 개발일정

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **단계별 세부개발 내용** | **개발기간 (월단위)** | | | | **비 고** |
| **9** | **10** | **11** | **12** |
| 프로젝트 제안 및 선정 |  |  |  |  | 완료 |
| 사용 기술 분석, 조사 및 개념 설계 |  |  |  |  | 완료 |
| 필요한 Dataset 수집 및 정리 |  |  |  |  | 완료 |
| 중간보고서 작성 및 제출 |  |  |  |  | 완료 |
| 오픈소스 조사 및 Google Colab 활용한 공유 |  |  |  |  | 완료 |
| Dataset 가공 및 활용한 프로그래밍 |  |  |  |  | 완료 |
| 최종 보고서 작성, 성과 발표 |  |  |  |  | 완료 |

◇ 설계사양

-MSI GE60-2PC (i7-4710HQ), Lenovo Ideapad 330(i3-7130U)

◇ 개발 환경 및 데이터의 종류

- 개발 환경 : Python, Google Colaboratory, Jupyter Notebook, Jupyter Lab

- Dataset : MP3 형식의 음원 파일 600여개, 4.7GB

|  |
| --- |
| **2. 프로젝트 개념 설계** |

**가. 아이디어 제시**

◇ 콘텐츠 기반(Content-based) 방식의 데이터 분석을 통해 인기도에 따른 바이어스(Popularity Bias)를 피하고 비슷한 데이터 표본을 찾고, Clustering하는 것을 목표로 한다.

◇ Python의 librosa 패키지를 통해 MP3 Format의 데이터 표본들에 대한 MFCC 등의 특징 값 추출(Feature Extraction)을 해서 각 음원 데이터를 벡터화한다. 이 때, 특징의 개수와 특징에 따라 차원 축소 이후 그래프에 나타난 X, Y 성분 값을 추측해 볼 수 있다.

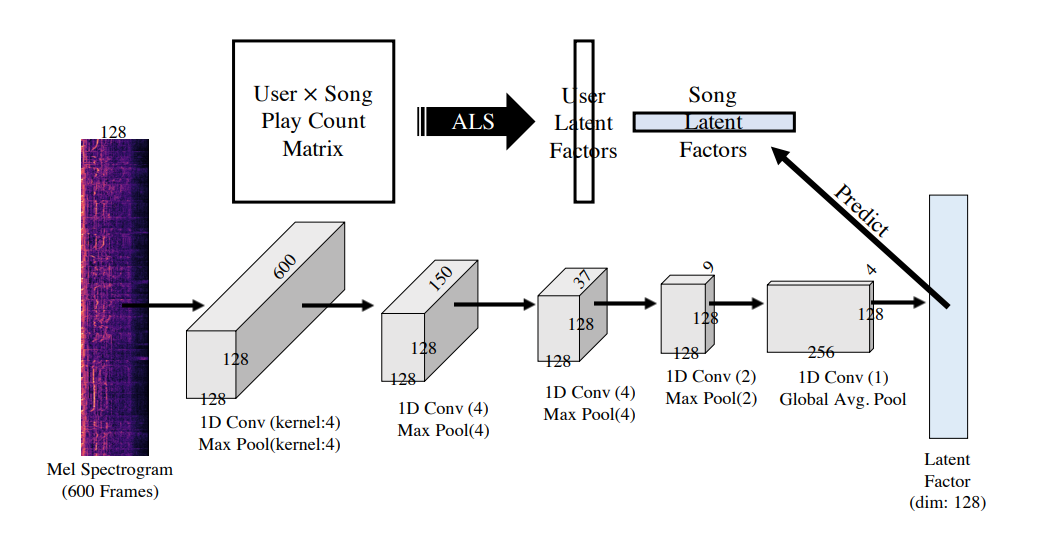
◇ 추출한 음원 벡터를 PCA(Principal Component Analysis)와 T-SNE(T-Distributed Stochastic Neighboring Embedding)기법을 통해 차원 축소한다. 그래프는 2차원으로 표현할 것이므로 이차원으로 축소한다.

◇ Dimensionality Reduction을 거친 음원 벡터들을 K-means 알고리즘을 이용해 Clustering한다.

◇ Clustering 결과를 분석하고 군집이 음원의 특징에 따라 잘 분류되었는지 확인한다.

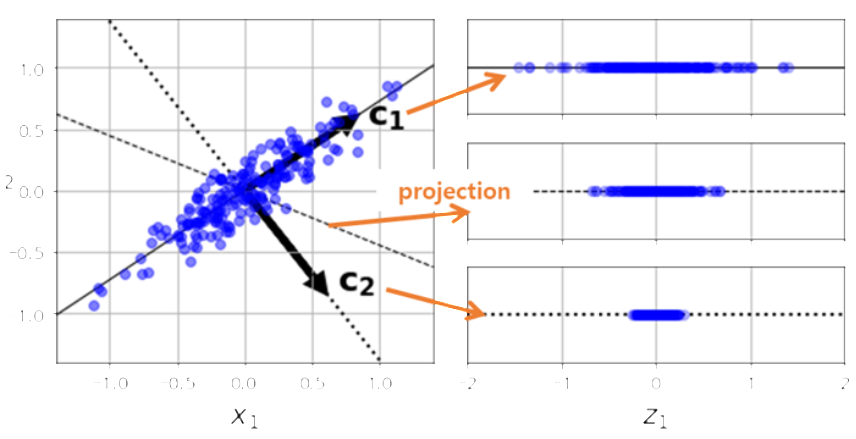
**나. 관련기술의 현황 및 분석**

◇ Content-Based Filtering



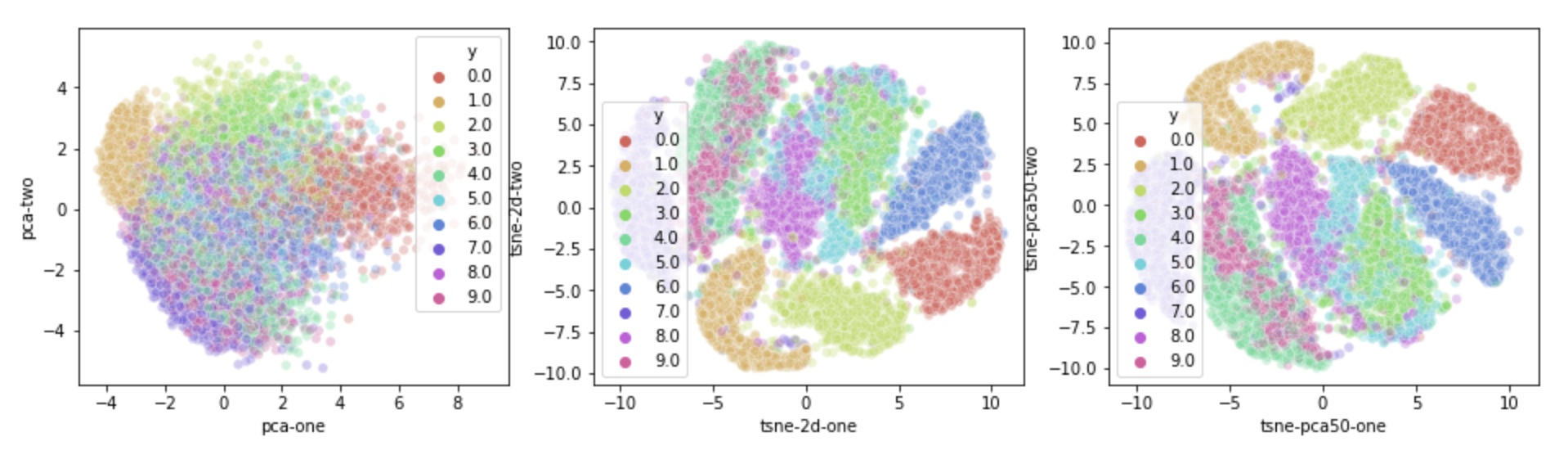
recommendation system에서 content를 기반으로 추천하는 방식으로, 제품끼리 유사한 특징을 기반으로 Clustring한 후에 user profile과 content 선호도를 기반으로 사용자가 선호할 만한 content를 선정해 추천하는 기법이다. content는 큰 변화가 없고 user 평가 전에도 정보가 충분해 cold start문제를 보완해 준다. 이번 프로젝트는 mp3 파일들을 recommendation 하기 전 데이터를 content 기반으로 Clustering 하는 부분까지 구현하는 것이 목표이다.

◇ PCA



고차원의 데이터를 저차원의 데이터로 Dimensionality Reduction하는 기법으로, 주성분(Principal Component) 분석이라고 한다. 원본 Dataset에서 데이터의 분산이 최대가 되는 축(원본 Dataset-투영된 Dataset 간의 평균 제곱거리를 최소화하는 축)을 찾고 이 축과 orthogonal하며 분산이 최대인 다음 축을 찾는 방식으로 Dataset의 차원 수만큼 축을 찾는다. 이런 방식으로 투영하면 데이터의 분산이 최대로 보존되는 점을 이용하여 데이터의 특성은 최대한 보존하며 Dimensionality Reduction를 진행한다.

◇ T-SNE



PCA와 마찬가지로 데이터의 Dimensionality Reduction에 사용되는 기법으로, 시각화에 적합하다. PCA는 선형 분석 방식으로 값을 사상하므로 차원 축소 시 다른 데이터 군집들이 합쳐질 수 있는데, T-SNE는 T분포 그래프를 이용하여 기준점과의 친밀도를 기반으로 데이터를 묶는 방법이다. 단점으로는 특성은 그대로 유지되나 값 자체가 변화한다는 점이 있다.

◇ MIR(Music Information Retrieval)

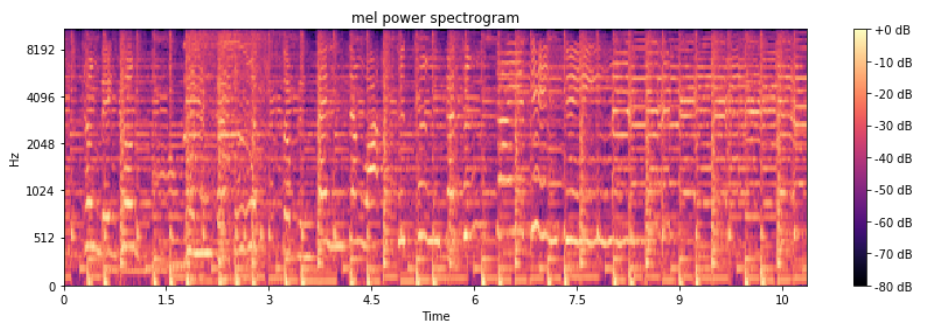
음악을 분류, 조작하는데 사용되는 학문으로 기능 추출을 통해 오디오 컨텐츠 자체를 분석한다. 이를 기반으로 Clustering을 수행하는데 도움을 얻을 수 있다.

◇ K-means Algorithm



주어진 데이터를 k개의 Cluster로 나누는 Algorithm으로, 데이터 집합에서 k개의 데이터 추출해서 Cluster의 Centroid로 설정하고 각 데이터들과 Cluster Centroid와의 유사도를 측정하여 Cluster에 데이터를 할당하는 방식을 반복한다. Cluster가 바뀌지 않을 때까지 수행하여 k개의 Cluster를 확정한다.

◇ Librosa



Librosa는 음원 파일들을 분석할 때 도움을 주는 Python Library로, 특정 특징 값 추출(Feature Extraction)을 하는 함수들을 제공하고 보기 쉽게 시각화하는데 도움을 준다. Mel-scale power spectrogram, MFCC(Mel Frequency Cepstral Coefficient), Chromagram, Tempo, Beats 등의 유용한 함수를 제공한다.

◇ Numpy

C언어로 구현된 Python Library로서 고성능 수치계산과 벡터 및 행렬 연산에 유용한 기능들을 Clustering 과정의 데이터 분석에 도움을 얻기 위해 사용한다.

◇ Pandas

Python을 통한 Clustering 과정의 데이터 분석을 할 때 필수적인 자료구조를 제공하는 Library이다.

|  |
| --- |
| **3. 세부 기술** |

**가. 음원파일의 분석**

우리는 음원 파일을 N차원 벡터로 표현하는데 있어, 음원 데이터를 미가공한 상태로 Dimensionality Reduction하는 것은 크게 의미가 없다고 생각했다. 그래서 음원 파일을 어떻게 분석을 할 것인지 자료 조사를 진행하였다. 음악은 크게 음색(timbral), 박자(temporal), 음조(tonal) 3가지 특징으로 구분할 수 있다는 사실을 발견했고[13], 이 특징을 구분하기 위해 측정 방법들을 찾아보았다.

◇ MFCC(Mel frequency cepstral coefficients)

MFCC는 음성 인식에서 가장 널리 사용되는 알고리즘으로, 실제 유효한 소리의 특징을 추출하는 것을 목표로 한다. 입력된 소리를 일정 구간으로 나누어, 이 구간에 대한 스펙트럼을 분석하여 특징을 추출하는 기법이다.

◇ Spectral centroid, flatness, skewness, kurtosis

스펙트럼 롤오프, 밀도, Flux와 같은 푸리에 변환을 이용하여 시간 기반 신호를 주파수 영역으로 변환하여 얻은 스펙트럼 형상으로, 노트, 피치, 리듬, 멜로디 등을 식별하기 위해 사용하는 지표들이다. 이번 프로젝트에서는 정규분포를 활용한 Spectral kurtosis를 사용할 것이다.

◇ The complex spectral difference onset detection function

음악의 속도는 템포, 타이밍, 운율 구조 등을 통해 계량할 수 있다. 리듬 형상의 연산은 온셋 또는 저수준 특징으로 표현되는데, 스펙트럼 에너지와 결합했을 때 박동 소리 뿐만 아니라 비트 위치 추정에 사용된다.

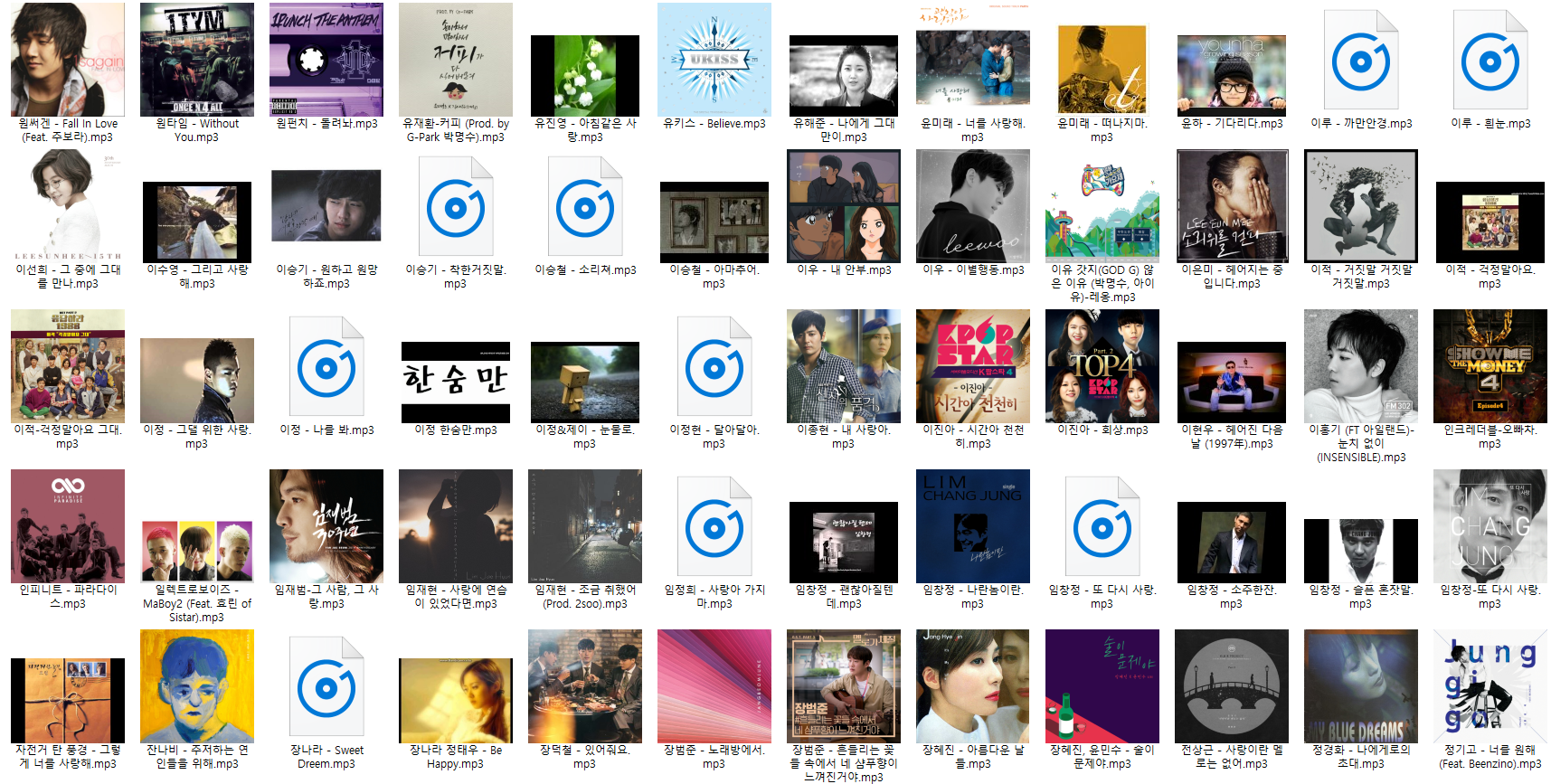
◇ HPCP(Harmonic Pitch Class Profile)

음조는 언어에서 음을 사용하는 것으로, 음정을 사용하여 감정적 및 다른 가언적 정보를 표현할 수 있다. 이를 측정하는 데는 HPCP라는 지수는 사용하며, 12개의 세미톤 피치 클래스의 강도를 나타내는 12차원 벡터로 표현된다.

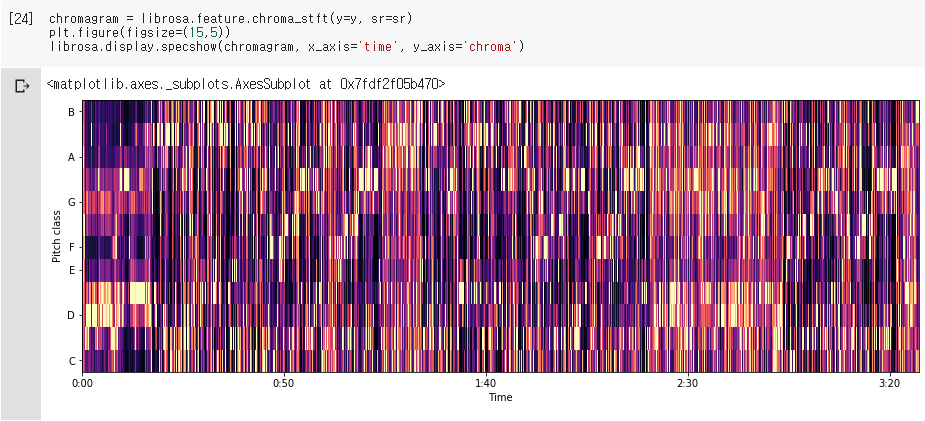
이번 프로젝트에서는 MFCC, Flatness, Spectral Rolloff- Frequency, Tonnetz, Tempo등의 음악적 특징들을 사용하여 음원 벡터들에 대해 분석을 진행할 것이다.

**나. Dataset 수집 및 가공**

◇ Dataset 수집 및 Jupyter lab

****

프로젝트를 진행하는데 필요한 MP3 형식의 음원을 수집하였다. 현재 615개의 음원 파일을 수집하였으며, 군집화에 있어 어려움이 있으면 더 많은 데이터를 수집할 예정이다. 이 파일들을 Google Drive에 업로드 했고 Colab을 이용해 사용하기 위해 마운트했다. 이중 하나의 음원 파일을 분석하기 위해 불러왔다. 이후 음원 파일을 앞서 서술했던 필터 중 하나인 Chromagram을 이용하여 그래프를 그리면 다음과 같이 나타난다.



**다. Dataset 특징 추출**

우리는 음악의 3요소인 음색(timbral), 박자(temporal), 음조(tonal)를 유의미하게 잘 나타낼 수 있는

특징이 무엇이 있을지 조사해 보았고, 6가지 특징으로 각 음원 데이터를 분석했다. 각 특징은 다음과

같다.

◇ Roll-Off Frequency

Roll-Off Frequency는 음원을 일정 구간으로 나누어, 각 frame에서 Frequency의 에너지 값을 판별하는 지표이다. 우리는 roll\_percent를 0이나 1에 가까운 값으로 설정하여 각 음원의 frame마다 최소, 최대 Frequency를 추출했다. 이는 각 frame의 최저음, 최고음을 나타내는 값으로 분석에 사용했다.

◇ Tempo

Tempo(Beat)는 일반적으로 알다시피 곡의 빠르기를 알 수 있는 지표로서 분당 몇 비트로

연주되는가를 BPM으로 나타낸다. 음악에서 BPM은 특정한 장르의 특징을 나타내는 지표이기도

하므로 이를 음원 데이터 특징 추출의 한 요소로 사용했다.

◇ MFCC

MFCC(Mel frequency cepstral coefficients)는 음원을 일정 구간으로 나누어, 이 구간에 대한

스펙트럼을 분석하여 특징을 추출하는 기법이다. 각 frame에 대해 Power Spectrum(주파수 관련 값)

을 계산한 후 이 값에 Mel Filter Bank를 적용하여 주파수들이 얼마만큼의 에너지가 있는지를 측정

한다. 그리고 여기에 로그를 취하면 실제 우리 소리와 유사한 분포를 가진 소리의 특징을 만들 수

있다. 마지막으로 DCT(discrete cosine transform)를 적용하여 에너지들의 상관관계를 분리해서,

유의미한 소리의 스펙트럼 결과 값을 얻을 수 있고 이를 분석에 사용했다.

◇ Tonnetz

Tonnetz는 Tone-Network같은 개념으로 음원을 일정 구간으로 나눈 frame의 코드 진행을 음조의

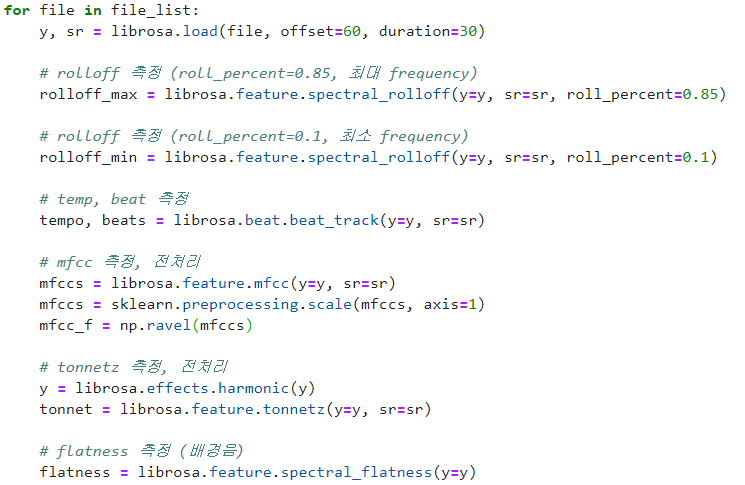
연결 관계로 표현한다. librosa에서는 이를 frame마다 추출해 배열로 저장할 수 있게 함수들을

제공하고 이를 이용해서 분석에 사용했다.

◇ Flatness

Flatness는 음원 데이터에서 음성을 제외한 리듬을 추출한 지표이다. 음악에서 가수의 목소리를

제외한 음원 요소도 장르를 나타내는데 중요한 요소이므로, 우리는 음원을 일정구간으로 나누어

각 구간에 대해 Flatness값을 추출해서 음원 데이터 특징의 한 요소로 사용했다

**라. T-SNE를 이용한 Dimension Reduction**

우리는 Clustering을 하기 위해 앞에서 추출한 음원 데이터의 특징을 차원 축소를 통해 2차원 공간에

표현해보기로 했다. 그런데 음원 데이터를 일정 구간으로 잘라 frame별로 분석했기 때문에 특징 data가

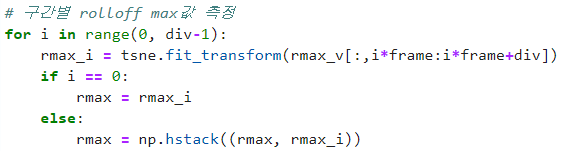
2차원 배열로 나와서 음원별로 모으면 3차원 배열이 되었다. 그래서 2차원 배열로 나오는 각 음원별 특징에

대해 T-SNE방식을 적용해서 음원 데이터의 각 frame마다 실수 값이 나오도록 1차원으로 차원 축소를 했다.

그 후에 음원 데이터에 대해 특징을 다시 모아서 2차원 배열로 만든 후 T-SNE방식을 다시 적용해서

Dimension Reduction을 진행하였다. Python의 Scikit-learn library를 활용해 T-SNE를 수행했다.

◇ 각 특징에 대해 T-SNE 적용



◇ 전체 음원에 대해 T-SNE 적용

****

****

(new\_data는 위의 vector를 파일로 저장하고 불러온 데이터이다.)

**마. K-means Clustering과 시각화**

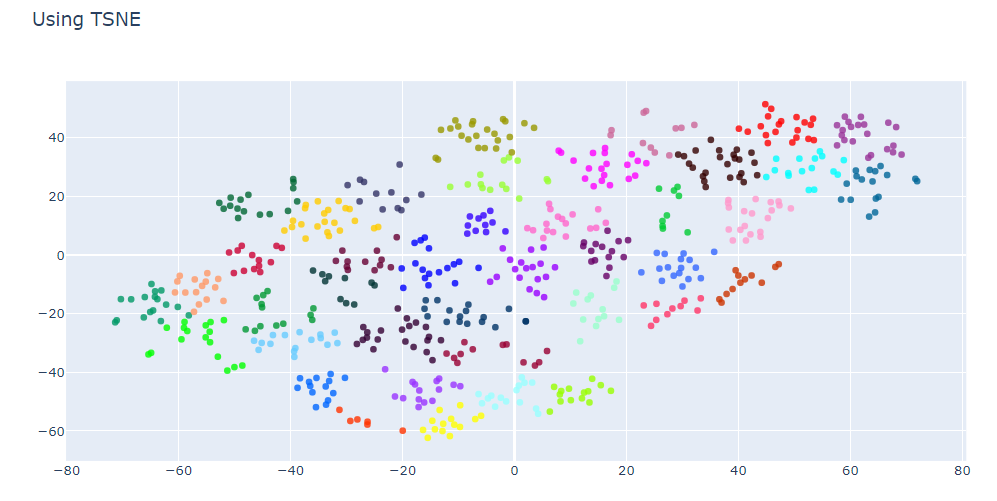
다음으로T-SNE로 Dimension Reduction된 Data를 가지고 K-means Clustering을 통해 유사도가 높은 Data끼리 군집화를 수행해보았다. . Python의 Scikit-learn library를 사용해 K-means Clustering을

수행했다





시각화에는 Python의 Plotly library를 이용했고, 그래프로 가시화한 결과는 다음과 같았다. 각 점은 음원 Data를 의미하고 커서를 올리면 어떤 음원인지 나타나게 해서 분류를 확인할 수 있게 구현하였다. 결과를 가지고 표를 만들어서 클러스터별로 한눈에 Data분류를 확인할 수 있게 만들었다.





|  |
| --- |
| **4. 결과 및 고찰** |

• 음악의 여러 구성 요소를 라이브러리를 통해 low data를 추출하고, 분석, 가공을 통해 음원들을 클러스터링 할 수 있었다. 같은 곡이지만 이름을 다르게 넣었을 때 분석과 클러스터링이 잘 진행되는지 실험하였고, 이를 확인하였다.

• MP3 형식의 파일 뿐만 아니라, 다른 형식의 음원 파일도 분석 가능하게 작성하였다.

• 대체적으로 비슷한 음원이거나, 장르가 같은 음원이거나, 같은 가수가 부른 음원일 경우 같은 클러스터로 묶이는 것을 확인할 수 있었고, 이를 통해 분석이 잘 되었음을 확인하였다.

• 더 많은 데이터셋을 가지고 진행하고 학습을 정확히 진행시켰다면 더 정확한 결과를 얻을 수 있었을 것이다. 하지만 실행환경이 오래된 노트북이었기 때문에 실행 시간이 길었고, 데이터셋을 더 구할 수 있었지만 추가하지 못해서 아쉬웠다.

|  |
| --- |
| **5. 참고자료** |

[1] [https://ko.wikipedia.org/wiki/MP3](http://ko.wikipedia.org/wiki/MP3) - MP3파일의 특징

[2] [https://ko.wikipedia.org/wiki/%EC%B6%94%EC%B2%9C\_%EC%8B%9C%EC%8A%A4%ED%85%9C#Content-based\_filtering](http://ko.wikipedia.org/wiki/%EC%B6%94%EC%B2%9C_%EC%8B%9C%EC%8A%A4%ED%85%9C\#Content-based_filtering) - Content-based Filtering

[3] [https://en.wikipedia.org/wiki/Music\_information\_retrieval](http://en.wikipedia.org/wiki/Music_information_retrieval) - Music Information Retrieval

[4] [https://pdfs.semanticscholar.org/eb07/9679026014cda0a9a240697d3b43f05ef8bb.pdf](http://pdfs.semanticscholar.org/eb07/9679026014cda0a9a240697d3b43f05ef8bb.pdf)

- A content-based music recommender system

[5] [https://dspace.inha.ac.kr/bitstream/10505/18330/1/18618.pdf](http://dspace.inha.ac.kr/bitstream/10505/18330/1/18618.pdf)

- 음악 추천시스템에서 대표 선율을 이용한 내용 기반 음악 필터링 기법

[6] [https://brunch.co.kr/@kakao-it/282](http://brunch.co.kr/@kakao-it/282) - 눈으로 듣는 음악 추천 시스템

[7] [https://blog2.lucent.me/ml/music-recommender-systems-101-intro](http://blog2.lucent.me/ml/music-recommender-systems-101-intro) - 유사곡 추천 101

[8] <http://benanne.github.io/2014/08/https://pdfs.semanticscholar.org/eb07/9679026014cda0a9a240697d3b43f05ef8bb.pdf05/spotify-cnns.html#content-based-recommendatioN>

- Content-Based Recommendation and How to filter music

[9] [https://medium.com/tencent-thailand/music-information-retrieval-part-1-using-librosa-to-extract-audio-features-6e8569537185](http://medium.com/tencent-thailand/music-information-retrieval-part-1-using-librosa-to-extract-audio-features-6e8569537185) - librosa 라이브러리를 이용한 특징 추출하는 법

[10] [https://excelsior-cjh.tistory.com/167](http://excelsior-cjh.tistory.com/167) - PCA(주성분분석)을 통한 차원축소

[11] <http://mac.kaist.ac.kr/~juhan/gct634/2018/finals/deep_content_based_music_recommendation_report.pdf> - DEEP CONTENT-BASED MUSIC RECOMMENDATION

[12] [https://musicinformationretrieval.com/index.html](http://musicinformationretrieval.com/index.html) - Notes on Music Information Retrieval

[13] Schedl, Knees, McFee, Bogdanov, & Kaminskas, 2015, p. 458

[14] [https://librosa.github.io/librosa/feature.html](http://librosa.github.io/librosa/feature.html) - librosa github