스포티파이
개인청취데이터 분석

SELF

WRAPPED

EDA
전처리
분석
번외

1. EDA

```
spotify.T
```

	0
ts	2021-02-05T07:38:44Z
username	31woljgnadjmqxsbalebu5oqo5b4
platform	iOS 14.0 (iPhone12,8)
ms_played	98026
conn_country	KR
ip_addr_decrypted	39.7.231.186
user_agent_decrypted	unknown
master_metadata_track_name	Session 32
master_metadata_album_artist_name	Summer Walker
master_metadata_album_album_name	Session 32
spotify_track_uri	spotify:track:2ktg2oZDyFAX3iY1QNkXl5 spo
episode_name	NaN
episode_show_name	NaN
spotify_episode_uri	NaN
reason_start	playbtn
reason_end	trackdone
shuffle	False
skipped	NaN
offline	0.0
offline_timestamp	1612510625264
incognito_mode	False

ts : 플레이 시작 시간
ms_played : 총 재생 시간
Master_mate_track_name : 곡 이름
Master_meta_artist_name : 아티스트 이름
Maste_meta_album_album_name : 앨범 이름
reason_start : 곡 시작 이유
reason_end : 곡 끝 이유
shuffle : 셔플 여부
skipped : 스킵 여부

1. EDA

```
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   ts                              235495 non-null  object
 1   username                        235495 non-null  object
 2   platform                        235495 non-null  object
 3   ms_played                       235495 non-null  int64
 4   conn_country                    235495 non-null  object
 5   ip_addr_decrypted               235495 non-null  object
 6   user_agent_decrypted            235494 non-null  object
 7   master_metadata_track_name      235401 non-null  object
 8   master_metadata_album_artist_name 235401 non-null  object
 9   master_metadata_album_album_name 235401 non-null  object
 10  spotify_track_uri               235401 non-null  object
 11  episode_name                    31 non-null      object
 12  episode_show_name               31 non-null      object
 13  spotify_episode_uri             31 non-null      object
 14  reason_start                    235495 non-null  object
 15  reason_end                      235495 non-null  object
 16  shuffle                         235495 non-null  bool
 17  skipped                         101104 non-null  object
 18  offline                         235353 non-null  float64
 19  offline_timestamp               235495 non-null  int64
 20  incognito_mode                  235495 non-null  bool
dtypes: bool(2), float64(1), int64(2), object(16)
memory usage: 34.6+ MB
```

전처리 할 항목

-곡 제목/가수명 누락 행 삭제
-팟캐스트 관련 항목 삭제
-재생환경 정보 삭제(OS,IP 등)
-플레이 시작시간 년/월/일/시/분/초 분해
-ms_played > second로 변환

2. 전처리 - ts

```
spotify.ts
0          2021-02-05T07:38:44Z
1          2021-02-05T07:42:10Z
2          2021-02-05T07:43:52Z
3          2021-02-05T07:46:44Z
4          2021-02-05T07:50:36Z
                   ...
```

year	month	day	hour	minute	second	weekday
2021	2	5	16	38	44	금

```
from datetime import datetime, timedelta

date_string = list(df.ts)

#UTC > KST
date_objects = list(map(lambda x : datetime.fromisoformat(x) + timedelta(hours=9),date_string))

new_date_string=list(map(lambda x : x.strftime("%Y-%m-%d %H:%M:%S"), date_objects))

df.ts = new_date_string

date_time_str = list(df.ts)

date_=list(map(lambda x : datetime.strptime(x, '%Y-%m-%d %H:%M:%S'),date_time_str))

a=[]
for i in range(len(date_)):
    a.append(datetime.date(date_[i]).weekday())
```

날짜
UTC > KST로 변환
년, 월, 일, 시, 분 초 컬럼 생성

2. 전처리 - ms_played

```
df['ms_played'] = (df.ms_played/1000).astype(int)
df.rename(columns={"ms_played":"s_played"},inplace=True)
```

1. 밀리세컨드 > second 변환

```
df.dropna(subset=['master_metadata_track_name'], inplace=True)
df.drop(df[df.s_played==0].index,inplace=True)
df["SA"] = df.master_metadata_track_name +"-"+ df.master_metada
곡길이=df.groupby('SA')['s_played'].transform(lambda x: x.mode()
df['곡길이'] = df.groupby('SA')['s_played'].transform(lambda x:
df['곡길이'] = ((df['곡길이']).astype(int)//60).astype(str) +'m '+
df.곡길이
0          1m 38s
1          3m 26s
```

2.원본 데이터에 곡 길이정보X

-ms_played 컬럼을 활용하여 곡의 최빈값을 기본 길이로 간주
(재생시간의 최빈값이라면 곡의 러닝타임과 같음을 이용)

-최빈값이 여러개인 경우 가장 긴 값을 기본 길이로 간주
(이 경우 실제 곡 러닝타임과 차이가 있을 수 있음)

	spotify.ms_played
0	98026
1	206080
2	100383
3	173053
4	231253

	df.곡길이
0	1m 38s
1	3m 26s
2	3m 23s
3	0m 1s
4	3m 51s
	...

3. 전처리 – 스킵여부 결정

```
spotify.skipped.isnull().sum()
```

```
134391
```

원본 데이터 skipped 컬럼 결측치가 절반 이상
스킵의 기준도 알 수 없음

2.9 skipped

```
df.skipped = False
```

```
condition = df.s_played < 20
```

```
df.loc[condition, 'skipped'] = True
```

자체적으로 곡 재생시간이
20초가 안될경우 스킵이라 간주.

재생시간이 0~1초인경우는 듣다가 스킵이 아닌 곡을 찾기위해 넘긴 경우로 간주
> 삭제

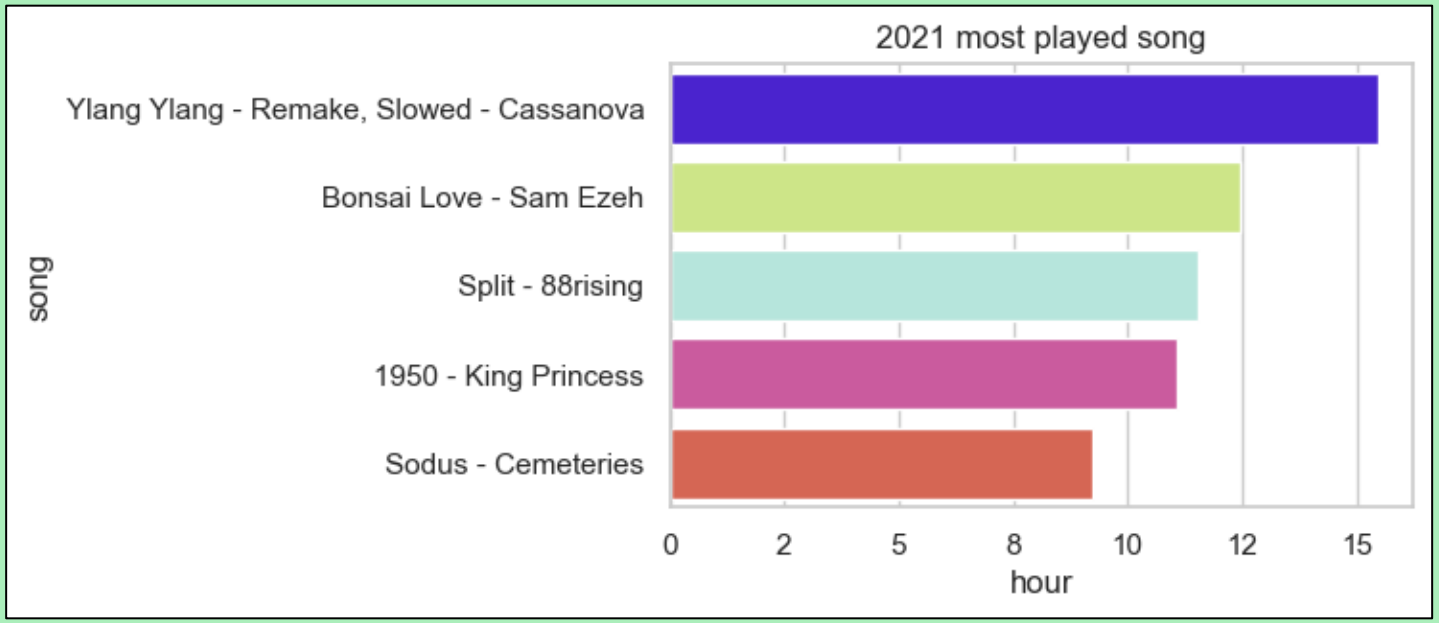
3. 전처리 – 장르추가

```python
import pylast

API_KEY = ''
API_SECRET = ''

network = pylast.LastFMNetwork(api_key=API_KEY, api_secret=API_SECRET)

def get_artist_genre(artist):
    try:
        artist_obj = network.get_artist(artist)
        top_tags = artist_obj.get_top_tags()
        if top_tags:
            genres = [tag.item.get_name() for tag in top_tags]
            return genres
        else:
            return None
    except pylast.WSError as e:
        # 아티스트 정보를 가져오는 중에 오류가 발생한 경우
        print(f"Error fetching info for artist: {artist} - {e}")
        return None
```

```
['k-pop',
 'rnb',
 'pop',
 'electronic',
 'Hip-Hop',
 'indie',
 'Korean',
 'indie pop',
 'soul',
 'Lo-Fi',
 'dream pop',
 'female vocalists',
 'rap',
 'country',
 'seen live',
 'jazz',
 'ambient',
 'chillwave',
 'synthpop',
 'piano',
 'indie rock',
 'House',
 'shoegaze',
 'hip hop',
 'dance',
 'funk',
 'french',
 'trance',
 'minimal',
 'electro',
```

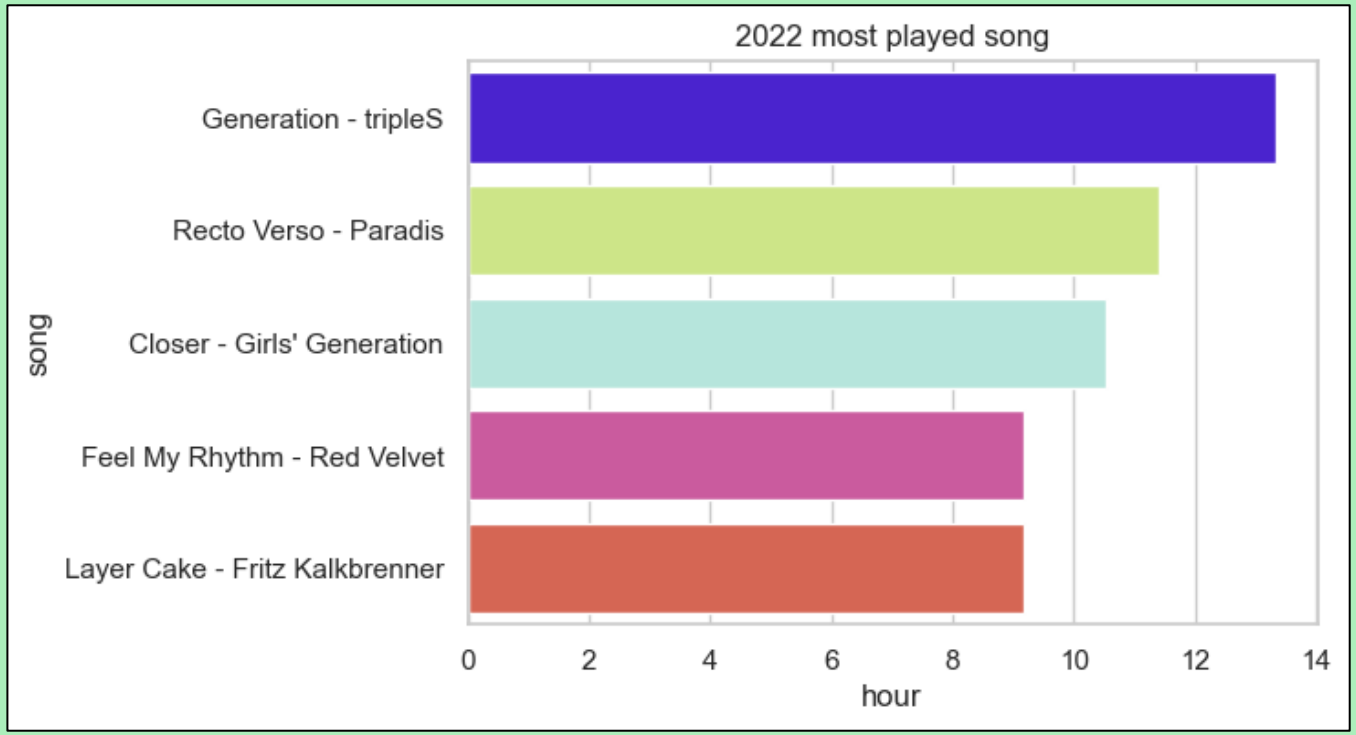
원본데이터 곡 장르정보 X
Last.fm API 활용하여 곡 장르 추가

4. 분석 – 2021년



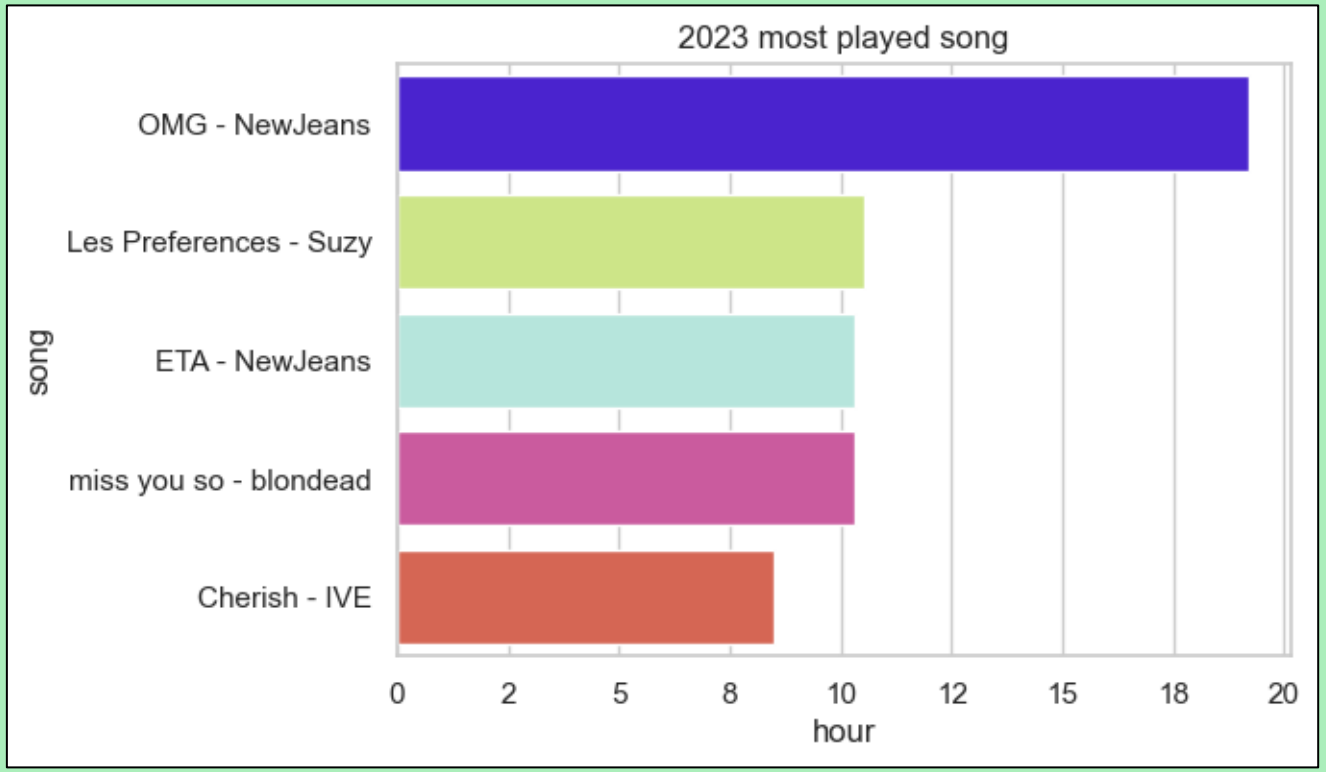
1 청취시간 : 72일
2 러닝타임 긴 가수 : LanaDelRey(61시간 26분)
3 러닝타임 긴 곡 : ylang ylnag(15시간 46분)
4 러닝타임 긴 장르 : RnB
5 러닝타임 긴 요일 : 목요일(평균 6시간 1분)
6 러닝타임 짧은 요일 : 토요일 (평균 4시간 17분)
7 러닝타임 긴 월 : 4월 (총 11일)
8 하루평균 재생시간 : 5시간 21분

4. 분석 – 2022년



1 청취시간 : 73일
2 러닝타임 긴 가수 : Red Velvet(49시간 16분)
3 러닝타임 긴 곡 : tripleS – generation (13시간 35분)
4 러닝타임 긴 장르 : Kpop
5 러닝타임 긴 요일 : 수요일 (평균 5시간 2분)
6 러닝타임 짧은 요일 : 토요일 (평균 4시간 12분)
7 러닝타임 긴 월 :3월 (총 6일 19시간)
8 하루평균재생시간 : 4시간 46분

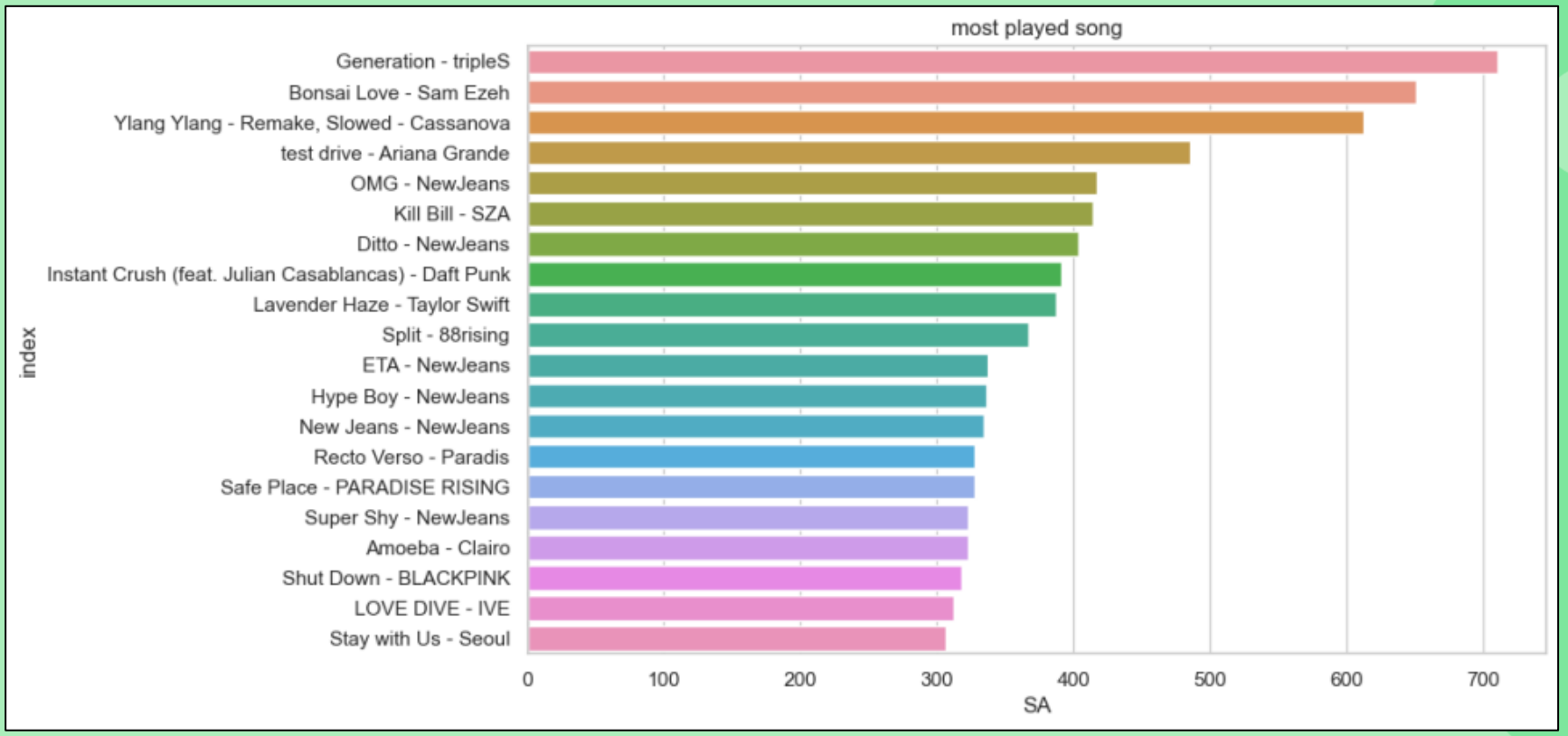
4. 분석 – 2023년



*데이터 요청시점 2023년 10월
>2023년 데이터가 다 모이지 않았음

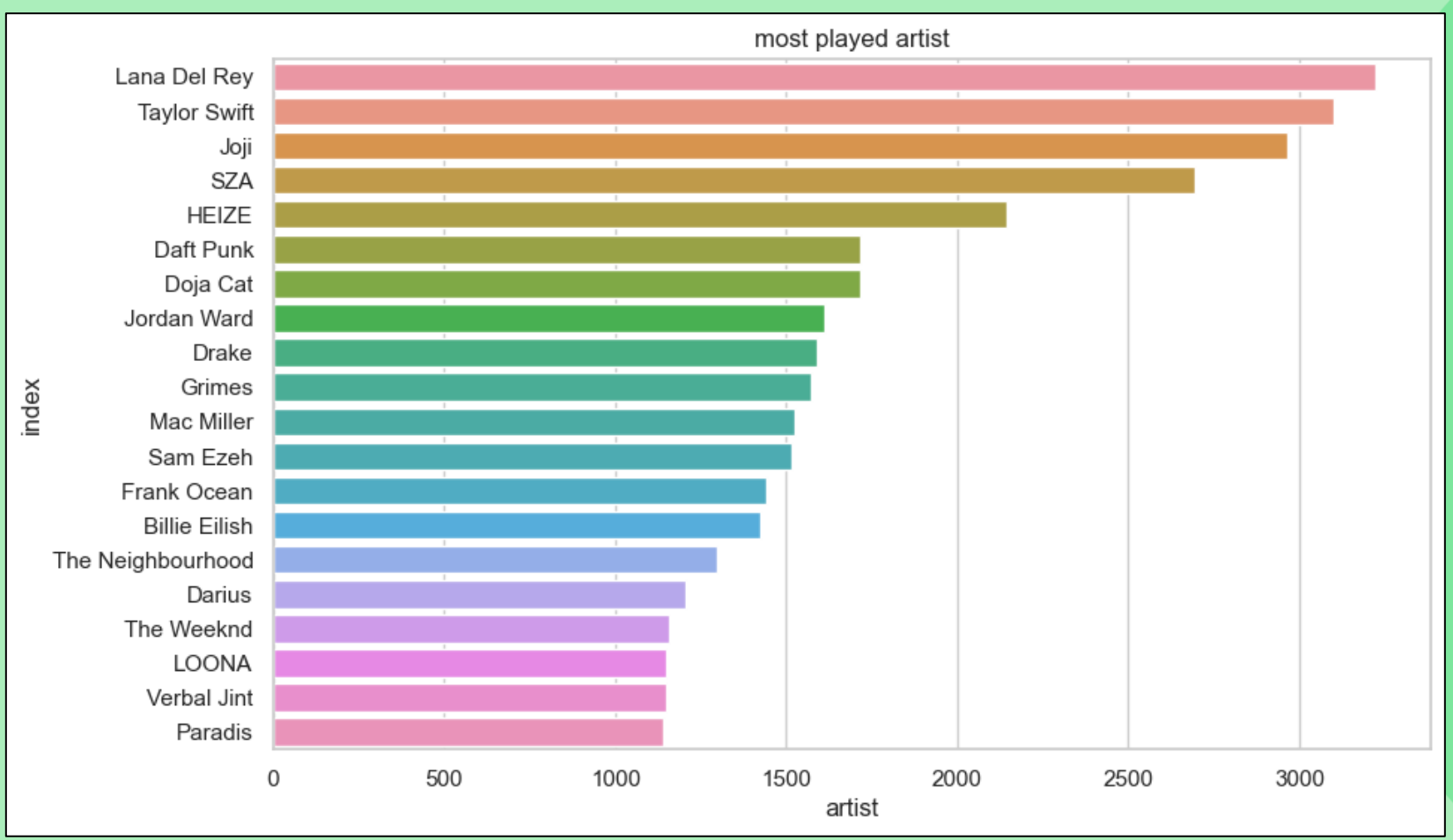
1 청취시간 : 65일
2 러닝타임 긴 가수 :NewJeans (64시간 40분)
3 러닝타임 긴 곡 : Newjeans-OMG(19시간 19분)
4 러닝타임 긴 장르 : Kpop
5 러닝타임 긴 요일 : 수요일(평균 6시간 11분)
6 러닝타임 짧은 요일 : 토요일(평균 3시간 6분)
7 러닝타임 긴 월 : 1월 (총 9일 2시간)
8 하루평균재생시간 : 5시간 14분

4. 분석 – 3년 합 재생 횟수 (곡)



Generation - tripleS
총 700번 이상 재생

4. 분석 – 3년 합 재생 횟수 (가수)

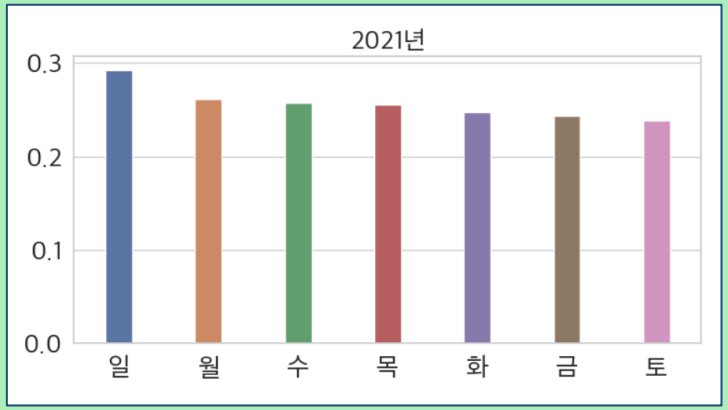


Lana Del Rey
총 3000번 이상 재생

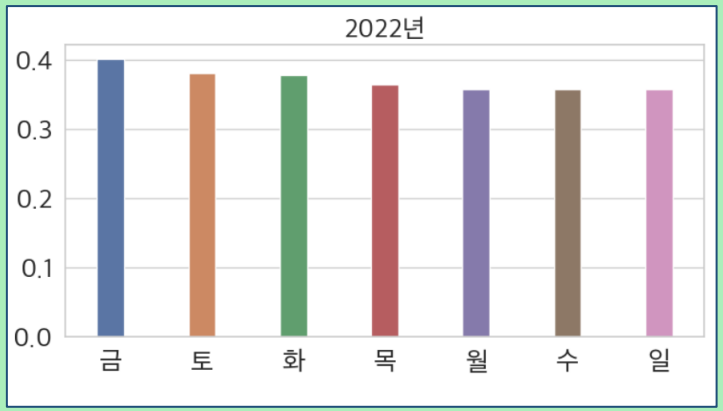
4. 분석 – 스킵

스킵비율 순위 그래프

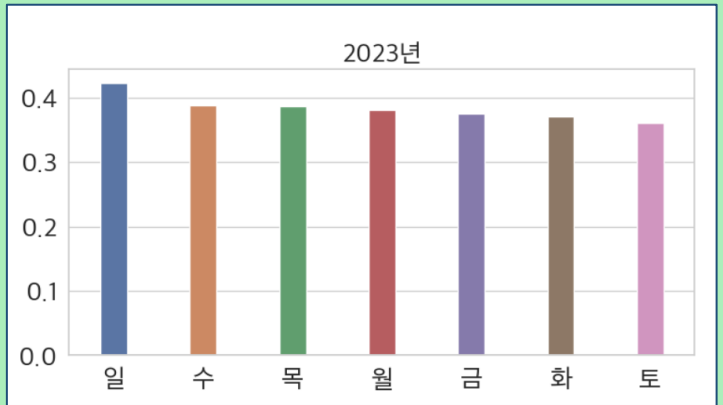
요일별 총 재생 곡 수
ex)2021년 모든 일요일 합 6909곡 재생



	weekday	song	s_played	스킵곡수	스킵비율
4	일	6909	821382	2018	0.292083
3	월	6946	859027	1817	0.261589
2	수	7171	910735	1849	0.257844
1	목	7878	996030	2008	0.254887
6	화	7546	955782	1869	0.247681
0	금	7776	995709	1893	0.243441
5	토	5554	725271	1325	0.238567



	weekday	song	s_played	스킵곡수	스킵비율
0	금	9602	876164	3849	0.400854
5	토	8355	802411	3181	0.380730
6	화	9420	893289	3561	0.378025
1	목	9337	912236	3402	0.364357
3	월	9590	937152	3433	0.357977
2	수	9396	943651	3362	0.357812
4	일	8873	896230	3170	0.357264



	weekday	song	s_played	스킵곡수	스킵비율
4	일	6021	504223	2543	0.422355
2	수	10519	956606	4085	0.388345
1	목	9971	925925	3861	0.387223
3	월	9836	907574	3741	0.380338
0	금	9933	911530	3728	0.375315
6	화	10177	947367	3770	0.370443
5	토	4860	459167	1750	0.360082

2021년, 2023년은 주말 아르바이트 진행
>주말 재생 곡 수가 적은 이유

일요일이 재생 곡 수가 대체적으로 적지만
스킵하는 비율은 제일 높다
>아르바이트 하고 곡 선정에 까다로워 지는걸까..?

4. 분석 – 나의 음악 청취 습관

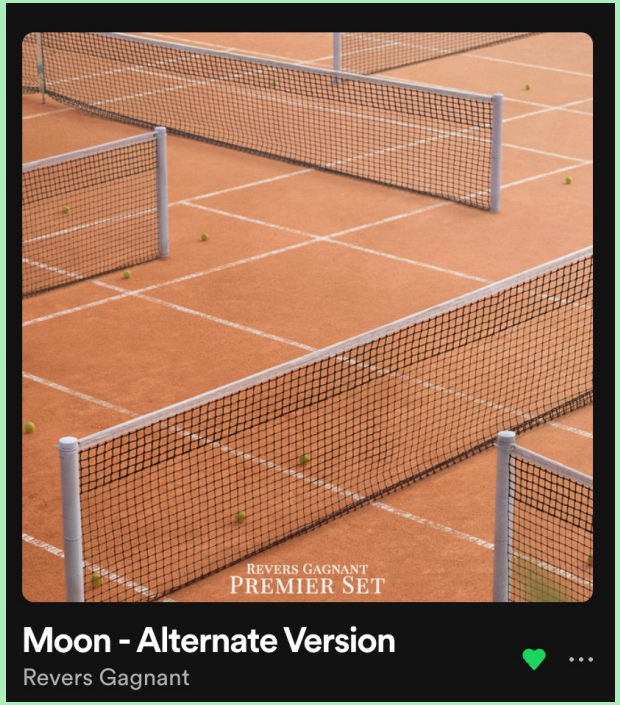
1. 수요일에 가장 오래 듣는다
2. 토요일에 가장 적게 듣는다
3. 하루에 5시간은 듣는다
4. RnB와 Kpop을 제일 선호한다
5. 일요일엔 스킵하고 넘기는 경우가 많다
6. 700번 넘게 들은 곡이 있다
7. 3000번 넘게 들은 가수가 있다
6. 41분 더 들으면 Newjeans의 OMG은 도합 20시간을 듣게된다 (도전..)

5. 번외 – 연속재생TOP3

Ylang ylang (Remake, Slowed) **1위!**

2021년 12월 8일 0시 1분 ~ 3시 23분까지
연속으로 116번 재생

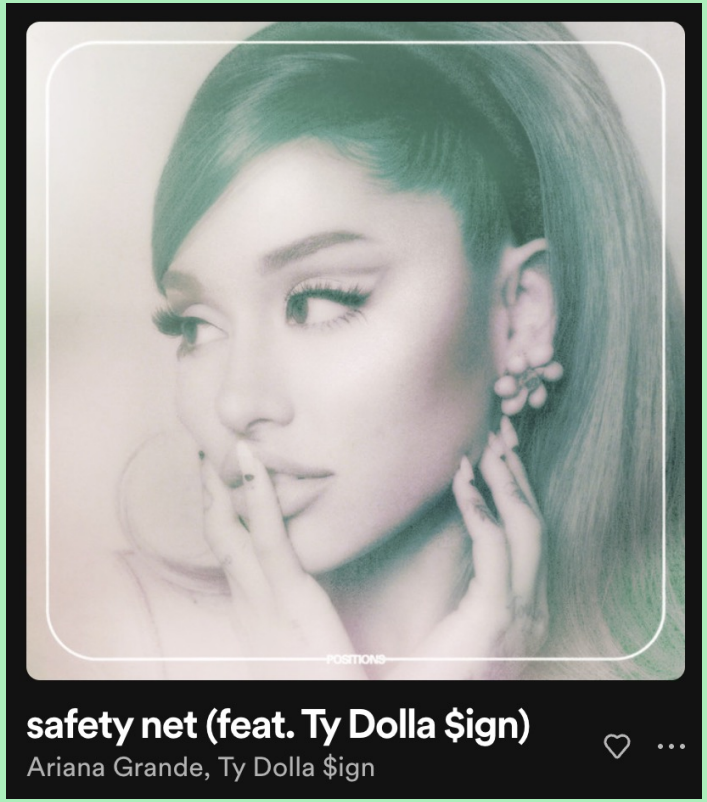
5. 번외 – 연속재생TOP3



2위!

2023년 3월 21일 12시 34분 ~ 13시 46분까지
연속 46회 재생

5. 번외 – 연속재생TOP3



2021년 5월 13일 10시 35분~12시 9분까지

연속 43회 재생

3위!